UNIVERSITY OF BERGAMO

School of Doctoral Studies

Doctoral Degree in Engineering and Applied Sciences

SSD:ING-INF/05 - Information Processing System

35° Cycle – PhD scholarship funded within the scope of the collaboration agreement Regione Lombardia-ENEA (DGR 7792/2018)

From Big Data to Smart Data for City Street Lighting Analytics

Supervisor:

Prof.ssa. Patrizia Scandurra

Co-Supervisors:

Fabio Moretti, Ph.D.

Author:

Mubashir Ali

Student ID: 1067637

ACADEMIC YEAR 2021 / 2022

To my parents, whose sacrifice in every aspect of life enabled me to execute as best as I could. They have all the way been behind me through their prayers to encourage me fulfilling the necessary requirements during this tedious process, making me more energetic and enthusiastic to successfully ending up this work in order to complete my PhD program and to get an opportunity to see you people again after a long time.

To my dearest wife for all her support and prayers for ending up this work in a graceful way. It would not have been possible for me to achieve these goals without her caring support and love throughout the journey. She has been very much supportive whenever I was desperate or found myself hopeless in the dark patches, during all the way to destination.

To my brothers and sisters, whose financial, moral, and ethical support has never let me divert my attention to anything else that might lead me astray somewhere far away from where I am standing today. I have always been thankful to them for their supporting behavior throughout my academic carrier even when I was quite young having a limited domain of knowledge. They paved my way to excellence through their prayers, enlightened thoughts, and broader vision that they had gained with time.

Acknowledgment

This thesis would not have been possible without the guidance and support of many important people, who have continuously been a source of courage and inspiration to accomplish this work.

First of all, I would like to thank my supervisor, Prof.ssa Patrizia Scandurra, that with her passion for research inspired me to become a curious (and hopefully better) researcher. She also helped me out learning the real way how an early career researcher should take a step forward. She provided me with continuous and invaluable support during my Ph.D. studies, and I am incredibly grateful to have shared with her an immense amount of time discussing about research work and writing papers. I am thankful to her for the mistakes she has pointed out, which have been a source of continuous and quick learning for me during these days.

I would like to thank my co-supervisor Dr. Fabio Moretti. His deep expertise in machine learning and data science helped me to think systematically about problems and solutions. Here, I acknowledge that it could not have been possible without the selfless support and availability of my supervisors. I admire the way they obliged me offering their valuable time and suggestions whenever needed.

I would also like to thank Prof. Giuseppe Psaila from the department of Engineering and Applied Sciences, who also guided well in accomplishing different research activities. I always found him courteous and always available whenever he was requested.

I would like to extend my sincere thanks to my friends, Asad Hussain and Anees Bqir for helping me during the progress of my work with their constant comments and suggestions for improvement of the research.

I would like to express my heartfelt gratitude to my brother Dr. Hafiz Husnain Raza Sherazi, for their unwavering support and kindness throughout my journey. Their guidance and encouragement have been invaluable, and have played a significant role in my success. Thank you for always being there for me, and for being such a wonderful person. Your generosity and compassion have truly made a positive impact on my life, and I am forever grateful.

During this period, I got an opportunity to visit Czech Institute of Informatics, Robotics and Cybernetics (CIIRC), Czech Technical University (CTU) Pragure for for a short period under the supervision of Dr. Martin Macas. Dr. Martin is really an inspiring personality and fantastic human being. He supported me in every possible way, from officially supporting my exchange application to helping me out with finalizing the research directions carried out during my stay in Prague.

Grabbing this opportunity, I also acknowledge the selfless support and encouragement of all my teachers throughout my academic career. Especially, Muhammad Haneef Saleemi, a senior teacher from my high school, and my master's research supervisor at Bahria University, Prof. Dr. Shehzad Khalid, who contributed altruistically to make it happen. They have always been cooperative during these years of higher studies to make me learn as best as they could. They not only practically helped me out but also supported me morally with their ongoing prayers during these years.

I would also like to mention all friends I met in Bergamo, that made me always feel at home, particularly Mohammad Latif Bhatti, I can not forget the super delicious food of their home, and the cherished time spent with Asad Hussain, Umair Shahab, Waseem Raziq, and Italian friend Nicola Cortesi.

Last, but not the least, I am also indebted to the Reviewers, Prof.ssa Raffaela Mirandola and Prof.Mauro Caporuscio, for their encouraging comments and useful suggestions which made it possible for me to improve the quality of this thesis.

Thank You all.

Mubashir Ali

Contents

List of Figures

List of Tables

Acronyms

SCP smart city platform

KPIs Key Performance Indicators

IoT Internet-of-Things

ML Machine Learning

ICT Information and Communication Technologies

BD Big Data

GPS Global Positioning System

IP Internet Protocol

ZB zettabyte

JSON JavaScript Object Notation

XML Extensible Markup Language

HDFS Hadoop Distributed File System

YARN Yet Another Resource Negotiator

SPOF Single Point Of Failure

POSIX Portable Operating System Interface

JVMs Java Virtual Machines

APIs Application Programming Interfaces

APIs Application Programming Interfaces

RM	ResourceManager
AM	ApplicationMaster
NM	Node Manager
RAM	Random Access Memory
CPU	Central Processing Unit
RDD	Resilient Distributed Dataset
SQL	Structured Query Language
MLlib	Machine Learning Library
BDA	Big Data Analytics
SC	Smart City
DL	Deep Learning
BI	BUsiness Intelligence
MISE	Ministry of Economic Development
QoS	Quality of Services
AI	Artificial Intelligence
POD	Points of Delivery
LV	Low Voltage
MQTT	Message Queue Telemetry Transport
RDBMS	Relational Database Management System
GPP	Green Public Procurement
EC	Energy Consumption
MP	Minimum Power
DECD	Daily Energy Consumption Deviatio

ADEC Actual Daily Energy Consumption

TDEC Teoratical Daily Energy Consumption

AECD Annual Energy Consumption Deviation

MAECD Maximum Annual Energy Consumption Deviation

OOP Object Oriented Programming

CDR Call Detail Records

NYC New York City

LOF Local Outlier Factor

OC-SVM One-Class Support Vector Machine

DBMSCAN Density-Based Micro Spatial Clustering of Applications with Noise

KNN K-Nearest Neighbors

SVM Support Vector Machine

LR Logistic Regression

MLP MultiLayer Perceptron

LSTM Long Short-Term Memory

CNN Convolutional Neural Netwrok

RBM Restricted Boltzmann Machine

DBN Deep Belief Networ

DNN Deep Neural Network

kWh kilowatt-hours

DBSCAN Density-Based Spatial Clustering of Applications with Noise

OPTICS Ordering Points to Identify the Clustering Structure

MinT Minimum Threshold

MaxT	Maximum Threshold
PELL	Public Energy Living Lab
NN	Neural Network
IQR	Interquartile Range
ARIMA	Autoregressive Integrated Moving Average
RF	Random Forest
FFNN	Feed Forward Neural Network
CART	Classification and Regression Tree
SVR	Support Vector Regression
ANNs	Artificial Neural Networks
EvoHyS	Evolutionary Hybrid System
R	Correlation Coefficient
MAPE	Mean Absolute Percentage Error
MAE	Mean Absolute Error
RMSE	Root Mean Squared Error
MaxAE	Maximum Absolute Error
RT	Regression Tree
STLF	short-term load forecasting
ReLU	rectified linear unit
ANN	Artificial Neural Network
LReLU	leaky rectified linear unit
PReLU	parametric rectified linear unit
ELU	exponential linear unit

GRU Gated Recurrent Unit

GBM Gradient Boosting Machine

XGB Extreme Gradient Boosting

HVAC Heating, Ventilation, and Air Conditioning

AR Autoregressive

RBP Rule-Based Predictor

DT Decision Tree

NP Naive Predictor

HC Historic Consumption

Abstract

Applications of Big Data (BD) to smart cities are nearly limitless. Big Data Analytics (BDA) is one of recent technologies that has a huge potential to enhance smart city services by transforming city information into city intelligence. As growing digitization becomes perfectly integrated to our daily lives, enormous heterogeneous amounts of data collected everyday can be fruitfully used in several application domains such as transportation, healthcare, energy management, environmental monitoring, etc. Public lighting energy management is one of the critical problems for city managers. It is a domain with a huge number of energy consumers, especially public buildings (such as government, health, and educational institutes) and streets. Another important concern is that energy demand is expected to increase in the near future due to climate change, population growth and inefficient usage of energy.

Although BD technological solutions make the collection and availability of very large amounts of data possible, they also demand for new paradigms for massive data and models that go beyond processing, storing and accessing records rapidly. However, the transition from BD to smart data providing insights about information and issues that matter is not simple and obvious to achieve. The greater the amount of data and the more heterogeneous they are, the more complex their processing will be. New challenges related to scalability, quality, processing and visualization have emerged with handling and structuring data with new magnitudes of dimensionality, complexity, heterogeneity, and timeliness.

In the light of these challenges, in this dissertation, we design and implement a Smart Data-centric software architecture for the analysis of energy consumption data in the context of public street lighting. Though the increasing number of smart city initiatives adopting BDA, these technologies have not been exploited enough in this domain. The proposed Smart City Platform (SCP) is developed in collaboration with the Italian national research agency ENEA[1] as part of the (Public

[1]ENEA Italian National Agency for New Technologies, Energy and Sustainable Economic De-

Energy Living Lab)PELL[2] project; in brief: the PELL SCP. By leveraging on BD technologies, we have designed and prototyped different types of analytics services as part of the PELL SCP; these include: Key Performance Indicators (KPIs) to measure and monitor electric energy consumption of public street lighting plants, detection of anomalies from energy consumption data, and forecasting street light energy consumption.

Evaluation is performed through an extensive set of experiments on real data sets about street lighting energy consumption, which were collected and managed through the PELL SCP. The results about the efficacy and performance of the proposed analytics framework showed that the PELL SCP enriched with such analytical services can cope promisingly with the processing, analysis, and scalability issues of such BD.

velopment `https://www.enea.it/en`

[2]`https://www.pell.enea.it/enea/`

Chapter 1

Introduction

Several cities in the world are seeking to become "smart" by managing urban processes and services for citizens via Information and Communication Technologies (ICT)-based solutions established by municipalities and business service providers. Recently, there is a growing trend in considering a *citizen-centric* concept of Smart City and in adopting software platforms [1, 2, 3], which can aggregate as much information and "polymerize" the city services realized via heterogeneous siloed ICT-based solutions by public-private partnerships in specific business domains (such as, waste management, city surveillance, energy consumption, parking, public transport, etc.). These initiatives are giving rise to the concept of *smart city as a platform* where citizens can access the services easily and uniformly thanks to end-to-end Application Programming Interfaces (APIs), and service providers can easily realize new value-added city services thanks to one source of aggregated information. Such software platforms for smart cities typically make use of emerging technologies related to Internet-of-Things (IoT), cloud computing, and BD management [4]. IoT-Smart City ecosystems are made of sensors and smart city apps ("citizens as sensors") that generate large amounts of heterogeneous data in real time and space, such as information from city infrastructure monitoring, performance and health events from used tools, and application execution logs. These urban data streams are vital information that can be processed to provide novel services and manage or optimize different concerns of a smart city.

BDA is an emerging technology that has a huge potential to enhance smart city services by transforming city information into city intelligence. As growing digitization becomes perfectly integrated to our daily lives, enormous heterogeneous amounts of data collected everyday can be fruitfully used in several application domains [5] and be very helpful to city governments [6, 7] to analyse trends, measure and optimize business performance. Moreover, data analytics techniques (such as statistical inference and machine learning) can play a vital role in further improving the business performance by transforming information into intelligence capable of making data-driven decisions [8, 9]. Development of intelligent smart services in the Smart City context is nearly limitless. Lopez et al. [10] have identified and reviewed existing intelligent business and management models in the BD era. They stated that "Intelligence" is understood as a process of gathering, analyzing, interpreting, and disseminating high-value data and information at the right time for use in the decision-making process.

Public lighting management is one of the most important concern of the city managers [11]. The digitalization of energy industry and the introduction of the smart grid are allowing energy suppliers to collect an enormous amount of energy data. This data contains significant information that can be used by the administrative authorities for accounting and decision making. Moreover, analysis of energy data can also provide valuable insights for both consumers and suppliers. Public lighting is a domain with a huge number of energy consumers, especially public buildings (such as government, health, and educational institutes) and streets [12]. Another important concern is that energy demand is expected to increase in the near future due to climate change, population growth and inefficient usage of energy [13]. Consequently, there is a high interest in leveraging energy consumption data to build understanding of the consumption behaviors and inefficiencies. This is not only significant for efficient energy management, but also for the impact of energy on the environment and its importance in the economic gains.

However, though BD technological solutions make the collection and availability of very large amounts of data possible, they also demand for new paradigms for massive data and models that go beyond processing, storing and accessing records rapidly. The real benefit of BD is not on the data itself, but in the ability to uncover interesting hidden patterns and derive knowledge from it by using appropriate data mining techniques [14]. The term Smart Data refers to the challenge of transforming (unstructured) raw data into quality data that can be appropriately utilized to achieve valuable insights [15]. Therefore, smart data can be described as BD that has been cleansed, filtered, and prepared for advanced analytics.

New challenges of scalability, quality, processing and visualization have emerged with handling and structuring data with new magnitudes of dimensionality, complexity, heterogeneity, and timeliness [16, 17]. These new facing challenges, along with the lack of best practices and of reported development experiences may increase sense of confusion about the use of the BD technology and the effectiveness of the process for gaining valuable insights from data, thus hindering their application for driving better business and shaping smarter city services.

In the light of these challenges, in this dissertation, we design and implement a Big Data-centric software architecture for the analysis of big energy consumption data in the context of public street lighting. Though the increasing number of smart city initiatives adopting BDA, these technologies have not been exploited enough in this domain [18]. The proposed SCP is being developed in collaboration with the Italian national research agency ENEA[1] as part of the PELL[2] project; in brief: the PELL SCP. By leveraging on BD technologies, we have implemented different analytical services such as desrciptive analytics, and predictive analytics. These include KPIs to measure and monitor electric energy consumption of public street lighting plants, detection of anomalies from time series street lighting energy data, and forecasting of street light energy consumption. All these services are implemented as as part of the PELL SCP analytics framework.

1.1 Problem Definition

The main research question that we tackle in this dissertation is formulated as follows:

RQ: How to endow a smart city platform collecting big street lighting data with data analytics features?

From the characteristics of applications of street lighting domain, we identify the following research objectives:

O1: Exploration from a software-architecture perspective of a system engineering approach to architect a smart-city platform with BDA.

[1]ENEA Italian National Agency for New Technologies, Energy and Sustainable Economic Development `https://www.enea.it/en`

[2]`https://www.pell.enea.it/enea/`

O2: Providing descriptive analytics through formulation and implementation of KPIs to analyze, measure and monitor electric energy consumption of public street lighting plants.

O3: Devise a predictive analytics for the detection of anomalies from time series of street lighting energy consumption data.

O4: Empirical investigation of predictive models to select a suitable forecasting model for street lighting energy consumption.

1.2 Scientific Approach

The scientific approach of this dissertation is based on the design science research methodology [19]. The design process (see the dotted boxes in Figure 1.1) of this thesis encompasses five steps: problem identification and motivation, definition of the objectives for a solution, solution design and development, demonstration, and evaluation. The very first phase of the design process defines the specific research problem and explains the value of a solution. We analyze the research background and related work that affect our main research questions as shown by the solid boxes in Figure 1.1. Then we identify the four main research objectives for the definition of a solution. To achieve the desire goal, we use different research methods for each of the research objectives.

In order to address **O1** and gain a systematic understanding of the latest research made for the development of smart-city platforms with BDA, we perform a systematic literature review by following the guidelines from [20]. A systematic literature review is an established approach to collect and analyze data within a specific area of interest. To address **O2**, we follow the outcomes of **O1** to recognize most common architectural styles to design and implement a Big Data-centric software architecture for tracking energy consumption data and census data in the context of public street lighting. We adopt Apache Spark ecosystem for large-scale data processing and analytics. It helps us towards turning BD into Smart Data by presenting some KPIs to measure and monitor electric energy consumption of public street lighting plants. To address **O3** and design a predictive analytics-based approach for the detection of anomalies from time series street lighting energy data, we adopt unsupervised learning techniques. This approach enables us to detect the anomalies from time series energy data without training the model on historic energy data. To address the last objective **O4** of the dissertation, we select ML-based forecasting models for empirical evaluation. We also develop a rule-based predictor for street lighting energy consumption forecasting and a comparative analysis helps to determine the most suitable model for street lighting energy consumption forecasting. In order to carefully evaluate the performance of developed services, see Figure 1.1, we perform an extensive set of experiments on real street lighting energy data collected and managed through the PELL SCP.

1.3 Contributions

By fulfilling the research objectives, this dissertation presents the following four major contributions:

- **Outcomes of O1:** A systematic literature review on software platforms for smart cities focusing on self-adaptation and BDA that provides a comprehensive view on the existing software platform for smart cities enabling self-adaptation and BDA features. Such a study enabled us to make an assessment of the existing software platforms providing smart city services by revealing features, applicability, and limitations of current software architectural solutions, but also challenges ahead for enabling self-adaptability and machine intelligence.

- **Outcomes of O2:** An approach for the descriptive analysis and development of lighting KPIs as part of the PELL SCP that enables the municipalities to analyze, measure, and monitor electric energy consumption of street lighting plants. The main goal of these KPIs is to provide a feedback about the actual behavior of the lightings plant in respect of the expected behavior according to the declared information given in the static data.

- **Outcomes of O3:** An unsupervised learning-based approach for the detection of anomalies from street lighting energy data. This approach enables to detect the anomalies from time series energy data efficiently and effectively. It is also capable of detecting anomalies without a prior training on historical energy data.

- **Outcomes of O4:** A forecasting approach for street lighting energy consumption based on a rule-based predictor and Machine Learning (ML) forecasting

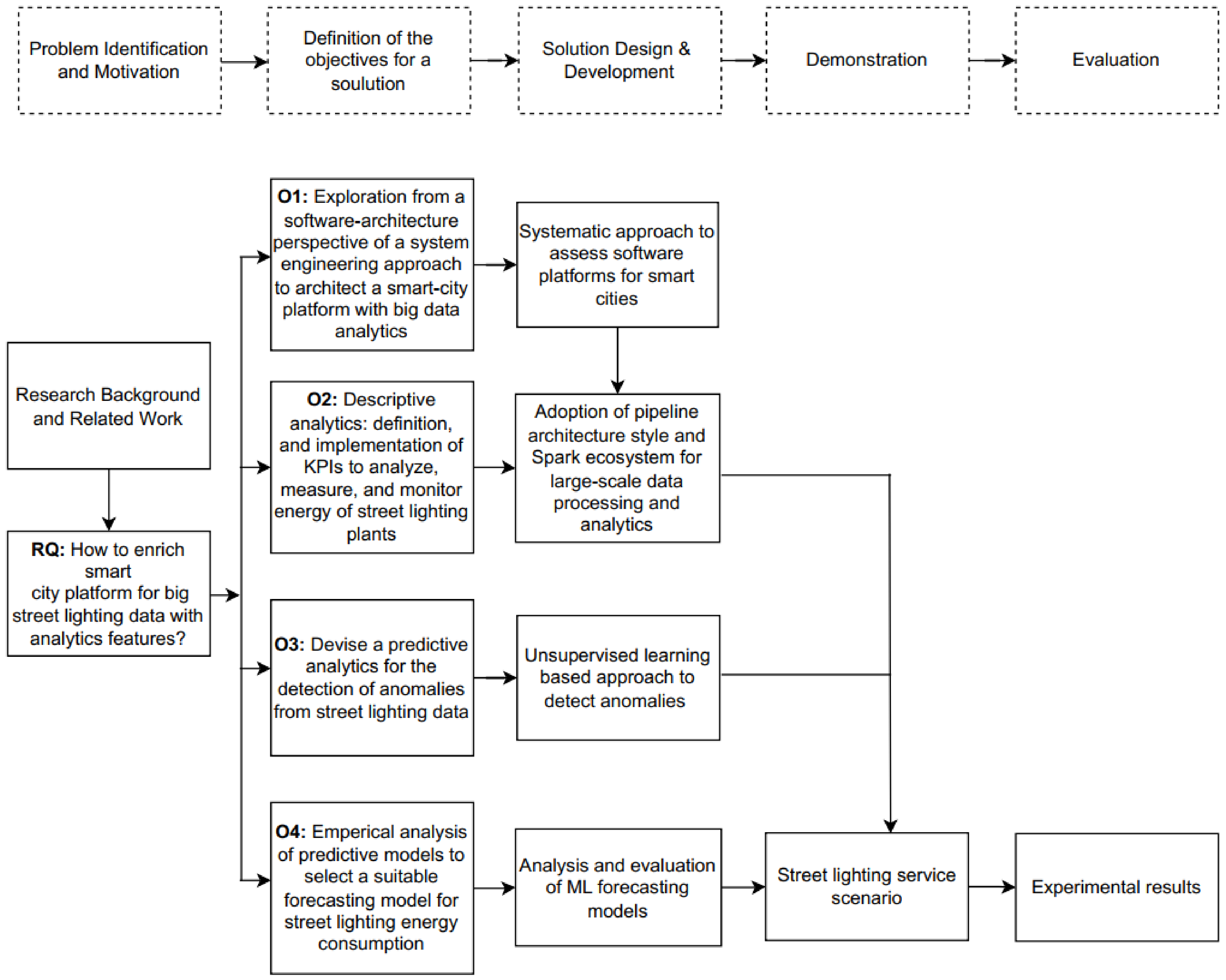

Problem Identification and Motivation
Definition of the objectives for a soulution
Solution Design & Development
Demonstration
Evaluation
Research Background and Related Work
RQ: How to enrich smart city platform for big street lighting data with analytics features?
O1: Exploration from a software-architecture perspective of a system engineering approach to architect a smart-city platform with big data analytics
O2: Descriptive analytics: definition, and implementation of KPIs to analyze, measure, and monitor energy of street lighting plants
O3: Devise a predictive analytics for the detection of anomalies from street lighting data
O4: Emperical analysis of predictive models to select a suitable forecasting model for street lighting energy consumption
Systematic approach to assess software platforms for smart cities
Adoption of pipeline architecture style and Spark ecosystem for large-scale data processing and analytics
Unsupervised learning based approach to detect anomalies
Analysis and evaluation of ML forecasting models
Street lighting service scenario
Experimental results

Figure 1.1: Dissertation scientific approach - design science research methodology

models. The rule-based predictor is a generic predictor and can be applied on any street lighting energy data set. The proposed forecasting approach is capable of determining the most accurate model for street lighting energy consumption.

1.4 Thesis Overview

The structure of this dissertation is organized as follows:

- Chapter 2: *Background Concepts about BD.* This chapter provides the background concepts about BD. It includes dimensions of BD, enabling technologies for BD, and finally the applications of BD in different fields.
- Chapter 3: *PELL Project and PELL Smart City Platform.* This chapter presents the background and overview of the PELL project. The goals of PELL project, structure of the PELL project and data management in PELL project is also discussed. It also discuss the PELL SCP with a brief discussion on the components of the PELL SCP.
- Chapter 4: *Self-Adaptation and BDA in Software Architectures for Smart Cities.* This chapter deals with the first objective of this dissertation by discussing on smart-city software platforms. In particular, it presents a systematic literature review that investigates the research efforts in the development of smart-city platforms with BDA.
- Chapter 5: *From BD to Smart Data-centric Software Architectures for City Analytics: the case of the PELL Smart City Platform.* This chapter deals with the second objective of this thesis by providing a descriptive analysis, definition and implementation of lighting KPIs as the part of PELL SCP.
- Chapter 6: *Anomaly Detection in Public Street Lighting Data using Unsupervised Clustering.* It deals with the third objective of this dissertation. In particular, this chapter introduces an unsupervised learning-based approach to anomaly detection from time series of energy consumption data.
- Chapter 7: *Energy Consumption Forecasting in PELL SCP.* It tackles the fourth and final objective of this thesis by introducing a predictive approach to street lighting energy consumption. A novel domain specific rule-based predictor is presented to forecast street lighting energy consumption.

- Chapter 8: *Conclusions and Future Research Directions:* This chapter summarizes the overall contribution of this dissertation by discussing the key findings of the overall work. It then highlights some cutting edge research directions for future work.

Chapter 2

Background Concepts about Big Data and Related Works

In this chapter, we discuss the fundamental concepts of BD and a detailed discussion about BD technologies. We start by discussing the dimensions of BD such as volume, velocity, variety, value, and veracity. The most important enabling BD technology stack is described in detail, it includes a detailed discussion on Apache Hadoop components and the Apache Spark ecosystem. After that, we present the BD applications in different sectors. Finally, we discuss a number of approaches that are related to our work. We explore state of the art on BDA for SCP and work about smart city services reported in the public lighting domain.

2.1 What is Big Data?

BD is a technical term refers to large growing datasets that includes heterogeneous formats such as structured, unstructured and semi-structured data. BD has a very complex nature that requires powerful technologies and advanced algorithms to process such massive sets of data efficiently. The conventional techniques used to process, analyse, retrieve, store and visualise BD are now unsuitable and inadequate. Advanced analytics are undoubtedly required to address the challenges of managing and analysing a wide variety of BD islands [21], which are expanding exponentially as a result of massive amount of data being generated by sensors, transaction records, meta data, and social media to name just a few of the many data sources.

In literature, various definitions of BD are available. But the concept of BD dates back to the year 2001, where the challenges of increasing wide variety of data were addressed with a 3Vs model [22]. These 3Vs, also knows as the dimension of BD and represent the Volume, Variety, and Velocity of data [23]. The proposed model was not originally used to define BD but later has been used eventually by numerous renowned enterprises such as Microsoft and IBM [24].

In 2010, Apache Hadoop discribed BD as:

> *"Datasets, which could not be captured, stored, managed, and analyzed by general purpose computers within an acceptable scope"* [25].

In 2011, McKinsey Global Institute discussed BD as:

> *"Datasets whose size is large enough which is beyond the ability of typical database software tools to capture, store, manage, and analyze"* [26].

One of the most comprehensive and renowned definition of BD is provided by the European Commission as:

> *"Large amounts of different types of data produced from various types of sources, such as people, machines or sensors. This data includes climate information, satellite imagery, digital pictures and videos, transition records or Global Positioning System (GPS) signals. BD may involve personal data: that is, any information relating to an individual, and can be anything from a name, a photo, an email address, bank details, posts on social networking websites, medical information, or a computer Internet Protocol (IP) address"* [27].

2.1.1 Dimensions of Big Data

Initially BD was characterized into three dimensions: *Volume*, *Velocity* and *Variety* suggested by Lanny [22]. 3Vs model have been used as a common framework to describe BD [28, 29]. These BD dimensions have been extended over the years, such as 4Vs (volume, velocity, variety, and value) [30], 5Vs (volume, velocity, variety, value, and veracity) [31, 32], 7Vs (volume, velocity, variety, veracity, validity, volatility, and value), [33] and 9Vs (veracity, variety, velocity, volume, validity, variability, volatility, visualisation, and value) [34]. In this section, brief discussion is provided on 5Vs of BD.

1. *Volume:* It refers to the magnitude of digital data that is being generated continuously from millions of applications and devices (sensors, smart phones,

ICTs, logs, social networks, etc). IoT devices and social networking sites generate large *volume* of unstructured data such as audio, videos, images etc. According to [35], 2.5 exabytes data were generated every day in 2012, and this amount is growing up to two times in every 40 months approximately. In 2013, according to International Data Corporation (a company which publishes research reports), the total *volume* of digital data created and consumed was estimated as 4.4 zettabyte (ZB). This huge *volume* of data is doubling every 2 years, and by 2015 the amount of digital data grew to 8 ZB [36]. Although this abundance of digital data has created serious challenges on storage capacity, but these challenges are less severe due to advancement in storage technologies as well as decrease in the cost of storage hardware [37]. However, to perform analysis on such massive data is the actual challenge.

2. *Velocity:* It refers to the speed at which data are being generated and processed. The *velocity* of data increases day by day. Initially, companies analyzed data by following traditional data analytics approach such as batch processing due to the slow and expensive nature of data processing. As the speed of data generation is increased, BD should be analyzed in real or near real-time to make informed decisions. The *velocity* of data requires advanced solutions to process and analyze stream of heterogeneous data and infer useful information [38]. For example Facebook, Twitter, Wallmart (an international discount retail chain), YouTube, etc. are good examples that illustrates the *velocity* of BD.

3. *Variety:* Variety refers to the collection of heterogeneous types of data from diverse sources. This include data in disparate formats such as structured (data available in the tables of relational databases etc.), semi-structured (email, Extensible Markup Language (XML), JavaScript Object Notation (JSON), and other markup languages, etc.), and unstructured (pictures, text, audio, video, IoT devices data, etc). Due to the development of new analytical techniques, unstructured data are generated at much faster rate than structured data and the data type becomes less of an impediment for the analysis [39].

4. *Value:* Oracle introduced *value* as an additional dimension of BD [39], and it represents the outcome produced by BD analysis (i.e. new information) [40]. The primary challenge of BD analysis is the mining of massive sets of data in the quest for hope to add *value*. Data may have low *value* in its original

shape, but the applications of data analytics will transform the data into a high-value strategic asset. IT experts need to evaluate the benefits and costs of collecting and/or generating BD, select high-value data sources, and develop analytical solutions which are capable to provide value-added information to policy makers or managers.

5. *Veracity:* IBM added *veracity* as another dimension of BD [39], which refers to the accuracy and correctness of data. It also refers to objectivity vs subjectivity, credibility vs implausibility, and truthfulness vs deception [41]. Demchenko et al. [40] proposed different factors which can help to ensure the *veracity* of BD, these factors included but not limited to: (i) reliability and security of data store; (ii) trustworthiness of data origin; and (iii) data availability. Different statistical tools and techniques have been developed to cope with the challenges of *uncertainty* and *unreliability* of BD. Because data with poor quality can have severe consequences on the outcomes of the analytical process. Therefore, it is very important for IT professionals to understand how to extract valuable and veracious data.

2.2 Enabling Big Data Technologies

2.2.1 What is Hadoop?

Apache Hadoop is a well known BD technology that is used to efficiently store and process BD sets. Hadoop was designed to cope with the challenges such as low latency and complexity when processing and analyzing BD by using traditional technologies. Instead of using a single machine to store and process large data, Hadoop allows parallel clusters and distributed file system to analyze big data sets more qickly. Unlike traditional data processing technologies, Hadoop do not copy in memory the whole data set to perform its computations, instead Hadoop run tasks where data are actually stored. Another major advantage of Hadoop is it's ability to run programs while ensuring fault-tolerance, which is very common to encounter in distributed environment. Hadoop consists of four main modules such as: HDFS, YARN, MapReduce, and Hadoop Common. In fact, the power of Hadoop is based on HDFS and MapReduce framework.

2.2.2 Hadoop Components

2.2.2.1 Hadoop Distributed File System (HDFS)

HDFS is a distributed data storage system for a Hadoop cluster [42]. It has the capacity to supports up to hundreds of nodes in a cluster and provides a cost-effective and reliable storage capability. Tome White [42] define HDFS as:

> *"HDFS is a filesystem designed for storing very large files with streaming data access patterns, running on clusters of commodity hardware"* [42].

According to the Tome definition HDFS is designed to address *"very large files"* up to petabytes in size. The term *"streaming data access"* implies that HDFS is an ideal choice for the write-once and read-many-times processing patterns. Moreover, *"commodity hardware"* means that very expensive and highly reliable hardware is not necessarily required, instead, commonly available hardware can be used to build Hadoop ecosystem. In HDFS there is no Single Point Of Failure (SPOF) issue because one of the design objectives of HDFS is to cope with the problem of a SPOF. So in this case, when a workstation is failed in a Hadoop cluster, then there will be no noticeable interruption to users. The main advantage of HDFS is its portability across heterogeneous hardware and software platforms. In addition, HDFS also helps to minimize network congestion and improve system performance by moving computations where the data is stored.

HDFS is based on a master-slave architecture, it distribute the massive sets of data across the cluster [43]. The cluster has a unique master (NameNode), which is responsible to manage file system operations, and many slave (DataNodes), which manage and coordinate data storage on individual compute nodes. It is important to note that HDFS is not an ideal choice when an application demands *low-latency data access* (e.g. in tens of milliseconds range). In such scenarios HBase and Kudu are an appropriate tools. If the data set contains lots of small files (hundreds of millions small files), then it will not be a good choice to make use of HDFS to store data. This is because of the HDFS storage structure, HDFS stores files to a block which is usually set to 128MB (by default) or 256MB, and this consequently will cause waste of storage. In order to stores small files, HBase is a perfect choice and provide fast ad hock access by using row key. Following are some basic concepts of Hadoop HDFS.

- **Blocks:** Each file system has its own blocks, which are the smallest quantity of data that may be read or write. The hard driver has its own block, which is typically 512 bytes in size. The default HDFS block size is 128MB or 256MB, which is configurable. If a file's actual size is less than the default block size, it will save only the actual size on disk, rather than the entire block. By using block, an extremely large file that could not stored on a single disk can now be stored on a cluster with many disks. Furthermore, abstraction of a block rather than a file simplifies data storage because blocks have a fixed size and easy to replicate between data nodes.

- **Data replication:** HDFS is designed to store very large files across a number of machines in a large cluster where each file is stored as a sequence of blocks. All blocks in the file have the same size except the last block. The blocks are replicated on different machines in the cluster for fault tolerance. The block size and replication factors are configurable when a file is created and can be changed later.

- **NameNodes and DataNodes:** HDFS is based on a master-slave architecture, it distribute very large data sets across the cluster. An HDFS cluster consists of a single master known as NameNode, which is responsible to manage file system operations, and many slaves called DataNodes, which manage and coordinate data storage on individual compute nodes. Users connect with the HDFS via NameNode by using Linux-like Portable Operating System Interface (POSIX) which hides the file storage details. The DataNodes are workhouses of HDFS, which are responsible to store and retrieve blocks of files. The following Figure 2.1 illustrates the HDFS architecture with one NameNode and two Racks in the cluster.

2.2.2.2 Hadoop MapReduce programming model

MapReduce is a software framework and programming model for distributed processing of massive volume of data in a reliable and fault-tolerant manner. It is one of the important steps for BD management and analytical tools. MapReduce simplifies the processing of huge amount of data through its efficient and cost-effective structure. It facilitate the programmers to write a programs that can support parallel

[1]https://hadoop.apache.org/docs/r1.2.1/hdfs_design.html

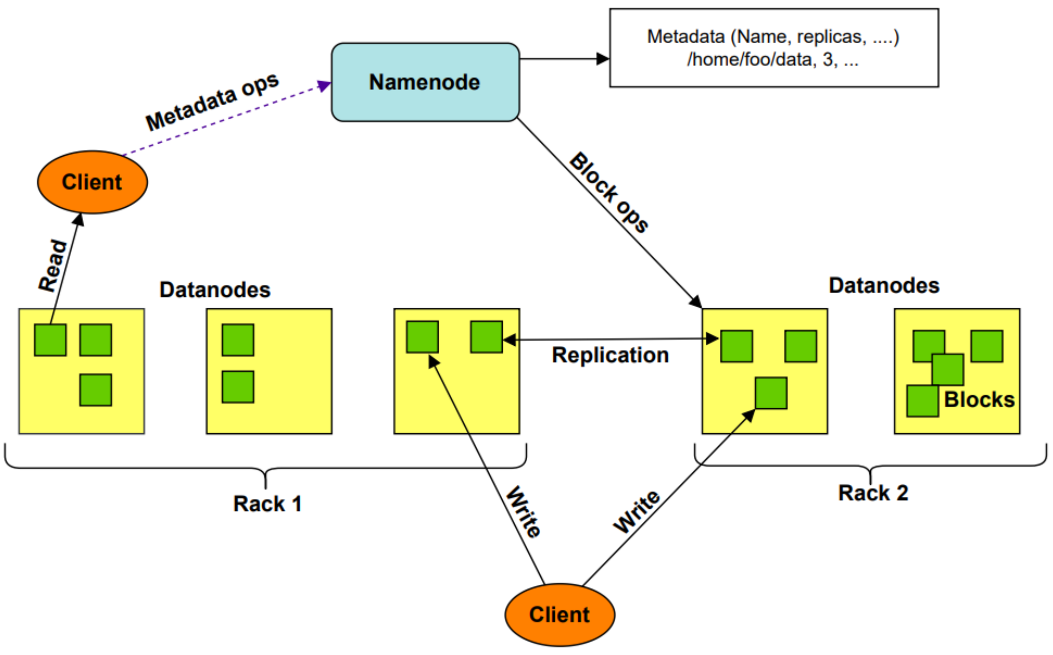


Figure 2.1: HDFS architecture [1]

processing. In MapReduce programming model data processing is based on two important functions: the Map function and the Reduce function. Map function deals with splitting and mapping of the data while Reduce function shuffle and reduce the data.

More precisely, working of a MapReduce program depends on the following series of operations:

1. First, the Map function split the input data into smaller homogeneous pieces (chunks) that constitute key-value pairs.

2. Then, the MapReduce framework forward all the key-value pairs into the Mapper that is responsible to processes each of of the pair individually, throughout several parallel map operations across the cluster. The Mapper outputs one or more intermediate key-value pairs. At this stage, the framework is charged to collect all the intermediate key-value pairs, to sort and group them by key. So the result of Map function is many keys with a list of all the associated values.

3. After this, the applications of Reduce function are used to process the intermediate output data produced by Map function. For each unique key, the Reduce function aggregates the values associated to the key according to the

predefined objectives (i.e., sorting, summarizing, filtering, etc.) of the program. After that, Reduce function produces one or more output key-value pairs.

4. Finally, the MapReduce framework store all the output in a separate output file.

The execution process of the MapReduce framework (e.g., execution of Map and Reduce tasks) is controlled by two entities such as: JobTracker and multiple TaskTrackers. The JobTracker acts like a master and it resides on the NameNode. Whereas, the TaskTrackers work like slaves which reside on the DataNode. NameNode runs a JobTracker to schedule the different jobs and distribute tasks over the slave nodes. It is the responsibility of JobTracker to monitor the status of the slave nodes and re-assign the task if any of slave node is failed. The individual task is then execute and monitor by a TaskTracker as specified by the JobTracker. Each TaskTracker can use multiple Java Virtual Machines (JVMs) to run multiple maps or reduce tasks in parallel. TaskTracker is responsible to monitor the execution of tasks and send the progress report back to the JobTracker.

2.2.2.3 Apache Hadoop YARN

YARN is a more generic version of MapReduce. In comparison to MapReduce, it offers higher scalability, parallelism, and improved resource management. The architecture design of the Apache Hadoop has been changed to incorporate YARN Resource Manager. YARN is a resource management system, which runs on the top of HDFS, and this allows numerous programs to run in parallel. It can handle batch processing as well as real-time interactive processing. MapReduce APIs is compatible with YARN. In fact, users need to do is recompile MapReduce jobs to run them on YARN. Figure 2.2 demonstrates the structure of YARN, and it also shows how jobs are executed on YARN.

Unlike MapReduce, YARN improves efficiency by dividing the two major functionalities of the JobTracker into two independent daemons (processes or programs running in background): (i) ResourceManager (RM), and (ii) ApplicationMaster (AM). The following are the major components of YARN:

- *Resource Manager (RM)*: It is responsible to allocate and manage resources across the Hadoop cluster. It runs on the master daemon. RM is based

[2]https://hadoop.apache.org/docs/current/hadoop-yarn/hadoop-yarn-site/YARN.html

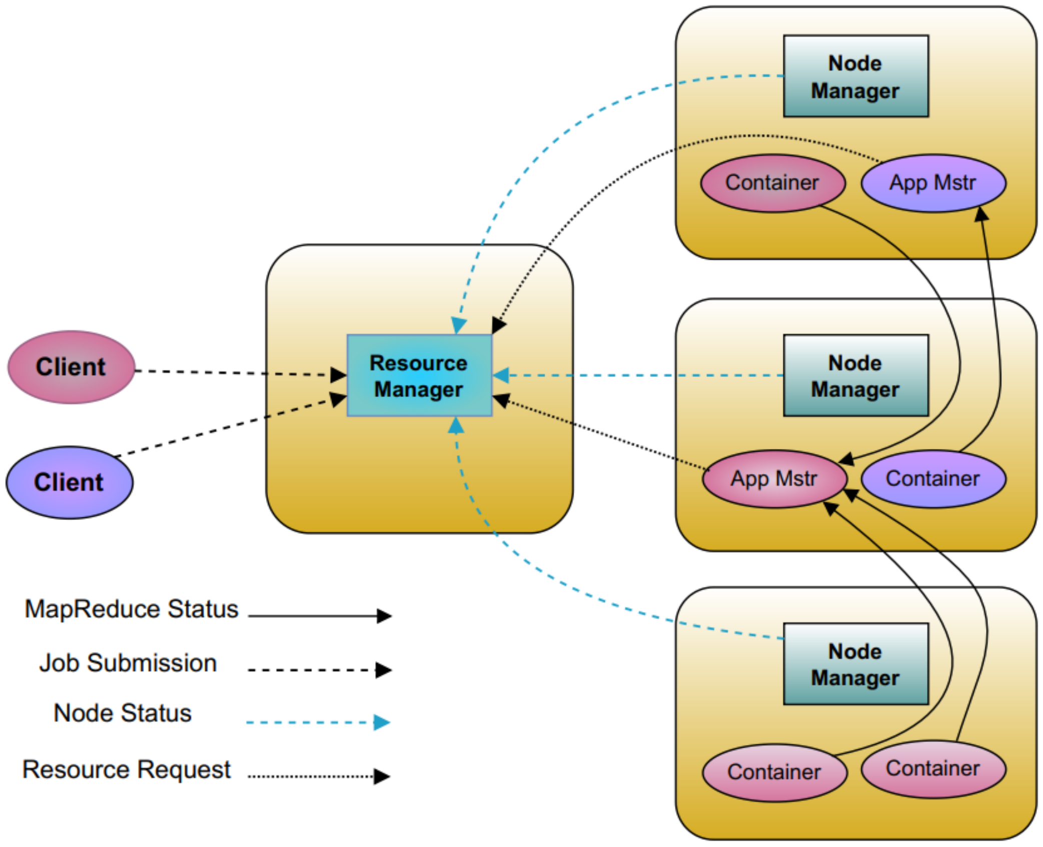


Figure 2.2: Structure of YARN [2]

on two components: Scheduler and Application Manager. The scheduler is responsible for allocating resources to various running applications; whereas Application Manager is responsible to accept job-submission, negotiate the initial container for executing an AM, and restarting AM container on failure.

- *Node Manager (NM)*: It operates on each work nodes. The major responsibilities of NM are: execution and management of tasks on the nodes, monitoring each container resource utilization, transmitting heartbeats with health status to RM, and killing of the containers as directed by RM.

- *Application Master (AM)*: For each application AM oversees the lifecycle of applications by requesting resources from RM Scheduler. Collaboration with NM keeps track of work completion and send heartbeats to RM to report on health.

- *Container*: On a single node, a resource package contains Random Access

Memory (RAM), Central Processing Unit (CPU) cores, network, hard drive, etc.

2.2.3 What is Apache Spark?

Apache Spark is an open source distributed processing framework for **DB!** (**DB!**) and machine learning that was originally developed at UC Berkeley AMPLab in 2009 [44]. Spark is similar to Hadoop, however it exploits in-memory computing to improve performance. It has an ability to run the program 100x faster than Hadoop and Hive for large scale data processing by exploiting in-memory computing and other optimizations. Spark also provides APIs for rapid application development in a variety of languages, including Java, Python, and Scala [45]. Spark is compatible with all Hadoop-supported file storage systems.

Spark data model is based on Resilient Distributed Dataset (RDD) framework, which distributes the data frame into smaller chunks on different machines for faster computation [46]. Spark supports also Map and Reduce functions for data processing. Apache Spark has some notable features such as: speed, ease of use, generality, run everywhere to name just a few.

- *Speed*: Spark can run programs up to 100x faster than Hive and Hadoop by exploiting in memory computing and other optimizations.
- *Ease of Use*: Spark provides easy-to-use APIs to process massive sets of data. This facilitate with a collection of more than 100 high level operations for data transformation as well as standard data frame APIs for managing semi-structured data.
- *Generality*: Apache Spark project consists of many components such as: Spark SQL, Machine Learning Library (MLlib) for ML algorithms, Spark Streaming, and GraphX for variety of applications. These standard libraries help developers to work more efficiently and can be used to create complex processes.
- *Runs Everywhere*: Spark can be run as a stand-alone platform. It can also run on variety of platforms such as: YARN, Hadoop, Apache Mesos[3], Kubernetes[4], etc., with the ability to access data stored in ample of data sources including HDFS, Alluxio[5], Apache Cassandra, Apache HBase, Apache Hive etc.

[3]https://mesos.apache.org/
[4]https://kubernetes.io/
[5]https://www.alluxio.io/

2.2.4 Apache Spark Ecosystem

In this section, we have discussed the components of Apache Spark ecosystem. Figure 2.3 illustrates the main components of Spark ecosystem.

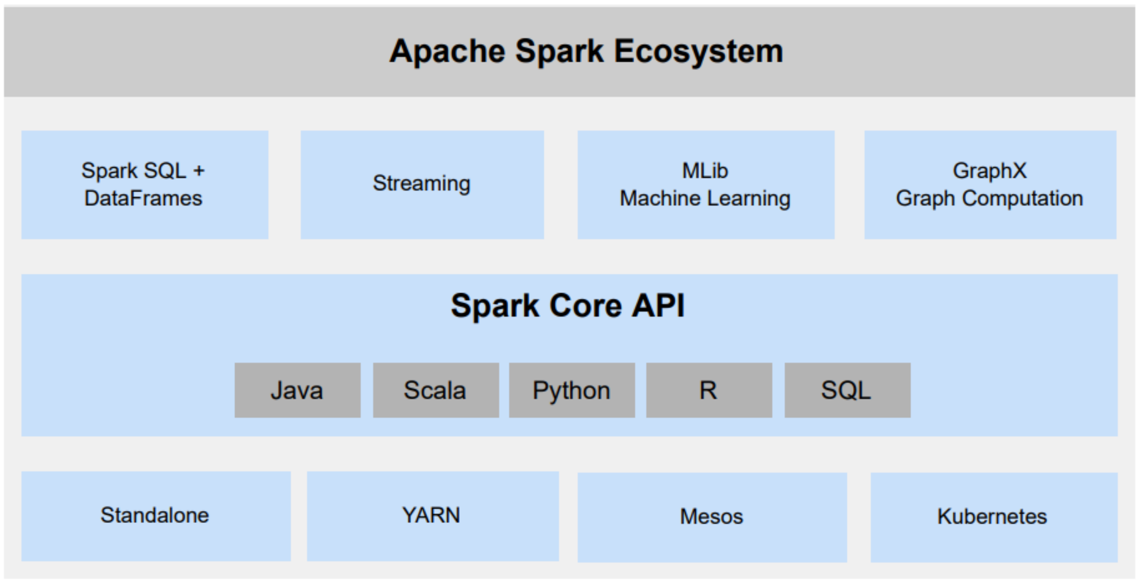


Figure 2.3: Apache Spark ecosystem

2.2.4.1 Spark Core

Spark core is the most essential component of the Spark ecosystem. It contains the basic functionality of Spark such as task scheduling, memory management, fault recovery, interacting with storage system and other functions [47]. It also contains APIs that support the RDD, which are Spark's main programming abstraction. RDD is a collection of elements that can be manipulated in parallel. Spark core also provides functional foundation for the Spark libraries such as Spark SQL, MLlib, Spark Streaming, and GraphX graph data processing. The Spark Core and cluster manager distribute the massive sets of data across the cluster and abstract it, this helps to handle BD efficiently and user-friendly.

2.2.4.2 Spark SQL

Spark SQL[6] is a Spark module that allows querying data via SQL as well as the Apache Hive variant of SQL known as Hive Query Language (HQL). One of the most important feature of Spark SQL is that it integrates relational structured data

[6]https://spark.apache.org/sql/

processing (relational tables) with Spark's functional programming APIs RDDs [48]. So application developers can easily intermix SQL queries with RDDs to query external data sets with complex analytics. Users can query data stored in existing RDDs as well as data imported from external sources (such as Parquet files and Hive Tables). Moreover, Spark SQL allows writing RDDs out to Hive tables or Parquet files. For this reason, it enables fast parallel processing of data queries over huge distributed data sets. In order to support fast application development, Spark has developed the Catalyst framework, which allows users to rapidly implement new optimization by using Spark SQL.

2.2.4.3 Spark Streaming

The primary objective of Spark streaming[7] is to provide a set of APIs that can easily handle and process live data streams [49]. Example of live streams of data include log files generated by production web servers, web click information, and sensor data collected by IoT devices. Spark streaming APIs allows developers to write batch-like processes in Java, Scala and Python to stream tasks. It not only allows batch processing, but also interactive queries on stream state. It has an ability to read data from HDFS, Flume, Kafka, Twitter, and many other variety of data sources.

2.2.4.4 MLlib

MLlib[8] is an open-source, scalable, and distributed machine learning library built on top of Spark [50]. MLlib offers various machine learning algorithms such as classification, clustering, regression, and collaborative filtering. Like Mahout, MLlib is a very powerful library for machine learning algorithms. MLlib also supports regression models, however, Mahout does not support such model. In comparison to Mahout, MLlib is a newer project.

2.2.4.5 GraphX

GraphX[9] is Apache Spark API that enables the Spark to manipulate graphs (for example, social network relationship graph) and execute graph-parallel computations. By taking the advantages of Spark parallel computation, GraphX provides perfor-

[7]https://spark.apache.org/streaming/
[8]https://spark.apache.org/mllib/
[9]https://spark.apache.org/graphx/

mance equivalent to the special graph processing system such as Apache Giraph[10] and GraphLab[11], while retaining the fault tolerance and highly flexibility APIs of Spark. Like Spark Streaming and Spark SQL, GraphX extends the capabilities of Spark RDD API by introducing Resilient Distributed Property Graph; thus multiple graph can be attached to each vertex and edge attribute [51]. In order to support graph computing, GraphX offers different operators such as: subgraph, mapVertices, MapReduceTriplets. As a graph processing framework of Spark, GraphX provides the most common graph processing algorithms such as: PageRank, Connected components, Label propagation, SVD++, and Triangle count.

2.3 Big Data Applications

The applications of BDA exist in every sector. In this section, we have discussed some examples of BD applications.

2.3.1 Healthcare

The medical data is being generated from heterogeneous sources such as: laboratories, pharmacies, clinical data, physician's written prescription, clinical decision support system, and patient symptoms data collected by using different IoT sensors [46]. The analysis of this massive sets of medical data has many interesting applications. It enables the personalization of health services and the adaptation of public health policies based on population symptoms, disease evolution and a variety of other factors. It is also useful to improve hospital operations and lower health-care costs. Traditional medical systems use different monitoring devices to monitor the vitals of a patient and trigger an alarm when a vital cross discrete numerical threshold value [52]. Such alarming mechanisms generate a considerable number of false alarms, as they depend on the single source of information which usually lack context of a patient's true condition from a broader point of view. This is one of the situations in which BD can help. By analyzing historical medical record of a patient collected from heterogeneous sources make it possible to take informed decisions from a broader point of view rather than isolated one [52].

[10]https://giraph.apache.org/
[11]https://en.wikipedia.org/wiki/GraphLab

2.3.2 Transportation

BD has the ability to enhance the safety and sustainability of the transportation systems. Many cities have installed monitoring devices such as: roadside sensors, cameras, and many other IoT devices to observe traffic condition and promote traffic safety. Moreover, these devices collect also a huge volume of traffic data which is very useful for the transportation managers to gain better understanding of traffic flow in their respective areas. This massive sets of heterogeneous traffic data contains many hidden patterns that can help the transportation department to optimize their services. For example, the data collected by roadside sensors can be analyzed to detect traffic congestion. As soon as traffic congestion is detected, travel alerts can be sent to the drivers to help them to find an alternative route and so reduce congestion [53]. Analysis of vehicle wait times at traffic lights can yield useful information and lead to improved approaches to manage traffic flow and optimize traffic light policy [5]. Furthermore, the analysis of video data can help to detect and recognize the objects (for example, vehicles or pedestrians), detect their trajectories, and recognize traffic events such as: veering, abrupt braking, and near misses [54]. These analysis can assist the decision-makers to take essential actions to improve the road safety, avoid collisions, and save lives. In addition, mining huge volume of traffic data can helps also to improve travelling business by predicting demand about public or private networks [55].

2.3.3 Smart Grid

The smart grid has become an important component of a smart city. It uses **ICTs!** (**ICTs!**) to establish a two-way communication between power producers and consumers to enhance grid productivity and reliability through system self-monitoring and feedback. This entails installing smart sensors and meters on production, transmission, and distribution systems, as well as consumer access points, in order to obtain detailed near-real-time data on current power production, consumption, and faults [5]. BDA helps in the identification of at-risk transformers as well as the detection of abnormal behaviors of the linked devices. Grid Utilities can then select the most appropriate treatment or response. The real-time analysis of the massive sets of data enable to model incident scenarios. This allows to establish strategic preventive plans to reduce corrective expenditures [55]. In addition, there are variety of potential applications of BDA on smart grid data such as: intelligent energy planning and pricing analysis, real-time processing of the electrical consumers energy consumption, load forecasting, automatic billing, outages detection, load management with demand response, energy forecasting under under high unpredictability, and asset management [56].

2.3.4 Law Enforcement

Law enforcement agencies are using BD in a range of daily operations and surveillance activities such as patrolling, investigation, and crime analysis. The use of BD by the Police department is subject to of contentious debate in policy, media, legal, regulatory, and academic circles. However, much of the discussion on the subject is theoretical, concentrating on the benefits and drawbacks of new forms of data-based surveillance [57]. Law enforcement agencies try to predict the next crime location by making use of historical data such as: type of crime, place and time of the crime, social networks data, and smartphone tracking. A group of researchers at Rutgers University developed an app known as RTM Dx to prevent crimes. This application is being used by the police department at Illinois, Texas, Arizona, New Jersey, Missouri and Colorado. This app helps police officials to measure the spatial correlation between the crime location and environmental factors [46].

2.3.5 Education

Thanks to the advent of computerized course modules which have made it possible to measure academic performance in real-time. This allows instructors to track the progress of their students after each module and provide immediate feedback on their learning patterns. It also assists teachers in evaluating their teaching pedagogy and making adjustments based on student performance and needs. Moreover, on the basis of students' performances in their previous modules, it is possible to predict the students who require special attention and students who can handle challenging assignments [46]. In an interesting study [58], authors used intelligent tutor software to study student reading comprehension and discovered that when students re-read an old story instead of a new one, their reading errors decreased significantly.

2.3.6 Political Analysis and Government Monitoring

Governments collect, manage, and analyse BD to enhance their abilities to serve their citizens as well as address major national issues that include the economy, health care, job creation, natural disasters, and terrorism. Many governments, including India and the United States, are mining data to track political trends and assess public opinion [55]. Many systems have been developed by the governments to collect the data from various sources such as: social network data, personal interviews, and voter compositions. These system enables to analyse the local issues in addition to the national issues. In addition, governments can use IoT sensors to monitor water flow in large networks. It helps the city managers to to rely on real-time monitoring system to detect leakages and illegal supply to ensure equitable supply of water to different areas of the city [55].

2.3.7 Marketing

BDA helps marketing managers to make informed decisions, in order to meet customer needs. Marketing analytics assists the business organizations in evaluating their marketing performance, analyzing consumer behavior and purchasing patterns, and analyzing marketing trends to aid in the modification of marketing strategies such as ad placement on a webpage, dynamic pricing, and offering personalized products [46].

2.3.8 Internet of Things (IoT)

Thanks to the recent advancement in low-cost sensors, wireless communication networks, pervasive computing, and Web technologies, the IoT has gained momentum and strong footing in connecting everyday objects and alleviating especially machine-to-machine communication with the physical world [59]. IoT sensors are basically utilized to collect large scale urban data, which used as input for BD applications. IoT-based BD application are increasingly maturing and rapidly proliferating due to the advance and prevalence of sensing technologies. The data collected by the IoT devices is processed and analyzed by using data mining and machine learning approaches to develop model, extract hidden patterns, make correlation, and deploy the obtained results to make decision and automate the urban operations. Nowadays, there are various IoT-based BD applications have been developed for different domains such as: health care, agriculture, smart environment, logistic, etc. For example, it is possible to track vehicles positions by using IoT sensors, wireless adapters, and GPS. As a results, data-driven applications enable companies to not only supervise and manage their staff but also to optimize delivery routes. This is accomplished by utilizing and combining a variety of data, including previous driving experience [55].

2.4 Related Work

The scientific community has been actively involved in the design and development of intelligent systems to improve the energy efficiency of public lighting. We here list the most relevant works that propose tools/software platforms based on BD technologies in the specific context of public lighting. The state-of-the-art includes two main streams of research: (i) BDA for smart city platforms, and (ii) works about smart city services reported in the public lighting domain.

2.4.1 Big data analytics in smart city platforms

The role of BD in the development of smart cities is undeniable [60]. BDA is an emerging technology that has a huge potential to enhance smart city services by transforming city information into city intelligence. Despite this, it has attracted attention in a rather restricted range of application domains, and its joint application with self-adaptation mechanisms is rarely investigated. BDA has been actively used in the development of smart city software architecture, in this section, existing state-of-the-art on smart city software architectures adopting BDA and self-adaptation are discussed.

Azzam et al. [61] proposed the architecture of the *CitySPIN* project for the development of smart services. The platform of the *CitySPIN* is assisted with methods and techniques, which are based on Semantic WEB and Linked Data technologies for the acquisition and integration of heterogeneous data of different formats (structured, unstructured, and semi-structured), including open data and social data. The *CitySPIN* project is based on a three-layered architecture: 1) back-end layer, which is responsible for data collection, pre-processing and data integration, 2) service layer, which provides the services of analysis by applying queries and prediction model. The prediction model is based on machine learning algorithms to facilitate the prediction by using historical data that can assist in decision making, and 3) front end layer or presentation layer, which facilitate the users to interact with the system and perform different kinds of analysis of their need.

In another study [62], a *CityPulse* framework is presented for the development of smart city services by enabling the integration of heterogeneous data streams, interoperability, (near-) real-time data analytics, and applications development in a scalable framework. The CityPulse framework is composed of a powerful data analytics module, which is empowered to perform intelligent data aggregation, quality assessment, event detection, contextual filtering, and decision support. All the components of the *CityPulse* have been developed as reusable entities and application development is facilitated by open APIs.

Pedro et al.[63] proposed a project called *CityAction* in the context of smart city, which facilitate the city managers to take actions/decisions on the bases of real-time city data. The main objective of the project is to support the design and development of an integrated platform that has the ability to combine city data coming from different sources with heterogeneous devices and perform intelligent data analysis. The architecture of the *CityAction* is based on four independent layers: 1) Device layer, in which IoT sensors, actuators and communication gateways correspond to different vertical systems, 2) M2M Connectivity layer, which is responsible for the devices interconnection to the internet, 3) Middleware layer, this layer has the responsibility to integrate several blocks like data broker, monetization, data management and analytics, vertical management M2M management, and API management, 4) Application layer that also has an ability to incorporate the open data to enrich the application portfolio. Mohamed et al. [64] came up with another approach to transform BD into a smart data. In this study, they introduced a system called *CityPro*. The architecture of the *CityPro* is discussed for surveillance system. In the architecture of the *CityPro*, a federated star-schema is used in the storage repository and repository only store the summarized data instead of huge amount of data.

In another study [53], an approach is discussed for the development of next-generation BD applications. they have proposed a *CAPIM* (Context-Aware Platform using Integrated Mobile services) platform, which is designed to automate the process of collecting and aggregating the context information on a large scale. An intelligent transportation system is developed by using *CAPIM* platform, which helps the user and city managers to understand the traffic problems of their city. In another interesting study, Paula et al. [65] proposed a simple and scalable *hut* architecture to extract valuable historical insights and actionable knowledge from IoT data streams. The developed *hut* architecture supports both historical as well as real-time data analysis. It is applied on two real-world applications scenarios in a smart city environment such as transportation and energy management. The implementation of the *hut* architecture is based on open-source components and can be replaced or customized according to the need. In another study [66], authors proposed a system called CrowdNav to enable self-adaptation in a complex large-scale software-intensive distributed system by using BDA. The novel contribution of the developed system is to use the operational data, which is generated at run-time for adaption and the seamless integration of self-adaptation with the latest BD technologies.

2.4.2 Smart public lighting

The proposed PELL SCP is initially validated in the public lighting domain, so hereafter, we have presented some notable studies reported in the public lighting domain. Marijana et al. [67] proposed an approach to address the issues of energy efficiency of the public buildings. The contribution of this approach is two-fold: 1) apply machine learning models to predict the energy consumption of the public building, a real dataset of Croatia that composed of 17,000 public buildings is used for experimental evaluation. Three well-known ML methods i.e. deep neural network, RPart decision tree, and random forest were used, and it is observed that random forest produces the highest accuracy, and 2) architecture of an intelligent ML-based energy management system called: MERIDA that is composed of six layers i.e. i) BD collection, ii) data pre-processing, iii) ML models for prediction, iv) data interpretation and visualisation, v) decision making, and vi) benefits. This proposed study has extended and modified the approaches presented in [68] [69] [70].

In another study [69], authors proposed an advanced IoT-based intelligent energy management system for public buildings. The architecture of the proposed system consists of three modules: 1) data collection module 2) data integration module, and 3) prediction models /rules and action plans. In the first module of the proposed system, authors introduced five pillars such as building's data, energy production, energy prices, weather data, and end-users' behavior. In data integration module, a semantic framework for data integration is proposed, which is based on Ztreamy system, a Python-based semantic service and Optimus ontology is also created. The third module integrates prediction models, rules, and a MariaDB database that is used to store the results. In [68], authors discussed an IoT-based system comprised of three-layered architecture. The identified IoT layers are: (1) the perception layer which is composed of internet-enabled devices (sensors. cameras, GPS, RFID, etc), (2) the network layer, which is responsible to forward data from the perception layer to the application layer, and (3) the application layer, which process the data coming from previous two layers and suggest better power's distribution and management strategies. Authors stress that Supervisory Control and Data Acquisition (SCADA) systems are the core of decision-making in the smart grid, and these systems are used for real-time monitoring and control over the power grid.

Galicia et al [71], proposed a ML-based ensemble method for predicting big time series data. The ensemble model is composed of a decision tree, gradient-boosted trees, and a random forest. The system is implemented by using MLlib library of the Apache Spark framework to ensure the scalability and suitability for BD. The experimental evaluation is performed on two different datasets i.e. Spanish electricity consumption data of 10 years and Australian solar data. The experimental results showed that the dynamic ensemble model provides best prediction results in comparison with the static ensemble and individual ensemble members.

In another similar study [72], a Deep Learning (DL) based approach is proposed for big-time series data forecasting. The deep feed-forward neural network is used with the Apache Spark platform for distributed computing. The system is evaluated on a real-world dataset composed of electricity consumption in Spain and the authors observed that deep learning is one of the best techniques to process big time series data along with the decision tree, in term of scalability and accuracy. In [70], an intelligent building management system is proposed to manage the public sector buildings of Croatia. The system is based on a three-layered architecture, which collects building data, their energy, water consumption, monitor consumption indicators, detects anomalies or irregularities, sets energy efficiency targets and reports energy and water consumption savings.

An IoT and BDA-based smart home energy management system is presented in [73], in which IoT devices are installed with home appliances to collect the energy consumption data, then collected data is forward to a centralized server for further processing and analysis. The proposed system has utilized off-the-shelf BUsiness Intelligence (BI) and BDA software components to manage energy consumption. In [74], authors presented an adaptive lighting system for smart city environment. The developed system has the ability to autonomously control the lighting level of a street lamp by exploiting the vehicles data (car, bus, bike motorcycle) and pedestrian traffic in the targeted area. The system is making use of locally installed controllers, motion sensors, video cameras and electronic devices for video processing. Authors reported that by using this proposed system up to 65% energy can be saved in comparison of tradition street lamp system.

From this preliminary state-of-the-art review, it seems there does not exist a big data-driven software architecture for public street lighting, which is intended to collect, represent, control, predict, and possibly optimize the behavior of public street lighting plants. This establishes the research gap that we address in this thesis. We design and implement a Smart Data-centric software architecture for the analysis of energy consumption data in the context of public street lighting. we have designed and prototyped different types of analytics services as part of the proposed smart city platform.

2.5 Summary

In this chapter, we have provided a brief introduction to BDA concepts and discussed the most important 5 Vs (volume, velocity, variety, and veracity) to explain the BD dimensions. After this, we discussed enabling BD technologies. Regarding the enabling technologies, we provided a brief introduction to Apache Hadoop and the components of Hadoop such as HDFS, Hadoop MapReduce programming model, and Apache Hadoop YARN. As part of the enabling technologies, we then discussed Apache Spark and the ecosystem of the Apache Spark. We also discussed the components of Spark ecosystems such as Spark Core, Spark SQL, Spark Streaming, MLlib, and GraphX. Then, some of the most important domains of prospective applications of BD are presented discussing several use-cases of each domain in great detail.

We then discussed works related to this thesis. The analysis of the literature highlights the gap in existing studies. In the next chapter, we provide a brief introduction to PELL project along with the objectives of the PELL system. This work is done as part of the PELL project within the scope of the collaboration agreement Regione Lombardia-ENEA.

Chapter 3

PELL Project and PELL Smart City Platform

This chapter starts with the introduction of the PELL project and throws light on the objectives of the PELL project. It discusses the structure of the PELL project along with the static and dynamic phases. It also discusses the different components of the static and dynamic phases. After this, It discusses the process of PELL data acquisition, which include static data as well as dynamic data, and data model to manage and exchange the PELL data.

3.1 PELL Background

The PELL (Public Energy Living Lab)[1] project was an initiative of Italian national research agency ENEA[2], started in 2014, and financed by Ministry of Economic Development (MISE). It involves several Italian stakeholders such as public authorities (like the S.p.A CONSIP and Acquirente Unico), utility providers, and energy service companies.

3.2 Objectives of PELL

The main objective of the project is to improve the development of a new efficient and effective management model for public lighting to help turn traditional cities

[1]`https://www.pell.enea.it/enea/`

[2]ENEA Italian National Agency for New Technologies, Energy and Sustainable Economic Development `https://www.enea.it/en`

into smart cities. The short-term goal is to collect, handle, organize and evaluate the strategic data of urban energy-intensive infrastructures (public street lighting and public buildings, e.g. schools), in order to provide a constant and dynamic assessment of their functional and energetic performances, improve their management and allow the start of urban reorganization processes. The public street lighting sector is the first one addressed by the PELL project; in fact, lighting system plants are widespread everywhere in the Italian territory, and a rationalization through a capillary mapping could serve as a baseline for core tasks such as maintenance or energy re-qualification. Moreover, public lighting is one of the highest expense items in municipal administrations, thus the potential improvement can lead to consistent economic savings.

3.3 PELL System Structure

PELL system structure is composed of two phases (1) static phase, (2) and dynamic phase. The ICT Living lab platform represents static phase to collect plant's static data, to evaluate their performance and functionalities, and data consumption acquisition represents dynamic phase to collect and monitor daily consumption measure. Figure 3.1 shows the complete PELL system structure. ICT Living lab platform represents the static phase to collect plant's static data, which is used to evaluate and diagnostic their performances and functionalities, whereas, data consumptions acquisition represents dynamic phase to collect and monitor daily energy consumptions.

3.3.1 Static Phase

The static phase of the PELL platform is responsible for collecting the plant's static data through a census sheet. As static data we refer to information concerning the lighting plant system in terms of its topology, the registry municipality, technical characteristics of the entities involved and other context info such as lighting control policies and annual expenses. To monitor and evaluate the the efficiency of operation of a plant, it is essential to know its whole infrastructure through its rigorous description.

In particular, the entities involved are:

- Point of delivery (POD). Point of delivery where electrical panels are con-

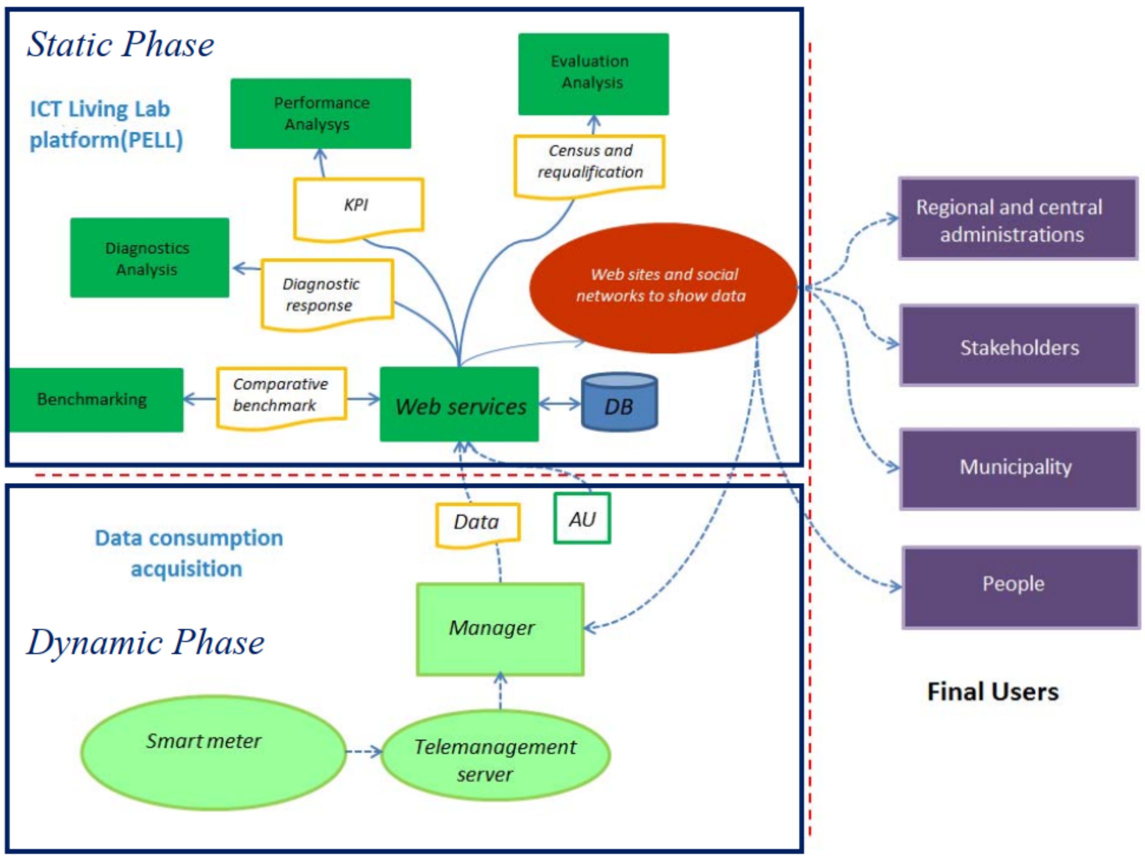


Figure 3.1: PELL system structure [75]

nected. PODs are uniquely identified at national level and they are the devices where the billable consumption is evaluated.

- Electrical panels. Devices that physically deliver electrical energy to the lighting points. Electrical energy absorbed is measured at this level through smart meters.

- Lighting points. Devices providing lighting to the streets. Its structure is usually composed by one or more lighting devices (e.g. lantern, light armor) containing lighting source/s (e.g. halogen, high pressure sodium, low pressure sodium) or LED module/s. Usually, lighting points are installed on a support (e.g. a pole)

1. *Evaluation analysis*: In the PELL system structure, evaluation analysis (Figure 3.2) provides information about the existing lighting plant and the associated KPIs. This information enables the municipal authorities to make

on-time decisions on any type of lighting system requalification. The PELL platform also has the ability to provide two services i.e. preliminary lighting simulation and economic-financial assessment (SAVE tool) to test some redevelopment hypotheses.

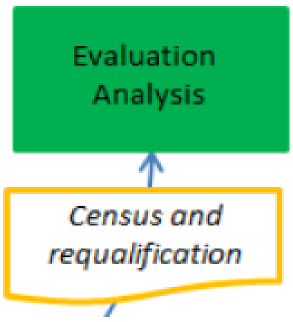


Figure 3.2: Evaluation Analysis.

The information about the technical characteristics of the entities is defined as *Census Tech Sheet*[3] which presents all the system management data and structured in XML format. The persistence system for this data is designed as a structured relational database because the scalability is not an issue since the amount of information concerning lighting plants is limited; the entities involved in Census Tech Sheet are well-defined and strongly related.

The Census Tech Sheet uses a tree structure that starts from the data relating to the system registry up to the detail of the light source. Figure 3.3 illustrates the structure of a Census Tech Sheet, which represents the complexity and completeness of the electrical and descriptive data of the plant.

[3] `https://www.pell.enea.it/assets/data/download/CensusTechSheet-ImplementationGuide.html`

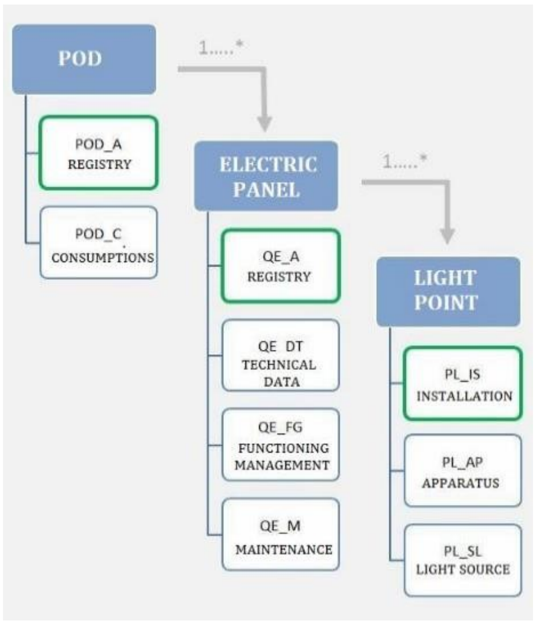


Figure 3.3: Census sheet structure [75]

A Census Tech Sheet is comprised of:

- The geographic position of the lighting plant, the geographic coordinates for the purpose of its physical delimitation.
- A list of PODs, each pod can have several electrical panels.
- Light points are associated with each electrical panel.
- Each light point has several luminaires (lantern, street armor, floodlight, street furniture, etc).
- A plant is configured on one or more homogeneous areas, to which more light points are attached

The electric panel registry shows the number of light points; each electric panel can be connected to N lights, each with defined characteristics such as the type of installation, the height, the possible inclination, the distance of the support from the roadway, the length of the arm, as well as the characteristics of the supports (material, age, status)[75]. All these features allow the complete definition of the characteristics of an entire plant, these features also enables us to evaluate both qualitative and quantitative characteristics of the plant, with the goal of achieving complete knowledge of the lighting system. Figure 3.4 describe the tree structure of the Census Tech Sheet.

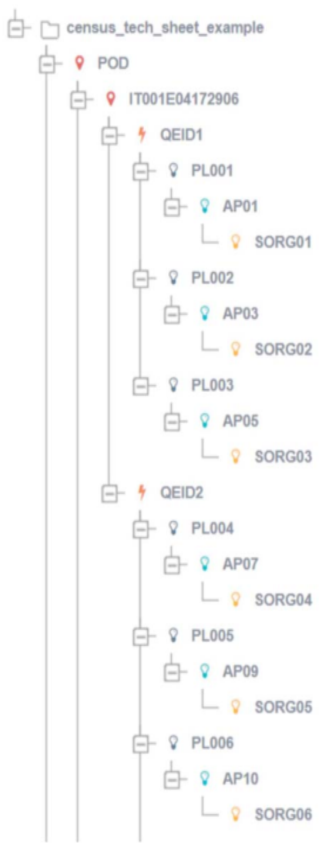


Figure 3.4: Census sheet tree structure [75]

2. *Performance analysis*: The performance analysis (Figure 3.5) provides different KPIs developed in the PELL project and refers to the static phase of the PELL system. These KPIs are Geometric KPIs, Technological KPIs, Dimming KPIs, BAU (Business as Usual) KPIs, and BAT (Best Available Technology) KPIs. Geometric KPIs describe the lighting project's quality in terms of guaranteed power per square meter. Technological KPIs are referred to as installed lamps energy efficacy. Dimming KPIs provides the efficacy of dimming strategies to manage and control the lighting output. BAU KPIs evaluates the energy advantages of utilizing installed lamps compared to Best as Usual (BAU) technology, and BAT KPIs measures the energy advantages of utilizing installed lamps compared to the best available technology (BAT) in the market.

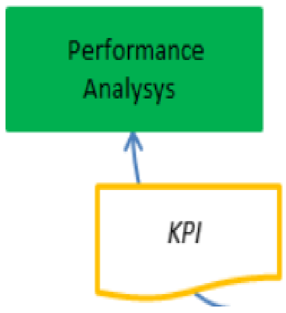


Figure 3.5: Performance Analysis.

3.3.2 Dynamic Phase

The dynamic phase of the PELL platform aims at collecting all the electrical measurements provided by utilities and acquired from the installed electrical panels and delivery points. In order to properly collect and process such data, a scalable BD platform based on Hadoop has been set up. The Hadoop platform is based on one name node, four data nodes, seized according to the amount of data want to collect.

As dynamic data, we refer to electrical measures collected by smart meters installed into the electrical panels. This data is retrieved through a MQTT broker and represented in the JSON format according to a specific data structure, defined as *Urban Dataset*. Such data structure is aimed to define a format for exchanging data in an interoperable way within smart city platforms. There are two specific categories of data set in the Urban Dataset: *Counter Reading* and *Counter Reading monophase*. The former is used to describe electric measures collected in three-phase lighting plants while the latter for single-phase ones. Such data requires a great scalability and high efficiency in data retrieving, thus an unstructured data lake based on the Apache Hadoop ecosystem is used to persist them.

Figure 3.6 illustrates the process of PELL dynamic data collection and storage. The *Counter Reading*, formatted as JSON files are sent through MQTT protocol to the broker, through publish system over each topic dedicated to each specific utility for each municipality. Then, an MQTT Bridge is subscribed to each topic, and routes the messages to the local SCP (Smart City Platform) if available, and to Hadoop infrastructure.

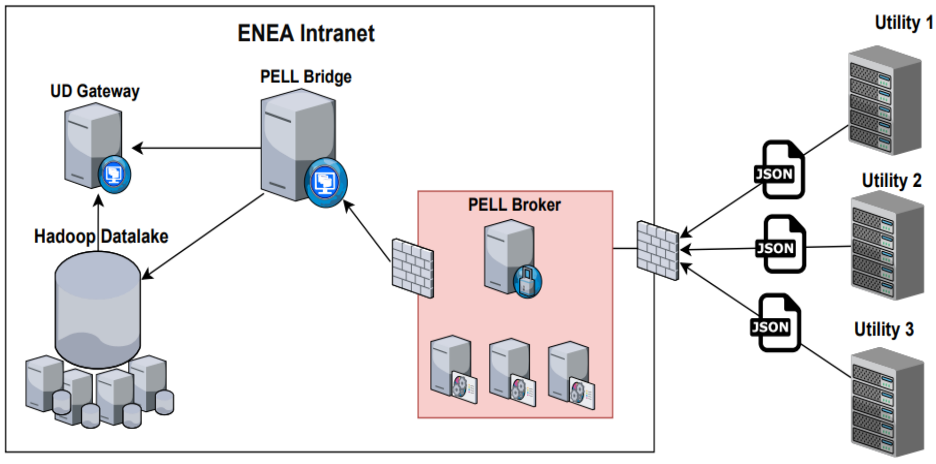


Figure 3.6: PELL dynamic data collection [75]

This electric consumption data combined with the census sheet static data will be used for the diagnostic analysis and comparative benchmark studies, as shown in Figure 3.7

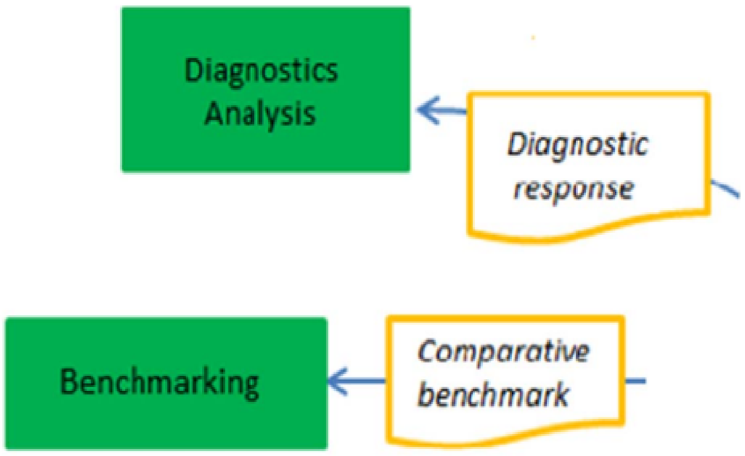
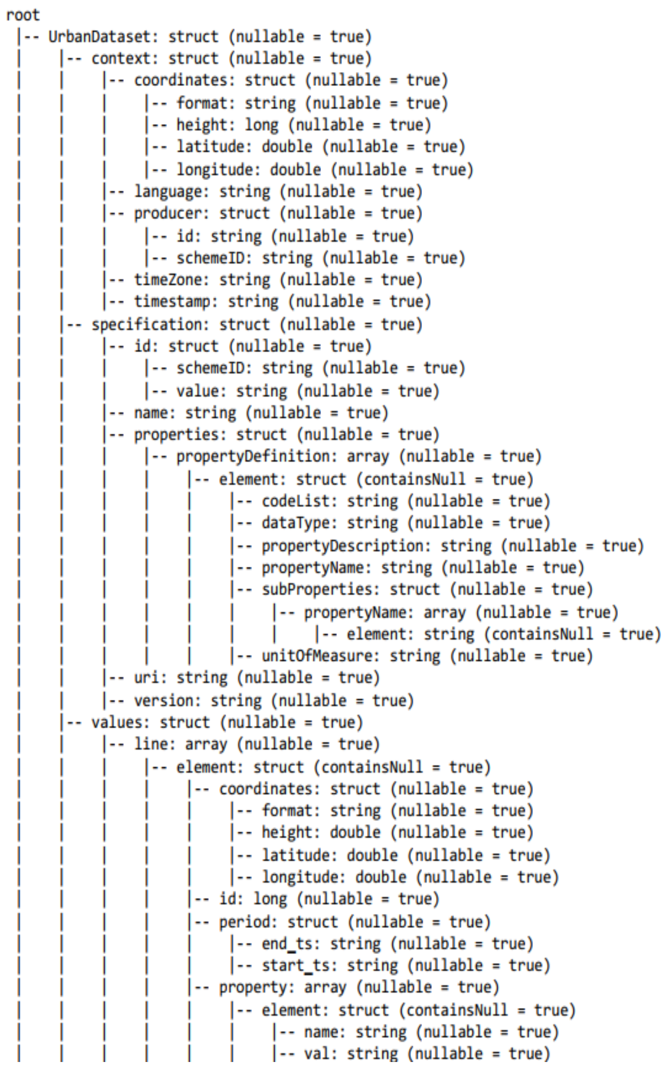


Figure 3.7: Diagnostics and benchmarking.

Each *Counter Reading* refers to the consumption data of a single day, and consumption is measured with the granularity of fifteen minutes. Therefore, each JSON file contains 96 values (4 values per hour and per 24 hours of a day). Figure 3.8 shows the structure of the counter reading as PySpark dataframe, components of the *Counter Reading* are discussed in Section 3.4.



```
root
 |-- UrbanDataset: struct (nullable = true)
 |    |-- context: struct (nullable = true)
 |    |    |-- coordinates: struct (nullable = true)
 |    |    |    |-- format: string (nullable = true)
 |    |    |    |-- height: long (nullable = true)
 |    |    |    |-- latitude: double (nullable = true)
 |    |    |    |-- longitude: double (nullable = true)
 |    |    |-- language: string (nullable = true)
 |    |    |-- producer: struct (nullable = true)
 |    |    |    |-- id: string (nullable = true)
 |    |    |    |-- schemeID: string (nullable = true)
 |    |    |-- timeZone: string (nullable = true)
 |    |    |-- timestamp: string (nullable = true)
 |    |-- specification: struct (nullable = true)
 |    |    |-- id: struct (nullable = true)
 |    |    |    |-- schemeID: string (nullable = true)
 |    |    |    |-- value: string (nullable = true)
 |    |    |-- name: string (nullable = true)
 |    |    |-- properties: struct (nullable = true)
 |    |    |    |-- propertyDefinition: array (nullable = true)
 |    |    |    |    |-- element: struct (containsNull = true)
 |    |    |    |    |    |-- codeList: string (nullable = true)
 |    |    |    |    |    |-- dataType: string (nullable = true)
 |    |    |    |    |    |-- propertyDescription: string (nullable = true)
 |    |    |    |    |    |-- propertyName: string (nullable = true)
 |    |    |    |    |    |-- subProperties: struct (nullable = true)
 |    |    |    |    |    |    |-- propertyName: array (nullable = true)
 |    |    |    |    |    |    |    |-- element: string (containsNull = true)
 |    |    |    |    |    |-- unitOfMeasure: string (nullable = true)
 |    |    |-- uri: string (nullable = true)
 |    |    |-- version: string (nullable = true)
 |    |-- values: struct (nullable = true)
 |    |    |-- line: array (nullable = true)
 |    |    |    |-- element: struct (containsNull = true)
 |    |    |    |    |-- coordinates: struct (nullable = true)
 |    |    |    |    |    |-- format: string (nullable = true)
 |    |    |    |    |    |-- height: double (nullable = true)
 |    |    |    |    |    |-- latitude: double (nullable = true)
 |    |    |    |    |    |-- longitude: double (nullable = true)
 |    |    |    |    |-- id: long (nullable = true)
 |    |    |    |    |-- period: struct (nullable = true)
 |    |    |    |    |    |-- end_ts: string (nullable = true)
 |    |    |    |    |    |-- start_ts: string (nullable = true)
 |    |    |    |    |-- property: array (nullable = true)
 |    |    |    |    |    |-- element: struct (containsNull = true)
 |    |    |    |    |    |    |-- name: string (nullable = true)
 |    |    |    |    |    |    |-- val: string (nullable = true)
```

Figure 3.8: Structure of the counter reading as dataframe

3.4 Dynamic Data Model: UrbanDataset

The abstract data model defined here has the objective of representing, in a syntax-independent way, the content that a document must have used to exchange data sets found on the properties that make up an *UrbanDataset*.

3.4.1 Requirements Analysis

The data model was defined taking into account the following requirements:

- the model must be independent of the syntax used for the creation of exchange documents;

- the model must be independent from any protocol / service for the exchange of electronic documents;
- the model must include:
  - the formal description of the *UrbanDataset* used (for example, the specific reference to which is added);
  - information to contextualize the data collected (for example, the system that has them calculated, the time zone of the timestamps, the time when the data was aggregated, ...);
  - the data collected, that is the values of the properties that make up the *UrbanDataset* and whose nature, therefore, depends on its type (for example, the time a car stays on a stretch of road, the internal temperature of an apartment, ...);
- the data model must be able to be used to send data collected on any property type of *UrbanDataset* (it must therefore support heterogeneous "properties");
- an instance of the data model contains data related to a single *UrbanDataset*.

3.4.2 Abstract Data Model

The abstract data model defined to represent documents used to exchange data sets related to the properties that make up an *UrbanDataset* and consists of the following parts:

- **Specification:** contains the information that describe the *UrbanDataset* used (for example the reference to the specification to which it adhere and the properties that compose it);
- **Context:** provides information that contextualizes the transmitted values (e.g. the time zone time stamps);
- **Values:** data collected on the properties that make up the *UrbanDataset*, grouped in rows.

The data model is independent of any syntax (XML, JSON) therfore the implementation of the provided reference (XML Schema and JSON Schema) may not have the same level of aggregation and a 1 to 1 correspondence with the elements presented in the tables representing the model. The abstract model is represented through tables and composed of the following columns:

- **Element:** label chosen for information contained in the model;
- **Description:** textual description of the information;
- **Occurrences:** number of repetitions allowed (cardinality) for the information and mandatory indicator when minimum value > 0;
- **Type:** type of data allowed for the information (eg integer, string, ...);
- **Example:** example value for the information.

The color of the rows of the tables indicates the level of information aggregation:

- **row with white background:** indicates elementary information (eg: "data", "temperature", ...) or, if the element name is preceded by the @ symbol, information that qualifies another elementary information;
- **row with gray background:** indicates aggregate information, made up of several elementary information (eg: "period", "source", ...).

3.4.2.1 The "Specification" Block

The specification block consists of the information given in Table 3.1.

Table 3.1: Specification block: *UrbanDataset*

Element	Description	Occr	Type	Example
specification References	Set of references that allow the identification of the specification of the UrbanDataset used.	1..1	aggregate	
@version	Specification version.	0..1	string	1.0
id	Specification identification code.	1..1	string	UD0000000000

Continued on next page

Table 3.1 – continued from previous page

Element	**Description**	**Occr**	**Type**	**Example**
@schemeID	Identifier of the identification scheme according to which the identifier was defined.	0..1	string	SCPS
name	Name associated with UrbanDataset.	1..1	string	Smart Building Average Consumption
uri	URI that identifies the specification (it can be an urn indicate the domain of the specification, or a url).	1..1	string	`http://ontologia.esample.it/UD0000000000`
property Definition	This element allows you to enter, within the message itself, the description of one of the properties that make up the UrbanDataset.	0..n	aggregate	

3.4.2.2 The "propertyDefinition" Block

The data model provides the ability to define both **elementary properties** (for example: magnitude of a seismic event, latitude, longitude, electricity consumption of a residential unit, heat consumption of a residential unit) is **aggregate properties**, i.e. properties composed of several elementary properties (for example: epicenter of a seismic event, which is composed of latitude and longitude, or total consumption of a housing unit, which is composed of electricity consumption and thermal consumption). The definition of an elementary property must be given by the following information presented int Table 3.2.

Table 3.2: propertyDefinition block: *UrbanDataset*

Element	Description	Occr	Type	Example
property Definition	Element structure in case of elementary property			
property Name	Name of the property.	1..1	string	Average consumption
property Description	Description of the property.	0..1	string	Average electrical consumption of a building, calculated over a specific period
dataType	Type of data with which the property value is expressed.	1..1	string DataType-Code	double
codeList	Reference to a list of codes containing the set of values allowed for this property and their meaning.	0..1	string	
unit Of Measure	Unit of measurement in which the value is expressed.	0..1	string	kWh

3.4.2.3 The "Context" Block

The context block consists of the information indicated in the Table 3.3

Table 3.3: Context block: *UrbanDataset*

Element	Description	Occr	Type	Example
producer	System data (Platform / Solution) that produced the data collected	1..1	aggregate	

Continued on next page

Table 3.3 – continued from previous page

Element	Description	Occr	Type	Example
id	Unique identifier of the system.	1..1	string	Smart Home Test
@schemeID	Identifier of the identification scheme according to which the identifier was defined.	0..1	string	SCPS
timeZone	Time zone of any timestamp present in the document.	1..1	string TimezoneCode	UTC
timestamp	Generation instant of this UrbanDataset (i.e. the moment in which the data was aggregated).	1..1	dateTime	2017-01-30T10: 52: 50
coordinates	WGS84 coordinates of the geometric center of the application network on which the transmitted values were detected.	1..1	aggregate	
@format	WGS84 format in which the coordinates are expressed. Possible options: - Degrees, Minutes, Seconds (WGS84-DMS) - Decimal Degrees (WGS84-DD)	0..1	string FormatCode	WGS84-DD
latitude	Latitude.	1..1	double	43
longitude	Longitude.	1..1	double	49.5215
height	Altitude.	0..1	double	3.0987
language	Language in which the text fields of the instance are expressed.	0..1	string LanguageCode	IT

3.4.2.4 The "Value" Block

The value block is composed of **one or more lines(1 row = 1 property group)**, each containing the information indicated in Table 3.4.

Table 3.4: Value block: *UrbanDataset*

Element	Description	Occr	Type	Example
id	Line number	0..1	integer	1
description	Textual description of the measured values.	0..1	string	Electric consumption daily unit housing
timestamp	Date and time (timestamp) of generation of the data reported in this row.	0..1	dateTime	2017-01-30T10: 52: 50
coordinates	WGS84 coordinates of the object to which the detected data indicated in this line refer.	0..1	aggregate	
@format	WGS84 format in which the coordinates are expressed. Possible options: - Degrees, Minutes, Seconds (WGS84-DMS) - Decimal Degrees (WGS84-DD)	0..1	string Format-Code	WGS84-DD.
latitude	Latitude.	1..1	double	43
longitude	Longitude.	1..1	double	49.52
height	Altitude.	0..1	double	3.09
period	Period during which the data was collected.	0..1	aggregate	
start_ timestamp	Time stamp indicating the start of the period.	1..1	dateTime	2014-01-01T10: 52: 50

Continued on next page

Table 3.4 – continued from previous page

Element	Description	Occr	Type	Example
end_ timestamp	Time stamp indicating the end of the period.	1..1	dateTime	2014-02-28T10: 52: 50
property	Property composing the UrbanDataset. For the structure of the element, see the property section. The element must be repeated for each property that makes up the UrbanDataset.	1..n	aggregate	

3.4.2.5 The "property" Block

The data model determines how to define both the values of the **elementary properties** and the values of the **aggregate properties**.

elementary properties must be expressed in accordance with the following structure:

Aggregate properties must be expressed in accordance with the following structure:

3.5 Summary

In this chapter, we have provided a brief introduction to PELL project along with the objectives of the PELL system. we further discussed the PELL system structure along with the static and dynamic phases of the project. Then, We discussed the collection and management of the static data as part of the PELL project, and how it is maintained by using the census tech sheet. After this, the dynamic phase of the project is discussed in great details. As part of the dynamic phase, we discussed how electric measurements are collected, retrieved, and stored in JSON format in the PELL data lake. Furthermore, the dynamic data model is discussed in details with a special focus on the structure and components such as specification, context and values of the JSON file known as *Counter Reading.*

In the next chapter, we will focus on the first objective: exploration from a software-architecture perspective of a system engineering approach to architect a smart-city platform with BDA.

Chapter 4

Self-Adaptation and Big Data Analytics in Software Architectures for Smart Cities

In this chapter, we address the first objective of this dissertation: exploration from a software-architecture perspective of a system engineering approach to architect a smart-city platform with BDA. To achieve this objective, we present a systematic literature review that investigates the research efforts made for the development of software platforms for smart cities focusing on self-adaptation and BDA. The investigation is made by means of a systematic literature review. We analyze selected studies to get an insight on existing software platforms providing smart city services by revealing features, applicability, and limitations of current software architectural solutions, but also challenges ahead for enabling self-adaptability and machine intelligence.

4.1 Self-Adaptation and Big Data Analytics

Several cities in the world are seeking to become "smart" by managing urban processes and services for citizens via ICT-based solutions established by municipalities and business service providers. Recently, there is a growing trend in considering a *citizen-centric* concept of Smart City and in adopting software platforms [1, 2, 3], which can aggregate as much information and "polymerize" the city services realized via heterogeneous siloed ICT-based solutions by public-private partnerships in specific business domains (such as, waste management, city surveillance, energy consumption, parking, public transport, etc.). These initiatives are giving rise to the concept of *smart city as a platform* where citizens can access the services easily and uniformly thanks to end-to-end APIs, and service providers can easily realize new value-added city services thanks to one source of aggregated information.

Such software platforms for smart cities typically make use of emerging technologies related to IoT, cloud computing, and BD management [4]. IoT-Smart City ecosystems are made of sensors and smart city apps ("citizens as sensors") that generate large amounts of heterogeneous data in real time and space, such as information from city infrastructure monitoring, performance and health events from used tools, and application execution logs. These urban data streams are vital information that can be processed to provide novel services and manage or optimize different concerns of a smart city.

However, several engineering issues related to integrability, extensibility, scalability and context-aware configuration (during startup-time or at run-time), which are typical of modern large-scale data-intensive applications, exist in the design, development and delivery of smart city applications. To this purpose, *self-adaptation* and *big data analytics* are key approaches to handle such issues and develop effective smart city software architectures and intelligent services [3, 53]. At on end, aside from BDA management tools, which are quite mandatory for the aggregation and mining of large volumes of changing and unstructured data, sophisticated and computationally intensive machine intelligence and data analytics solutions are needed to effectively extract emergent characteristics and comprehensible conclusions, and hence provide insights for city decision makers/planners [3, 76].

On the other end, self-adaptation [77, 78] is nowadays considered a key desired capability for modern software systems to handle the uncertainties faced by the system at run-time and ensure that the system meets its functional requirements and the expected Quality of Services (QoS) by autonomously changing its behavior and structure. In the context of Smart City, self-adaptation may help smart city platforms and the underlying digital infrastructures to meet the dynamic requirements of being self-configuring in response to situations and changing needs of the city and its citizens, self-optimizing on the base of key performance indicators of the network pressures and requests for smart city services, being more predictive and capable of transforming massive volumes of data into actionable insights for municipalities, etc.. For example, a city traffic management system may collect data from many sources (like intelligent traffic cameras, cars, traffic lights, etc.) and then optimize the traffic guidance.

The main problem in engineering self-adaptation in such large-scale data-intensive distributed systems is that they need to consider the current situation and possibly predict future situations, when they decide to adapt or not by processing the collected traffic-related data. Self-adaptation in such systems, like smart city software platforms, can be therefore guided by BDA based on a continuous stream processing and analysis of urban data for event predictions.

4.2 Related Work

To begin our review, we first searched for literature surveys and mapping studies focusing on smart city software platforms. In the following, we try to describe all these relevant studies.

There are surveys focusing on the fundamental concepts of the smart city, development of smart city software platforms, smart city data management and analysis, application domains, technologies and methodologies used for the development of applications and so forth. For instance, Gil-Garcia et al. [2] describe existing works discussing the core components of the smart city concept and practical tools used for the assessment of a smart city. They analysed the academic literature to investigate the core components of the smart city as a basis for a comprehensive conceptualization, and a detailed description is provided on the selected practical tools, which were used in the identification of the specific elements or aspects not treated in the academic studies. The primary goal of this work is the creation of a bridge between smart cities research and practice expertise.

Habibzadeh Hadi et al. [3] presented a multi-faceted survey of machine intelligence in modern smart city applications. They provide a partitioned view of the smart city infrastructure into application, sensing, communication, security, and data plans, but put a major emphasis on the data plane as the mainstay of computing and data storage. This work studies the applicability of existing data processing solutions to given applications and services, and in addition to this, it also investigates on a complementary application of data analytics, machine learning, and data visualization to enable machine intelligence in smart city applications.

Santana et al. [79] surveyed the state of the art in software platforms for smart cities. They consider 23 platforms to analyse enabling technologies and functional and non-functional requirements. They classify them into four major categories:

Internet of Things, Cyber-Physical Systems, Big DataBD, and Cloud Computing. On the basis of their findings they also proposed a reference architecture to guide the development of smart city software platforms. They also discussed the potential challenges cited in the literature and future research opportunities in the area.

Welington M. da Silva et al. [80] presented a survey on Smart City software architectures. The main intention of their work was to analyse the requirements handled by the existing smart city software architecture. Furthermore, based on the reviewed architectures, they presented and discussed a set of requirements that is necessary for the implementation of a smart city.

ChuanTao Yin et al. [81] conducted a literature survey on smart cities to understand its definition and application domains. Moreover, they also discussed some enabling technologies for the development of smart city platforms and also proposed a smart city reference architecture model, which is composed of four different layers: data acquisition, data vitalization, common data and services, and domain application.

Ruben et al. [82] presented a survey to analyze the work done so far in the domain of smart city. They provide an overview of the smart city definitions discussed in the literature, technologies and methodologies used in the development of smart city software platforms, smart city application domains, and at the end they highlight the research challenges and future opportunities for smart cities.

Eiman Al Nuaimi et al. [5] conducted a review to analyze the application of BD to support smart cities. Their review discusses the challenges and benefits of incorporating BD applications for smart cities. In addition to this, they identify the requirements that support the implementation of BD applications for smart city services.

Yosra Hajjaji et al. [4] reviewed BD and IoT-based applications in smart environments to identify key areas of application, current trends, data architectures, and ongoing challenges in these fields.

We have also found some other studies focusing on self-adaptability in the context of different niche application domains, such as in cyber-physical systems [83] and in mobile applications [84].

Considering all the mentioned works from both the two perspectives of self-adaptability and BDA, we found that the second perspective has received a lot of attention in the last few years in the smart city context, but we have not found any survey or review article specifically focusing on the first perspective and on the link between the two perspectives – the intended goal of our review.

4.3 Study design

In this work, we followed the classical three-stage process for running systematic literature reviews [85], namely: *Planning*, *Conducting*, and *Reporting*. This section briefly reports the activities carried out in the two main stages (conducting and reporting) for this study.

To allow replication and verification of our study, a replication package (including the protocol, the selected studies, and the collected and analyzed data) is publicly available as Google Drive repository [86].

4.3.1 Conducting Review

This stage consisted of three main steps carried out in sequence, namely definition of research questions, study search and selection, and data extraction and synthesis. The specification of these steps are described in the subsequent sections.

4.3.1.1 Identification of Research Questions

The following research questions shaped the whole investigation study:

- **RQ1**: *Which architectural design patterns are used in the development of software platforms for smart cities?* Rationale: by answering this research question we aim at characterizing the diversity of architectural patterns adopted. This may help evaluate possible emerging architectural styles for smart city platforms and identify a possible reference architecture.

- **RQ2**: *How is self-adaptation realized in software platforms for smart cities?* Rationale: the answer to this question will help understand the extent to which current software platforms for smart cities target (self-)adaptation, and which adaptation mechanism and degrees of adaptability they adopt.

- **RQ3**: *Which concerns (goals) of adaptation are handled in software platforms for smart cities?* Rationale: this question will help to determine which types of adaptive features, as well as self-* capabilities associated with autonomic computing [87] are pursued to handle one or more real concerns in one or more domains, possibly with the help of BDA.

- **RQ4**: *Which kinds of services are developed for smart cities using BDA?* Rationale: the answer to this question may help categorize the application domains requiring BD processing techniques, regardless of whether or not self-adaptation is realized.
- **RQ5**: *Which BD technologies are used in services for smart cities?* Rationale: the answer to this question may help identify which types of technologies/tools for BD applications can be more suitable for which domains.

4.3.1.2 Search and Selection Process

The search and selection process has been designed as a multi-stage process, so to be sure to include all potentially relevant studies. In this review, we have performed manual search as well as automatic search, as follows.

1. *Manual search:* The manual search was performed with the help of Google Scholar. We started with some pilot studies about well-known software platforms for Smart Cities that we identified based on our knowledge of the field. Then, to enlarge this initial set of studies we considered those papers cited by each study. The initial screening of these additional studies was based on their title and abstract, whereas the final decision about their inclusion or not was based on their content. As a result, we obtained 15 potentially relevant studies. Details of these selected papers are reported in Table A.3 of A.
2. *Automatic Search:* The automatic search was performed on five different digital libraries due to their wide and universal adoption in the academic communities. These digital libraries are: Scopus, ACM Digital Library, IEEE Explore, Science Direct, and Springer Link.

   The following search strings were used to search into the aforementioned digital libraries according to the two groups of formulated research questions, namely {RQ1,RQ2,RQ3} and {RQ4,RQ5}, respectively.

   First search string(s): s_1, s_1 ***AND*** s_2 where:

   $s_1 \equiv$ (*"smart city"*) ***AND*** (*platform* ***OR*** *software* ***OR*** *system* ***OR*** *framework*), and

   $s_2 \equiv$ (*adaptation* ***OR*** *adaptive* ***OR*** *"self-adaptation"* ***OR*** *"self-adaptive"* ***OR*** *"MAPE-K"*)

Second search string: (*"smart city"*) ***AND*** (*application* ***OR*** *service*) ***AND*** (*"urban data analytic"* ***OR*** *"big data analytic"* ***OR*** *"machine learning"*)

As preliminary screening, the search strategy targeted peer-reviewed journal papers, conference papers, and magazine papers, and was applied on the title, keywords, and abstract. The considered time frame ranges from 2010 to mid-2021. Of course, duplicates of studies have been removed as well.

In order to further filter the studies resulting from the this preliminary screening, we defined the following inclusion and exclusion criteria.

Inclusion Criteria:
– Studies focusing on the development of software platforms for Smart Cities.
– Studies on software platforms for smart cities adopting self-adaptation mechanisms.
– Studies in which smart city services are developed by using ML techniques and BDA.

Exclusion Criteria:
– Secondary or tertiary studies (e.g., systematic reviews, surveys, etc.).
– Studies in the form of editorials, tutorial, and poster papers, because they do not provide enough information.
– Commercial initiatives of smart city platforms by ICT companies (such as IBM, Microsoft, Huawei, etc.) since in the literature we found very little technical information about them.
– Studies not written in English.

When going through each selected study in detail for starting with data extraction, some other studies that were semantically out of the scope of this research were excluded. Moreover, as presented above, we have divided our main search string into two different sub-strings. Hence, as a final cross-check, papers resulting from the first sub-string which were on the scope encompassed by the second sub-string were included in the final sample of studies for the RQ group {RQ4,RQ5}, and vice-versa. The results of the overall automatic selection process for the two different groups of search strings is shown in Fig. 4.1. We obtained two final sample sets: 41 primary studies for the first search string(s) and 112 primary studies for the second search string. These studies are listed in Table A.1 and Table A.2, respectively, of A.

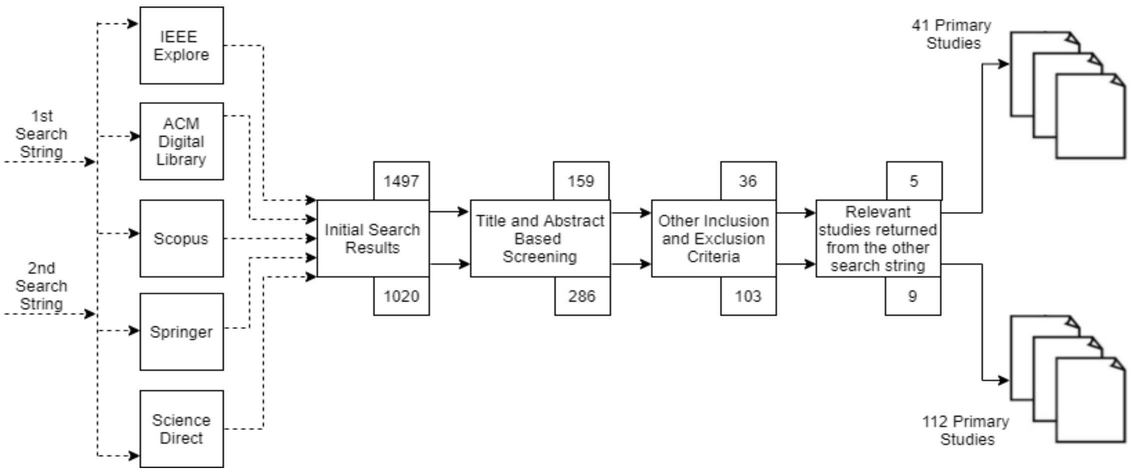


Figure 4.1: Multi-staged automatic search and selection process

4.3.1.3 Data extraction and synthesis

The goal of this phase is to extract the data from the primary studies to answer the research questions. This data was extracted by exploitation and synthesis of useful data features from the research questions (see Section 4.3.2). This data features were searched manually from each selected primary study. Specifically, we performed a combination of content analysis and narrative synthesis for understanding and interpreting the findings correctly. The results of data synthesis are presented in detail in Section 4.4.

4.3.2 Reporting Review

Data extracted from the selected primary studies was collected in two separate spreadsheets (one per sample set). All spreadsheets include, for each paper, the following main fields: paper ID, paper title, authors, a brief summary, and so on. Additionally, different fields were inserted for the data features used to answer the research questions depending on the sample set. The data features for the RQ set {RQ1,RQ2,RQ3} of the first search string (concerning software architecture and self-adaptation mechanisms of smart city platforms) include: dominant architectural design patterns (for RQ1); adaptation mechanism (MAPE-K loop, self-organizing agents, mixed), degrees of adaptability, and other dimensions according to the classification framework by Krupitzer et al. [88] (for RQ2); and concerns of adaptation, application domain, and big-data analitycs adoption (for RQ3). The data features for the RQ set {RQ4,RQ5} of the second search string (concerning smart city services andBDA) includes the following main fields: application domain (for RQ4), and BD technologies (for RQ5). For more details on the specific data features extracted, see the spreadsheets available in the replication package.

4.4 Results and Analysis

This section, by deeply analyzing the primary studies listed in A, provides an answer to the research questions presented in Section 4.3. The following subsections report the results of our review organized by the two RQs groups {RQ1, RQ2, RQ3} and {RQ4, RQ5}.

4.4.1 Answers to research questions {RQ1,RQ2,RQ3}

As results from the manual and automatic search, 56 primary studies (from both manual and automatic searches) were considered totally for data extraction. The following sub-sections detail the results.

4.4.1.1 Dominant architectural design patterns (RQ1)

Software platforms for smart cities can be analyzed from different areas of scientific knowledge. From the perspective of a software architect, we identified the common architectural design patterns used on the analyzed software platforms. Fig. 4.2 presents the design patterns identified in the studies, and their frequencies.

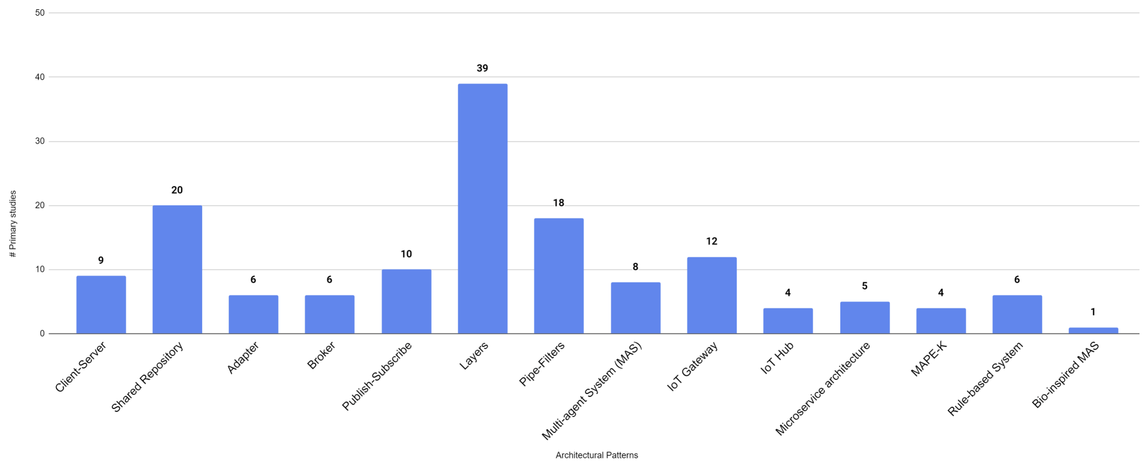


Figure 4.2: Distribution of architectural design patterns among the reviewed studies

The most recurring pattern is the *Layers pattern* together with the *Pipe-Filters*, *Shared Repository*, and *Broker*, which are very common in the BD context due to the need to manage and extract useful information from the massive volume of urban data generated at real-time frame rates. Other recurring patterns are the *Client-Server* and *IoT Gateway*, commonly used in the IoT Cloud-Edge context.

A small number of case studies (only 4) adopt a MAPE-K feedback control loop architecture[87], but (as better explained in the next paragraph for question RQ2) there are other studies that realize self-adaptation differently.

On the base of our analysis, a common architecture model (combining several of the above mentioned patterns) is that of a *layered architecture* where a central middleware layer aggregates data and connects third-parties vertical applications and infrastructures (ICT solutions targeted to specific domains and that use different technologies). Value-added services are built on the top of the middleware layer (dashboards, data management and analytics, etc.) Examples of smart city platforms adopting such architecture model to support integrability and extensibility are described in the studies [89, 90, 91, 92]. Of course, the (relatively new) concept of *microservice architecture* play a central role in the most recent studies (see, for example, the platforms InterSCity [93] and iQAS [94]) to achieve interoperability between heterogeneous parties and independence from the underlying infrastructures.

In most cases, the architecture was designed for specific purposes and requirements. No reference software architecture has emerged yet as de facto standard. There have been some candidate architecture models [95, 79, 80], but all have been superseded over time. Our vision is that a new concept of Smart City is emerging that is that of a *Sociotechnical System of Systems*[96], since there are several sociotechnical aspects to be taken into account, such as the number and heterogeneity of urban data sources and flows across inter-connected cross-sectoral systems, the number and diversity of application domains and city infrastructures, security and privacy issues related to the processing of sensitive data, uncertainty due to social dynamics, etc.

4.4.1.2 Dominant adaptation mechanism and degrees of adaptability (RQ2)

A self-adaptive software system is capable of modifying its runtime behavior autonomously in the face of a changing environment in order to achieve specific objectives and quality properties. Different self-adaptation mechanisms exist to engineering self-adaptation into software systems; the most notorious ones include the architecture-based MAPE-K control loop model, agent-based approaches modelling systems as a collection of autonomous interacting agents, nature-inspired adaptation, and reflecting programming or reflection [88, 97].

In order to answer to the research question RQ2, we investigated whether the collected software platforms perform self-adaptation to support decision-making in the context of smart city, and how it is engineered. To this end, we followed the dimensions for describing self-adaptation in software systems proposed in the taxonomy by Krupitzer et al. [88], namely: *Adaptation Control*, *Time*, *Reason*, *Level*, and *Technique.* We choose such a taxonomy among others since it reasonably answers the *5W + 1H* questions introduced by M. Salehie et al. in [97][1] for eliciting adaptation requirements, and includes context adaptation.

1. *Adaptation Control:* Based on the analyzed studies, we first investigated whether self-adaptation is supported or not, and therefore tried to understand the *Adaptation Control* mechanism ("How to adapt?"). As mentioned before, only 4 software platforms [76, 98, 94, 99] (papers[2] P11, P12, P17 in Table A.1, and P15 in Table A.3 of A) adopt the MAPE-K feedback control loop model as adaptation mechanism. With this limited number of studies, we therefore investigated whether the MAPE-K control loop model is supported "implicitly", i.e., by cause-effect dependencies of interactive software components that actually realize closed feedback control loops for monitoring and actuation (though the study does not explicitly states their role of MAPE components), or if another type of adaptation mechanism is used (e.g., self-organizing agents). Remarkably (see Fig. 4.3), most of the platforms adopt MAPE-K loops implicitly (64.3%), while 19.6% of the studies do not detail (in the text or in the software architecture) any type of adaptation. Only 7.1% use an agent-based mechanism to realize adaptation, also combined with explicit (1.8%) MAPE-K loops, and 1.8% apply a nature-inspired approach.

   For the studies using MAPE-K control loops implicitly, we proceeded with a fine-grained analysis to get more insight on the "completeness of control" by measuring it in terms of supported MAPE functions. The results are shown in

[1]As in [97], the question: "Who has to perform the adaptation?" is not answered here, since self-adaptability refers to the ability of a system to automatically adapt to changes.

[2]We use the notation Px to refer to a surveyed paper listed in Appendix A.

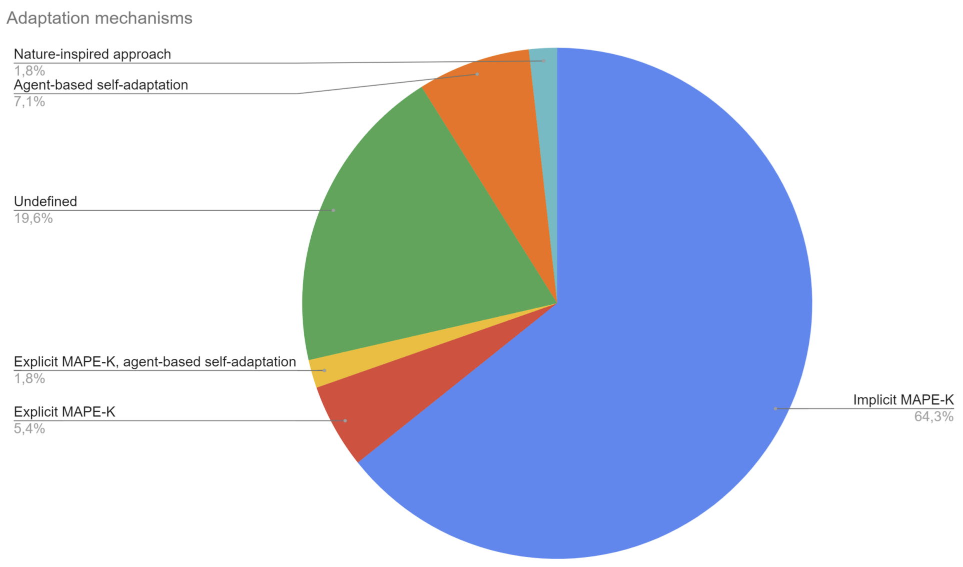


Figure 4.3: About the adaptation mechanisms

Fig. 4.4) by denoting, for example, with MA solutions supporting only sensing and analysis, with PE only planning and actuation, and so on.

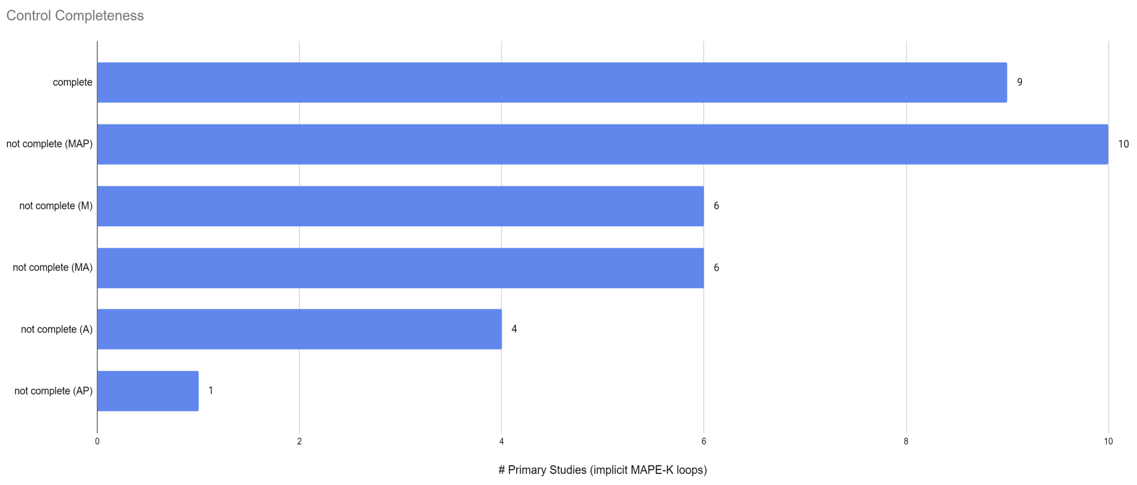


Figure 4.4: About control completeness of studies using implicit MAPE-K control loops

A good number of approaches cover all the MAPE functions, thus closing the feedback control loop. However, some other approaches support only one MAPE function, and therefore their proposed architectures do not have a broad scope to be used as free-standing solution.

2. *Time dimension:* This aspect is related to the when-question ("When should

we adapt?") and distinguish two time dimensions *before* or *after* the need for adaptation for describing the temporal aspect of adaptation [88, 100].

The simplest form is reactive adaptation; usually, self-adaptive systems tend to be reactive, i.e. they adapt in response to changes without anticipating what the next adaptation needs will be. Proactive adaptation addresses the limitations of reactive adaptation taking into account not only the current conditions, but how they are estimated to evolve when deciding to adapt. So, based on predicted events, proactive adaptation adjust the system in advance. With proactive adaptation, the monitored data is used to forecast system behavior or environmental state. Proactive adaptation requires computationally intensive reasoning that may be realized as part of the Analysis and Plan phases of a MAPE-K loop with the help of BDA and machine learning algorithms [101].

In smart city scenarios, reactive adaptation could lead to resource waste, transient unavailability of services and behavioral fluctuations. In contrast, proactive adaptation can help in preventing the occurrence of problems and/or in mitigating the effects of upcoming problems by dynamically re-planning adaptation actions in advance. As an example, if during the execution of a freight transport process a delay is predicted, faster transport services (such as air delivery instead of road delivery) can be scheduled to prevent the delay.

In this review, the distinction in proactive and reactive to denote when adapt (before or after the need for adaptation) is used for describing the temporal aspect of adaptation in the considered studies. Fig. 4.5 summarizes the results. As expected, the reactive nature is the most popular form. Instead, advanced formal reasoning and analytics for proactive adaptation in the context of smart city applications is still not explored enough, though it could play a central role into making self-adaptation in advance (on the base of forecasting and predictions) for critical services in domains such as energy consumption, transportation, and epidemic containment (to name a few).

3. *Reason:* This dimension is related to the why-question ("Why do we have to adapt?"). Adaptation can be triggered by changes in the managed technical resources (e.g., a defect of a hardware component, a software fault, or the availability of an alternative network connection or service), in the environment or context (e.g., the state change of a context variable like air quality),

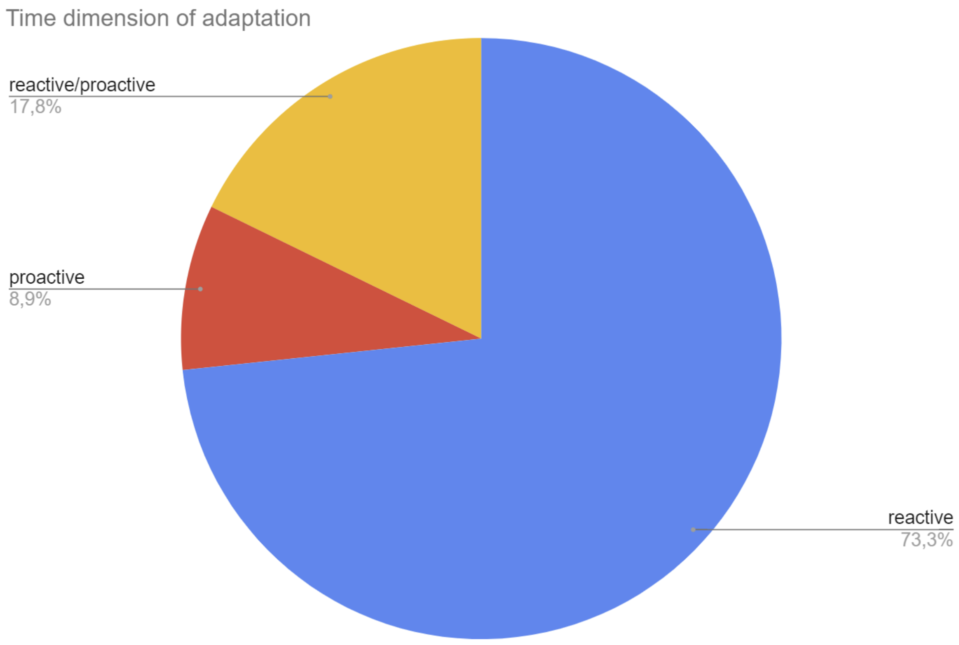


Figure 4.5: Time dimension for adaptation

or in the users (e.g., in the user preferences). The answer to this question is crucial to the software architect because it influences the design of the adaptation logic; the reasons determine the elements that have to be monitored and the adaptation/control actions to actuate. The left side of Fig. 4.6 shows the distribution of the primary studies among the main categories of changes (*context*, *technical resource*, *user*, and *mixed*) as identified in [88].

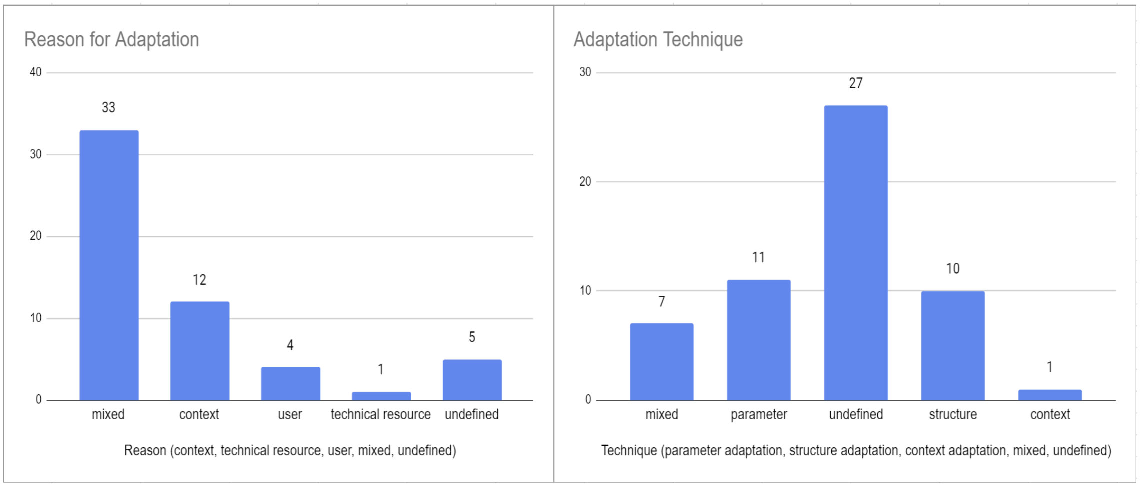


Figure 4.6: Reason and Technique dimensions for adaptation

In most of the examined studies, the trigger for the adaptation process are changes in almost all categories of observed elements (the technical resources, the context, and the users). This is an expected result since all categories of changes are common in the urban landscape of data and services. For a certain number of studies, it was not possible to determine the reason for adaptation, hence we have classified them in as a separate category (*undefined*).

4. *Level:* This aspect is related to the where-question ("Where do we have to implement change?"), i.e., it aims at identifying the levels, where the adaptation should take place. These levels include technical managed resources (such as hardware, application, middleware, physical network infrastructure, logical communication, etc.), the system environment, and the user(s) level. As this attribute resulted to be too fine grained for the purpose of our study, we do not report here any significant result. However, as explained in the next paragraph, we were able to identify the general type of adaptation actions to carry out.

5. *Technique:* This dimension is related to the what-question ("What kind of change is needed?"). In this review, we considered the distribution of the primary works (see the right side of Fig. 4.6) along the categories of techniques for adaptation as in [88], namely: *parameter adaptation*, *structure adaptation*, and *context adaptation*. Parameter refers to adaptation through the change of parameters. Structural adaptation enables the exchange of algorithms or system components dynamically at runtime. Context adaptation refers to any changes in the context. Combinations of techniques (*mixed*) in one adaptation plan is also possible, e.g., changing parameters of one component and adding further ones. We observed that studies are evenly distributed among all categories, except for modifying the context variables. For this last category, we found only one architecture example [102] that supports both real-time and historical data analytics in the transportation and energy domain to extract and spread actionable knowledge (e.g., generating a complex event representing an anomaly, which can then in turn be used to notify the user as well). A high number of studies were unclassified (*undefined*), mainly because of the incomplete adaptation control.

4.4.1.3 Adaptation concerns of approaches adopting BDA (RQ3)

When analyzing the primary studies, we also traced the application domains and concerns of adaptation in smart city platforms.

In order to relate the studies that adopt BD technologies with those performing self-adaptation, we specifically searched for those using machine intelligence (BD and AI together) for adapting [103]. The left side of Fig. 4.7, report the results of only those platforms that employ BDA to process data and extract useful knowledge for planning and guiding adaptation.

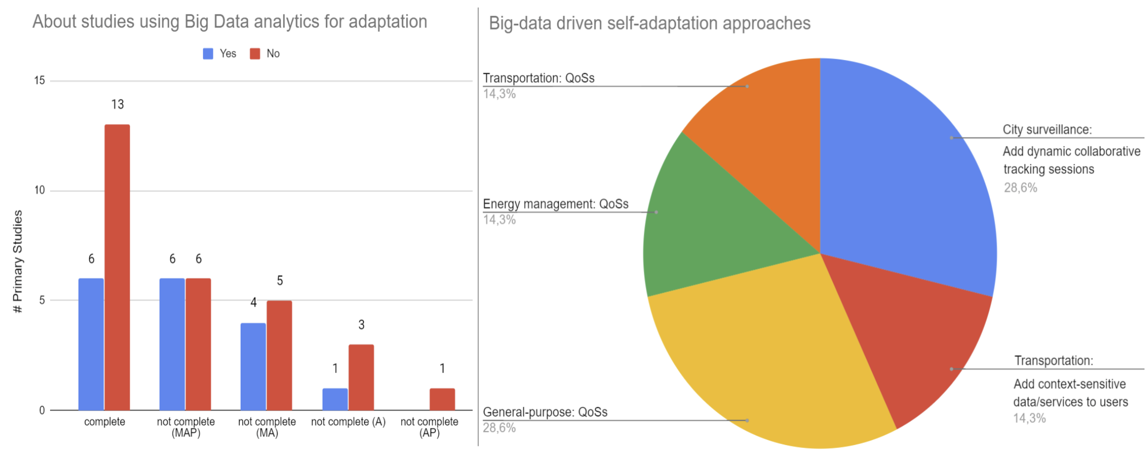


Figure 4.7: About studies using BDA to guide self-adaptation

Among the 45 studies partially supporting adaptation (considering those supporting at least the MAPE function "Analysis"), 17 studies adopt BD technologies, and among these only 6 studies (papers P18, P20, P24, P32 in Table A.1, P2 in Table A.3, and P1 in Table A.2 of A) adopt a complete *big-data driven self-adaptation* approach. For these last six studies, Fig. 4.7 (right side) shows their application domain together with their concerns of adaptation. The dominant application domains are those related to transportation and city surveillance. Primary concerns of adaptation are QoSs (response time, reliability, privacy and security) and the provision of *context-aware* services such as delivering traffic- and transport- related data/services to users depending on the user context, and adding collaborative tracking/monitoring surveillance sessions dynamically. Two studies do not mention any specific domain (papers P32[90] in Table A.1, and P2[104] in Table A.3) since their adaptation mechanisms are broadly applied to several domains, but act less autonomously and/or perform simple adjustments (like sending alert messages and/or generating recommendations).

4.4.2 Answers to research questions {RQ4,RQ5}

In order to answer to RQ4 and RQ5, we considered 112 primary studies as selected by the automatic search, and 2 further studies from those selected by the manual search (papers P1 and P2 in Table A.3 of Appendix A).

4.4.2.1 Smart city services (RQ4)

We investigated the application domains we encountered in our primary studies for the smart services that have been developed by leveraging the benefits of BD technologies. Literature findings (see Fig. 4.8) show that the majority of smart services are offered in the transportation domain (36.28%), hence the development of intelligent transportation systems is an active research area. Other (almost equally covered) application domains include: parking, healthcare, energy optimization, emergency management, road-surface monitoring, tourism, public safety and security, environmental monitoring, water, and mobility. Other minor appearances are also reported (9.73%). In view of the increasing importance of BD, there could be new opportunities for the provision of smart and intelligent services in these least represented domains and for identifying new areas for improvement.

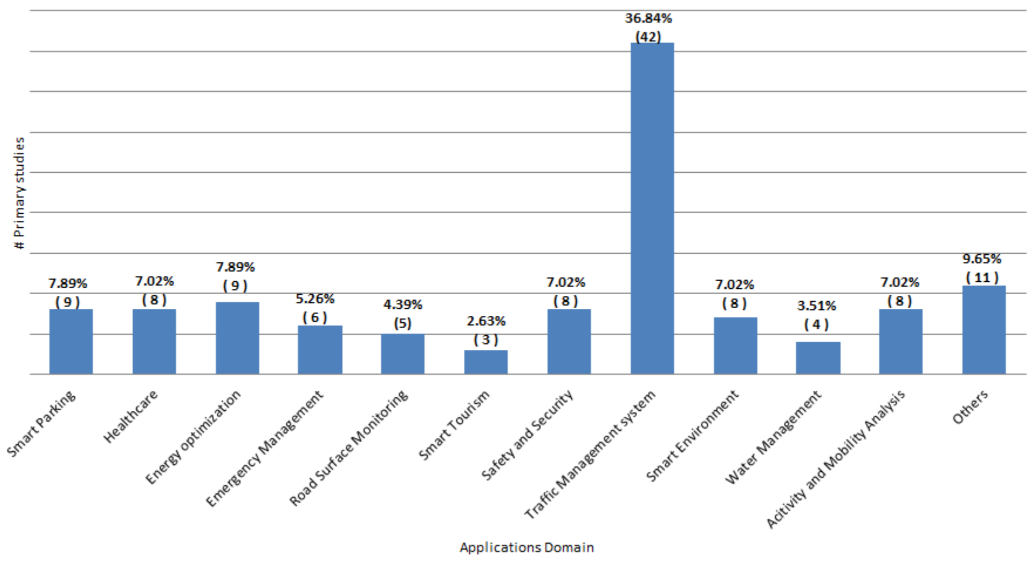


Figure 4.8: Distribution of application domains among the reviewed studies

4.4.2.2 BD technologies and smart city services (RQ5)

The primary goal was to investigate the adoption levels of BD technologies in the development of smart city services. The primary studies were extensively analysed to collect some aspects, and here we present the analysis results accordingly.

First, we categorized smart city services according to the three classes: (i) services that use urban data (without employing BD technologies), (ii) services that collect and aggregate urban BD (in the form of structured and unstructured data) and also analyze and extract value from them by means of Artificial Intelligence (AI) and analytics frameworks (the *city intelligence*), and (iii) services that apply BDA on urban data provided by third parties. The results of this investigation are shown in Fig. 4.9.

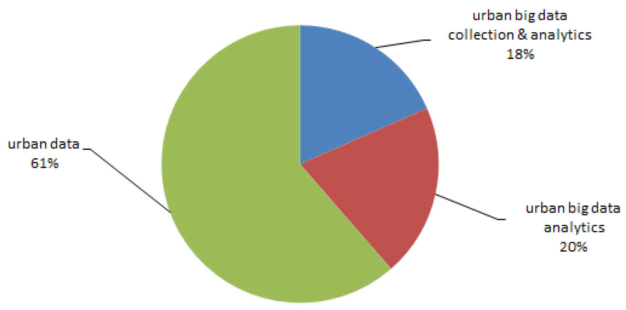


Figure 4.9: Smart city services and BD adoption levels

We observed that the majority of services (61%) do not adopt BD (i), hence not BD technology helped the smart city sector to grow. Only 20 studies (18%) device solutions using BD technologies (mostly open-source, like NoSQL databases and parallel data processing tools such as Apache Hadoop and Spark combined) to both collect large urban data sets and then make analytics on them (ii), and 23 (20%) services perform analytics to analyze urban BD collected by third party platforms or data provided by the city administration authorities (iii). This result may be due to several challenges that companies, in general, face in implementing BDA. These include the lack of data scientist skills (the technology is still in its infancy), and the ever-present security-governance. Data quality and its consistency also presents a big challenge to the management [105].

Finally, we identified the application domains of the smart services adopting BDA. The findings of this analysis are presented in Fig. 4.10. In this analysis, we discovered that traffic management is the most prominent domain in which 43%

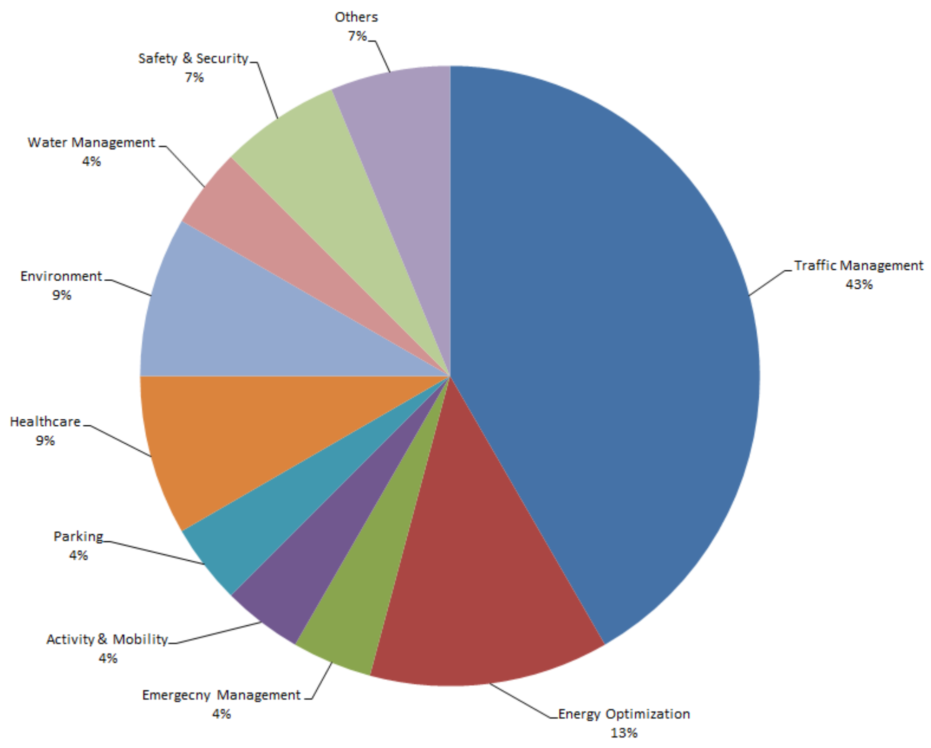


Figure 4.10: Application domains of smart services based on BDA

services are developed by using the BDA. The second common application domain is energy optimization, in which 13% of the services are based on BDA.

4.5 Threats to Validity

Some elements, common to all systematic literature reviews, could threaten the accuracy and validity of our study and therefore of our findings. The potential threats to the conclusions that we drew are as follows:

External Validity Since the study selection phase primarily was conducted by three researchers, there might be possibilities of biased subjectiveness in finding primary studies that may lead to incomplete data collection and/or not representative studies. Defining extensive and clear selection criteria through a systematic protocol helped us to mitigate this threat. In addition, at each stage of the study, all authors have determined unclear questions and discussed them together, and ensured that at least one other reviewer has reviewed the data extraction work.

Internal validity In our study, we chose to consider only peer-reviewed published research as we think it is an established requirement for quality publications. However, this position may have led to miss some relevant works from industry, which usually are in form of white paper report or brochure. Despite our academic focus, we have observed that many of the selected primary studies have authors with industrial affiliations. Moreover, to better mitigate this threat, we are planning a follow-up study using the same research questions to analyze the state of practice and discover if both the academy and industry are aligned together.

Construct validity This threat regards the validity of the obtained data and the research questions. For the systematic literature review, it mainly addresses the selection of the main studies and how they representatively answer the chosen research questions. We tried to mitigate this threat in several ways. First, we followed well-known systematic literature review guidelines. The automatic search was performed on several electronic databases so as to avoid the potential biases. We manually carried out the final selection process using the chosen inclusion and exclusion criteria, reported in Section 4.3.

Conclusion validity The background and experience of the researchers may have introduced some prejudices and a certain degree of subjectivity in collecting and analyzing data. To mitigate this threat, the researchers extracted and analyzed data strictly according to the screening strategy, and further discussed the differences of opinions. Moreover, in order to favour the objectivity of the data extraction results related to the self-adaptation aspect in smart city software platforms, our classification framework was based on established modelling dimensions of the taxonomy by Krupitzer et al. [88].

4.6 Research challenges and future research research directions

4.6.1 Research challenges

Based on the key findings aforementioned, we discuss some research challenges in this subsection.

1. *Emerging reference architecture:* Throughout this work several smart city software architectures were analyzed. We found no reference software architecture exists yet. There is no a clear consensus, in fact, on what requirements a smart city software architecture must meet, regardless of how it is implemented. This is mainly due to the fact that cities and their citizen generally have different characteristics and needs, and existing vertical city software applications/infrastructures have to be reused and connected. So the same architecture model does not apply to all cities. Only a number of common key aspects to consider during the design of a smart city software platform have emerged from our study and also from other similar ones (such as [80, 79, 106, 3], to name a few). These aspects include the principles of *Enterprise Architecture*, unified management of the heterogeneity of urban data sources (humans, devices, IoT network infrastructures and sensors), continuous real-time monitoring and analysis of large amounts of urban data, self-adaption and autonomous control capabilities.

2. *Self-adaptability:* Another point to consider is that a few number of software platforms expose adaptive features. Most of the architectures that support adaptability do not explicitly adopt a MAPE-K control loop architecture for self-adaptation and/or do not support complete MAPE loop functions. This indicates that the field is young, and more advances will emerge in the next years.

   In particular, new horizons and opportunities for smart cities could be revealed by the application of technologies for *decentralized self-adaptive computing* (at the Edge, in the Fog, and in the Cloud) through the integration of Cyber-Physical-Systems, Social-Technical systems, IoTs, and self-adaptive mobile apps, which are areas of growing scientific and practical interest [80, 83, 107, 84].

3. *BDA and machine intelligence:* BDA is another emerging technology that has a huge potential to enhance smart city services by transforming city information into city intelligence. As growing digitization becomes perfectly integrated to our daily lives, enormous heterogeneous amounts of data collected everyday can be fruitfully used in several application domains [5]. By employing the applications of statistic analysis, predicting modeling and many others analysis techniques business decisions can be enhanced.

Despite this potential, BDA has only attracted attention in a rather restricted range of application domains, and its joint application with self-adaptation mechanisms is rarely investigated. We argue that self-adaptation in smart city software platforms can be guided by BDA based on a continuous stream processing and analysis of urban data for event predictions and decision making in a proactive way.

4.6.2 Future directions

The proposed systematic review and its outcomes offer the insight to proceed with further critical analysis. As future work, we want to select a core subset from the examined architectures, considering those with a broad scope that may lay foundation to a reference architecture in the smart city context. In particular, we want to examine how the reported architectural patterns affect or benefit the accomplishment of requirements of quality attributes in smart city services.

We want also to analyze the state of practice by looking at smart city initiatives by the major ICT companies (such as IBM, Microsoft, Oracle and Huawei) to discover if both the academy and industry are aligned together.

As it is necessary to move from a reactive logic of urban management to a proactive logic enabling preventive actions, in the future we want also to analyze more in-depth all forms of data analytics (*descriptive*, *diagnostic*, *predictive*, and *prescriptive* analytics) and put them in relation to the time dimension of adaptation [88].

Moreover, we want to investigate better on models and advanced analytics that take into account uncertainty in their decision making.

4.7 Summary

In this chapter, our goal was to assess the state-of-the-art software platforms for smart cities at the architectural level, focusing on self-adaptation and BDA, and the relation between the two. We conducted a systematic literature review by searching major scientific databases, resulting in 154 primary studies from 2010 to mid-2021, after the application of carefully-defined exclusion and inclusion criteria. Our findings give insights on existing software platforms providing smart city services by revealing features, applicability, and limitations of current software architectural solutions, but also challenges ahead for enabling self-adaptability and machine intelligence. Findings from this study may help both researchers and practitioners of the field. City and municipality managers and ICT providers may also benefit from this study for making investments decisions and engineering software services for the Smart Urban Digital Ecosystem.

The findings of this study help us to recognize most common architectural styles to design and implement a BD-centric software architecture for tracking energy consumption data and census data in the context of public street lighting. In the next chapter, we will focus on design and development of the SCP and deals with the second objective of this thesis: descriptive analytics, formulation and implementation of KPIs to analyze, measure and monitor electric energy consumption of public street lighting plant.

Chapter 5

From Big Data to Smart Data-centric Software Architectures for City Analytics: the case of the PELL Smart City Platform

In this chapter, we describe our experience in designing and implementing a Smart Big Data-centric software architecture for tracking energy consumption data and census data in the context of public street lighting. We start by discussing the proposed SCP, which is developed in collaboration with the Italian national research agency ENEA[1] as part of the PELL (Public Energy Living Lab)[2] project; in brief: the PELL SCP. To this end, PELL SCP, deals with the second objective of this thesis: descriptive analytics, formulation and implementation of KPIs to analyze, measure and monitor electric energy consumption of public street lighting plant. We provide conceptualization examples of the lighting KPIs to measure and monitor electric energy consumption of public street lighting plants in the PELL SCP. A concrete examples of our processing pipeline including evaluation example of lighting KPIs is also presented.

[1]ENEA Italian National Agency for New Technologies, Energy and Sustainable Economic Development `https://www.enea.it/en`

[2]`https://www.pell.enea.it/enea/`

5.1 Big Data in the context of public street lighting

BD technologies require advanced techniques to efficiently process large volume of data (both structured and unstructured) within limited run times. Beside these general management aspects, which are common to almost all types of data, there are several significant challenges in software development linked to the processing of BD depending on the data specificity being generated and processed. In this Section we outline the main ones related to energy consumption in the public street lighting domain. Specifically, first we present the PELL project and then describe the type of BD generated in this application domain.

5.1.1 Big Data in the PELL SCP

We describe the type of BD generated in the PELL SCP in terms of the well known *5V* dimensions, namely *Volume*, *Velocity*, *Variety*, *Value*, and *Veracity* usually used to characterize the concept of BD [108].

1. *Volume:* The amount of data being generated is very high and is related to an energy-consuming public infrastructure, namely the urban public lighting plants and their energy consumption status.

2. *Velocity:* The speed at which data are being generated, accumulated, retrieved and processed depends on the type of analyses. Some analyses may involve short term periods (e.g real time, every minute, hourly), others may require a medium to long time window (e.g. weekly, monthly, yearly).

3. *Variety:* Data being generated include: *static data*, for instance data defining the specific characteristics of the lighting plants including the geographic position/coordinates (for the purpose of its physical delimitation), Points of Delivery (POD), electrical panels, associated luminary equipment (e.g., lanterns, street armors, floodlights, street furniture, etc.), the type of area (vehicular traffic, pedestrian circulation and cycle circulation), and road type (classified according to UNI EN 11248 [3] and evaluated according to UNI EN 13201-2[4] for taking into account the visual needs of road users, and environmental aspects of road lighting); and *dynamic data* that are all the electric measurements

[3] `http://store.uni.com/catalogo/uni-11248-2016`
[4] `http://store.uni.com/catalogo/uni-en-13201-2-2016`

provided by utilities and collected at electrical panel level and aggregated at POD level.

4. *Value:* The data being generated and collected are useful for diagnostics and benchmarking, such as evaluation of lighting performance with the verification of the luminaires efficiency based on the road context where are located, energy requalification simulations and economic/financial evaluations of the lighting plants, visual analysis across multiple spatial and time scales, and energy consumption estimation (to name a few).

5. *Veracity:* Regards the inherent data correctness subject to variation. The management of multiple data sources from several municipalities and different lighting points (e.g. modern LED modules and old incandescence lamp) could lead to lack of reliability in some data sources, moreover, the presence of noise in the network communication are all causes of uncertainty and imprecision in the data.

5.2 Related work

The scientific community has been actively involved in the design and development of intelligent systems to improve the energy efficiency of public lighting. We here list the most relevant works that propose tools/software platforms based on BD technologies in the specific context of public lighting, and works focused on defining KPIs for energy consumption in the same context.

The adoption of KPIs for the continuous monitoring of energy consumption is currently a management practice adopted by many operators and at different organizational levels [109] [110] [111]. In fact, monitoring and evaluation of energy-related KPIs provide means for anyone interested and involved (providers, decision makers and final users) to effectively assess energy use and identify needs and opportunities for improvement. Nevertheless, at the best of our preliminary analysis of the pertinent literature, the use of KPIs is not widely explored yet in this context, probably due to the granularity and complexity of the public lighting infrastructures.

Tagliabue et al. [112] use KPIs to provide a structured proposal for implementing the lighting systems of the neighborhoods and to support the city of Milan (Italy) with an effective strategic plan for developing the renovated university campus area. Grilo et al. [113] use standard KPIs to evaluate the quality of service for Low

Voltage (LV) grid as is the one typically used to supply the public lighting. Using an e-balance management system that processes information from neighborhood householders' smart meters, by means of KPIs, they evaluate the quality of supply.

In [114], Carli et. al described a decision support system for smart city energy governance, the Urban Control Center, based on a KPIs dashboard allowing viewing the city energy indexes. The set of KPIs, selected by the public administration, is automatically updated when the original data changes and provides a high-level information, which can be investigated. Within the strategic energy governance provided by the dashboard, the KPIs for the management of the public lighting are monitored in a panel devoted for this specific urban energy subsystem.

From this preliminary state-of-the-art review, it seems there does not exist a study in which BD technologies and KPIs are explored for the energy management of public street lighting. Instead, in our work we have been developing novel energy KPIs for public street lighting by exploiting a BD processing platform.

5.3 Proposed Smart Big Data architecture: the PELL SCP

The proposed BD processing architecture is shown in Figure 5.1. We followed a typical data pipeline architecture style for BD systems that need to process data in near real-time [108] [9], and adopted Apache Spark as main open-source technology for large-scale data processing and analytics. This architecture pattern allowed us to reduce the design complexity and effort in developing such a software platform. It also allowed achieving the desired engineering characteristics of *scalability* as the amount of ingested data increases, easy data *accessibility* (preferably through an analytic query language), and *efficiency* in term of readiness of data and analytic results with low latency [108].

The conceptual five stages (collection, ingestion, preparation, computation, and presentation) of a BD pipeline [108] are grouped into three data layers: the i) *Data sources and ingestion layer*, the (ii) *Data integration and processing layer*, and the (iii) *Data presentation layer*. The following paragraphs provide a detailed description of these layers.

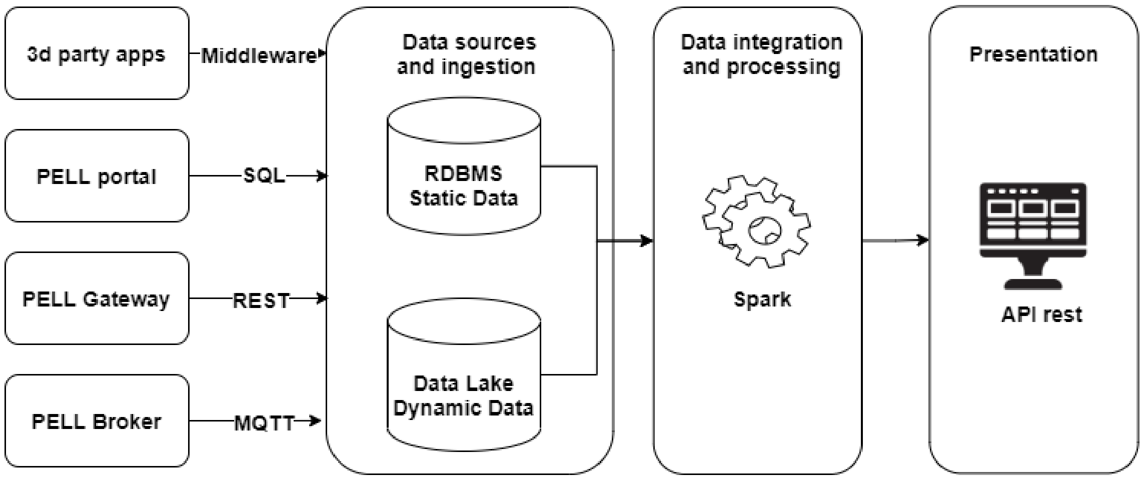


Figure 5.1: Proposed BD processing architecture

5.3.1 Data sources and ingestion layer

Data ingestion is carried out through heterogeneous data sources:

- 3d party apps is a general purpose source typology. Every application developed by partners joining PELL project that interacts with our BD platform through a generic middleware falls into this category.
- PELL Portal is the interface between the PELL project and the stakeholders. It enables utilities to manually send data through a dedicated upload page. Typically such data contains information about the lighting plant topology and technical characteristics. Data is sent to the data lake directly through SQL protocols.
- PELL Gateway is a M2M interface enabling utilities and municipalities to send data through a RESTful APIs. It is used to send automatically the data that would be otherwise sent through the portal.
- PELL Broker is a Message Queue Telemetry Transport (MQTT) protocol enabling sources to stream data to the data lake through the publish/subscribe pattern. This type of source typically sends data retrieved from smart meters, containing electric measures (dynamic data).

Data ingested is persisted according to two main systems: RDBMS (Relational Database Management System) for static data and data lake for dynamic data. We used two different persistence systems because the structure of the static and dynamic data is very different, thus, for optimization reason, a relational database is more suitable for the former, and an unstructured persistence system for the latter.

5.3.2 Data integration and processing layer

The data integration phase is the process where static and dynamic data are merged together in order to carry out KPI evaluation involving information coming from both the domains.

As static data we refer to information concerning the lighting plant system in terms of its topology, the registry municipality, technical characteristics of the entities involved and other context info such as lighting control policies and annual expenses. In particular, the entities involved are:

- POD. Point of delivery where electrical panels are connected. PODs are uniquely identified at national level and they are the devices where the billable consumption is evaluated.

- Electrical panels. Devices that physically deliver electrical energy to the lighting points. Electrical energy absorbed is measured at this level through smart meters.

- Lighting points. Devices providing lighting to the streets. Its structure is usually composed by one or more lighting devices (e.g. lantern, light armor) containing lighting source/s (e.g. halogen, high pressure sodium, low pressure sodium) or LED module/s. Usually, lighting points are installed on a support (e.g. a pole)

These information are defined as *Census Tech Sheet*[5] and structured in XML format. We designed the persistence system of this data as a structured relational database, because

- scalability is not an issue, since the amount of information concerning lighting plants is limited;

- the entities involved in Census Tech Sheet are well defined and strongly related.

[5] `https://www.pell.enea.it/assets/data/download/CensusTechSheet-ImplementationGuide.html`

As dynamic data, we refer to electrical measures collected by smart meters installed into the electrical panels. This data is retrieved through a MQTT broker and represented in the JSON format according to a specific data structure, defined as *Urban Dataset*. Such data structure is aimed to define a format for exchanging data in an interoperable way within smart city platforms. Regarding the domain of the presented work, there are two specific categories of data set in the Urban Dataset: *Counter Reading* and *Counter Reading monophase*. The former is used to describe electric measures collected in three-phase lighting plants while the latter for single-phase ones. Such data requires a great scalability and high efficiency in data retrieving, thus we used an unstructured data lake based on the Apache Hadoop ecosystem to persist them.

Data integration is carried out using Apache Spark library that provides an easy interface with different sources of persistence and potentially many other sources of data. Moreover it guarantees remarkable performances and scalability in data retrieving during the calculation process. In our case, using Spark we manage static data through SQL queries and dynamic data by simply parsing their JSON-based representation format. The resulting aggregated data is collected in *data frames*, the spark main in-memory data structure. Once data frames are obtained, they are merged together into a new data frame containing information required to the next elaboration for the final presentation. The resulting process is the final dataframe containing the output of the process, ready to be served to the presentation layer and third parties.

5.3.3 Data presentation layer

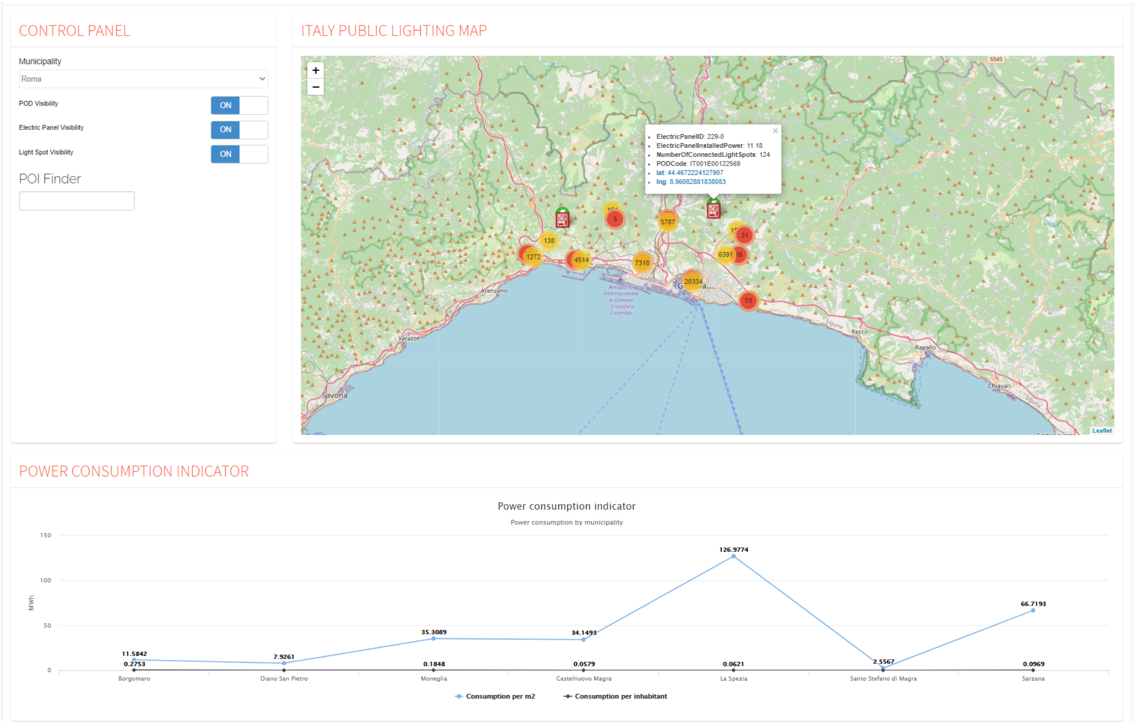
The data presentation layer is the final stage of the process. It refers to represent knowledge/insights, which are usually exposed as services and delivered through intuitively web dashboards. By tracking multiple data sources, this software layer allows monitoring and analysis of KPIs and analytical queries' results. Traditional data visualization tools would exhibit poor performance in functionality, response time, and scalability if adopted for BD. Rethinking how to visualize BD in a different manner is therefore necessary.

In this specific case, the processed data is provided to the presentation layer in the format parquet[6], which is a free and open-source column-oriented data storage format of the Apache Hadoop ecosystem providing efficient data compression and

[6]`https://parquet.apache.org/`

encoding schemes with enhanced performance to handle complex data in bulk. It can be accessed regardless of the choice of the data processing framework, data model or programming language. In our case we use Javascript libraries to connect the data to the web front-end.

Figure 5.2 shows an example of the dashboard provided to the user. In particular the user can visualize time series of electric consumption, navigate lighting plant elements on the map, visualize custom widget representing static or dynamic KPIs and so on. Since every user has a corresponding role and permission, the dashboard is configured such that data is accessed and/or aggregated according to corresponding permissions. A configuration interface has been designed in order to configure different types of views to the users matching their specific needs.


CONTROL PANEL

Municipality: Roma

POD Visibility: ON

Electric Panel Visibility: ON

Light Spot Visibility: ON

POI Finder

ITALY PUBLIC LIGHTING MAP

- ElectricPanelID: 229-0
- ElectricPanelInstalledPower: 11.18
- NumberOfConnectedLightSpots: 124
- PODCode: IT001E00122569
- lat: 44.4672224127907
- lng: 8.96082881838083

POWER CONSUMPTION INDICATOR

Figure 5.2: Dashboard

5.4 Conceptualization examples of KPIs in the PELL SCP

KPIs evaluation is a critical framework of the PELL SCP, providing useful insights about the behavior of the lighting plants in terms of technologies, control policies and usual operation. In the proposed framework, we divide the KPIs according to the context they refer to.

- Static KPIs refer entirely to static data information, e.g. geometric KPI indicates whether the installed power in the lighting points referring to a specific area of the street is compliant to italian Green Public Procurement (GPP) criteria, taking into account the class assigned to the street according to UNI EN 11248.
- Dynamic KPIs refer entirely to dynamic data information, e.g. number of outliers in power consumption per hour.
- Hybrid KPIs refer to both static and dynamic data, e.g. maximum daily consumption deviation between measured energy absorbed by smart meters and the theoretical one according to the operating conditions declared in the lighting plant technical information.

To illustrate how KPIs are computed, we provide here three examples of KPIs about energy consumption deviation (on a daily basis and on a yearly basis) with a special emphasis on the structure of the indicators (data elements, formulas and dimensions).

We first introduce the model elements we adopt for the computation of the indicators. The PELL abstract data model has the following set of nested entities. We denote by

$$PODs = \{pod_1, pod_2, pod_3, \ldots\}$$

the set of all PODs of a lighting plant. We denote by

$$EPs = \{ep_1, ep_2, ep_3, \ldots\}$$

the set of all electrical panels connected to a POD. We denote by

$$LPs = \{lp_1, lp_2, lp_3, \ldots\}$$

the set of lighting points associated with an electric panel. Finally, we denote by

$$LDs = \{ld_1, ld_2, ld_3, \ldots\}$$

the set of lighting devices connected to a light point.

5.4.1 Daily energy consumption deviation (DECD)

It is an indicator that compares the measured daily consumption and the theoretical maximum consumption that would occur if the system operates at maximum power with an operating schedule regulated according to the astronomical timer. Such comparison expresses how much the real consumption differs from the expected one, indicating possible malfunctions or abuses.

The indicator is calculated for each EP associated with a POD as follows:

$$DECD_{pod_j ep_k} = \frac{ADEC_{pod_j ep_k}}{TDEC_{pod_j ep_k}} \tag{5.1}$$

where pod_j is the j_{th} POD and ep_k the k_{th} electrical panel connected to pod_j. The numerator and denominator of the fraction are the actual and theoretical daily Energy Consumption (EC) values calculated as follows.

Actual daily energy consumption: is a consumption at POD and EP level for a selected day and can be measured by the formula:

$$ADEC_{pod_j ep_k} = \sum_{h=1}^{H} \sum_{i=1}^{I} AE^{hi}_{pod_j ep_k} \tag{5.2}$$

where H is the number of observed hours of a selected day, I is the number of time intervals considered for data acquisition in an hour, and $AE^{hi}_{pod_j ep_k}$ is the *active energy* of the selected EP ep_k w.r.t to its associated POD pod_j at specific interval i of the hour h for the selected day.

Theoretical daily energy consumption at maximum power: This consumption is derived from the census tech sheet by considering plants operating at maximum power (100%) and the total operating time that is set by activation and deactivation times of the astronomical timer. First, the terminal *maximum power* (MP) is computed as the sum of the power absorbed by all the connected lighting devices ld_n for $i = 1..N$:

$$MP = \sum_{n=1}^{N} power_{ld_n} \tag{5.3}$$

So:

$$TDEC_{pod_j ep_k} = DT_{ep_k} * MP_{ep_k} \tag{5.4}$$

where DT_{ep_k} is the total operating hours of ep_k for the selected day.

5.4.2 Annual energy consumption deviation (AECD)

This KPI reveals the deviation between the measured annual energy consumption and the expected consumption by considering the plant control policies. We calculate the indicator for each EP associated to a POD as follows:

$$AECD_{pod_i ep_j} = \frac{MEC_{pod_j ep_k}}{EEC_{pod_j ep_k}} \tag{5.5}$$

where pod_j is the j_{th} POD, ep_k the k_{th} electrical panel connected to pod_j, and the annual measured and expected energy consumption values (MEC and EEC, respectively) are calculated as follows.

Annual measured energy consumption: This is a consumption at POD and EP level for the selected year and can be obtained by using the following expression:

$$MEC_{pod_j ep_k} = \sum_{d=1}^{D} \sum_{h=1}^{H} \sum_{i=1}^{I} AE^{dhi}_{pod_j ep_k} \tag{5.6}$$

where D is the total number of days in a year, H the number of operating hours in a day, I is the number of intervals considered for data acquisition in an hour, and $AE^{dhi}_{pod_j ep_k}$ is the active energy of the selected EP ep_k w.r.t to its associated POD pod_j at specific interval i of the hour h for the selected year.

Expected energy consumption: This value is derived from the census tech sheet by considering the power of electrical terminals present in the various homogeneous zones [75] that refer to the POD, the plant operating time, and other regulation control policies if available. We consider two different control policies, *partialized mode consumption* and *dimming mode consumption*, and calculate the energy consumption accordingly. In the case of *Partialized Expected Energy Consumption* (PEEC), we have to turn off 50% of the lighting devices (lamps) of the EP, and others are turned on at 100% power:

$$\begin{gathered} PEEC = (T - PT) * \sum_{n=1}^{N} ld_n + \\ PT * \sum_{n=1}^{N} ld_n * RP/100 \end{gathered} \tag{5.7}$$

where $(T - PT)$ is the terminal operating time window, T is the total operating time of a terminal at maximum power, PT is the terminal operating time when half of the lighting devices (lamps) are turned off, and RP is the power reduction percentage of the terminal. So the measured annual energy consumption deviation by considering the partialized control policy can be computed as:

$$AECD_{pod_iep_j} = \frac{MEC_{pod_jep_k}}{PEEC_{pod_jep_k}} \tag{5.8}$$

In the case of *Dimming Expected Energy Consumption (DEEC)*, the luminous flux is reduced by a percentage on all the lamps:

$$\begin{aligned} DEEC = (T - RT) * \sum_{n=1}^{N} ld_n + \\ RT * \sum_{n=1}^{N} ld_n * RP/100 \end{aligned} \tag{5.9}$$

where $(T - RT)$ the terminal operating time window, RT is the terminal operating time when luminous flux is reduced by a value percentage on all the lamps, and RP is the power average reduction percentage. Hence, in this case:

$$AECD_{pod_jep_k} = \frac{MEC_{pod_jep_k}}{DEEC_{pod_jep_k}} \tag{5.10}$$

5.4.3 Maximum annual energy consumption deviation (MAECD)

It is the deviation between the measured energy consumption and the total one, as per project, assuming that the plant operates at maximum power for the entire operating period. It is calculated as:

$$MAECD_{pod_jep_k} = \frac{MEC_{pod_jep_k}}{TEC_{pod_jep_k}} \tag{5.11}$$

where pod_j is the j_{th} POD, ep_k the k_{th} electrical panel connected to pod_j, the measured energy consumption $MEC_{pod_jep_k}$ is calculated as in formula (6), and the theoretical annual energy consumption value at maximum power $TEC_{pod_jep_k}$ is calculated as the product of the total terminal operating time T_{ep_k} for the selected year with the terminal maximum power, which is the sum of all the power absorbed by all the lighting devices connected to the terminal:

$$TEC_{pod_jep_k} = T_{ep_k} * \sum_{n=1}^{N} power_{ld_n} \tag{5.12}$$

5.5 The PELL SCP pipeline at work

In this section, through a use case we exemplify the proposed pipeline described in section 5.3. First, we describe the data ingestion phase according to the two main persistence systems described previously, then we show how data integration and processing are carried out, and finally we present the results of the elaboration. The running use case is the calculation of the KPI *DECD*, described in subsection 5.4.1.

The test environment has been deployed on a Linux CentOS 7 virtual machine with 4 virtual CPUs and 16GB RAM. For practical testing reasons, we used a local Spark instance, instead of the production clustered version described in Section 5.3. The static data set is stored locally in a MySQL 8 DB, while the dynamic data set is stored locally in JSON files.

5.5.1 Data ingestion

As sources ingested into the persistence systems, static data coming from the PELL portal and JSON data coming from the MQTT broker have been used. Specifically, a sample Census tech sheet describing the use case lighting plant has been defined in XML and manually uploaded to the system through the portal, then a small subset of the electric measures collected by the smart meters of the electrical panel installed in the site R.C. ENEA Casaccia (in Italy), as described in the tech sheet, has been used as a real data set for the use case presented in this paper[7]. The observation period consists of 10 days ranging from the end of September 2020 and the beginning of October 2020.

5.5.2 Data integration and processing

This subsection describes the data integration phase of the static and dynamic data, followed by the actual calculation process of the KPIs.

In order to carry out the data integration, useful data for the specific use case must be extracted from both the two persistence systems: the structured SQL database for static data and the data lake for the JSON-based representation of the collected dynamic data. In particular, we are interested to the following specific values of the census tech sheet:

[7]This data set is available at the github repository: `https://github.com/fabiomor/pell-data`

- *Installed power.* Power installed on each device associated to the lighting point.
- *Operating hours.* Operating hours of the lighting plant considered for the purposes of the calculation depend on the KPI to be calculated. In the case of the daily KPI, the maximum number of operating hours is calculated on ARERA (Regulated authority for energy, networks and environment) resolution, in which for each decade of the year and for each geographical area, the number of conventional hours of switching on and off of a public lighting system is defined.
- *Regulation mode.* There are two types of regulation, midnight-all night partialization and luminous flux reduction. If one flag is true, the ignition hours are indicated in the relative fields, while the power to the terminals is reduced by the percentage indicated in the fileds *power reduction* and *average power reduction.*
- *Power reduction.* Percentage of the power reduction performed on the lighting plant according to regulation mode. It can be a fixed reduction or a dynamic reduction, thus it can be expressed as *power reduction* or *average power reduction.*

All the estimated consumption (at the denominator of the KPIs' formulas) is calculated from the static data of the census sheet by considering the sum of the consumption of each device present to the single POD and electrical panel. It is in practice calculated through a specific query to the database, which is implemented in PySpark. Table 5.1 shows an example of a resulting output dataframe for the consumption estimated from static data.

Table 5.1: Example of dataframe for the estimated consumption through static data

TerminalPower	PODCode	ElectricPanelID	day	hours	minutes	theorDailyEnergyKwh
400.0	IT012345678901	IT012345678901	2020-09-20	11	15	4500.0
400.0	IT012345678901	IT012345678901	2020-09-21	11	50	4733.3
248.0	UVAX	UVAXPANELID	2020-09-20	11	15	2790.0
248.0	UVAX	UVAXPANELID	2020-09-21	11	50	2934.6
538.0	UVAX	UVAXPANELID2	2020-09-20	11	15	6052.5
538.0	UVAX	UVAXPANELID2	2020-09-21	11	50	6366.3
870.0	UVAX2	UVAXPANELID3	2020-09-20	11	15	9787.5
870.0	UVAX2	UVAXPANELID3	2020-09-21	11	50	10295.0

Dynamic data retrieving is carried out importing Counter Reading samples into a dataframe structure using proper Spark libraries. Table 5.2 shows the schema for the dataframe containing the result of the parsing of the JSON-based representation for dynamic data. It includes the column name and datatype (essentially a string with the nullable property set to true to accept null properties) for the electrical data coming from the smart meters, along with the data relating to the electrical panel, POD and start/end reading time.

Table 5.2: Structure of the Counter Reading as dataframe

Data variables	**Data type**
timestamp	string (nullable = true)
start_time	string (nullable = true)
end_time	string (nullable = true)
PODID	string (nullable = true)
ElectricalPanelID	string (nullable = true)
ActiveEnergy	double (nullable = true)
ActivePowerPhase1	string (nullable = true)
ActivePowerPhase2	string (nullable = true)
ActivePowerPhase3	string (nullable = true)
ActivePowerPhase2	string (nullable = true)
ApparentPowerPhase1	string (nullable = true)
ApparentPowerPhase2	string (nullable = true)
ApparentPowerPhase3	string (nullable = true)
CurrentLine1	string (nullable = true)
CurrentLine2	string (nullable = true)
CurrentLine3	string (nullable = true)
PowerFactorPhase1	string (nullable = true)
PowerFactorPhase2	string (nullable = true)
PowerFactorPhase3	string (nullable = true)
ReactiveEnergy	string (nullable = true)
ReactivePowerPhase1	string (nullable = true)
ReactivePowerPhase2	string (nullable = true)
ReactivePowerPhase3	string (nullable = true)
TotalActivePower	string (nullable = true)
TotalApparentPower	string (nullable = true)
TotalReactivePower	string (nullable = true)

Continued on next page

Table 5.2 – continued from previous page

Data variables	Data type
VoltagePhase1	string (nullable = true)
VoltagePhase2	string (nullable = true)
VoltagePhase3	string (nullable = true)

Since the Counter Reading representation format contains context information not relevant for the calculation of the KPIs, a data pre-processing step has been carried out to filter only the data of interest. Table 5.3 shows an example of the resulting dataframe after filtering only the electrical data of interest for the purpose of calculating the KPIs, that are the dates relating to the period analyzed, the active energy value, the POD number and the electrical panel code.

Table 5.3: Example of dataframe for dynamic data

name	start_period	end_period	ActiveEnergy	PODID	ElectricalPanelID	dateFMT
Counter Reading	2020-09-26T00:00:00	2020-09-26T00:15:00	153.13	IT000000000ID2	IT000000000ID2	2020-09-26
Counter Reading	2020-09-26T00:15:00	2020-09-26T00:30:00	153.49	IT000000000ID2	IT000000000ID2	2020-09-26
Counter Reading	2020-09-26T00:30:00	2020-09-26T00:45:00	153.34	IT000000000ID2	IT000000000ID2	2020-09-26
Counter Reading	2020-09-26T00:45:00	2020-09-26T01:00:00	152.4	IT000000000ID2	IT000000000ID2	2020-09-26
Counter Reading	2020-09-26T01:00:00	2020-09-26T01:15:00	152.31	IT000000000ID2	IT000000000ID2	2020-09-26
Counter Reading	2020-09-26T01:15:00	2020-09-26T01:30:00	152.45	IT000000000ID2	IT000000000ID2	2020-09-26
Counter Reading	2020-09-26T01:30:00	2020-09-26T01:45:00	152.96	IT000000000ID2	IT000000000ID2	2020-09-26
Counter Reading	2020-09-26T01:45:00	2020-09-26T02:00:00	153.36	IT000000000ID2	IT000000000ID2	2020-09-26
Counter Reading	2020-09-26T02:00:00	2020-09-26T02:15:00	153.62	IT000000000ID2	IT000000000ID2	2020-09-26

Finally, the sum of the active energy values for the period of interest is carried out, grouping the data by POD and electrical panel. The static and dynamic dataframes, subsequently, are joined together on POD and Electrical Panel, the calculation of the KPI on the static and dynamic data is performed, and a new final dataframe is returned as output. See the example of output dataframe shown in Table 5.4.

In order to give an idea on how the implementation code of these KPIs in PySpark look like, Listing 5.1 shows a fragment of the Python code for calculating the KPI DECD. The function *evaluateDECD* takes as input parameters the date ranges and, optionally, it can be specified a particular POD or EP. Once the information about the selected POD and EP is retrieved, in *df_ static* is evaluated the theoretical energy consumption based on operating hours of each day. Then, in *df_ dynamic*, the sum of the active energy absorbed for each time step is evaluated, grouped by day, POD id and EP id. Finally, the two dataframes are joined together based on day, POD id and EP id in order to obtain all the information needed for the KPI

Table 5.4: Output dataframe for the Daily Energy Consumption Deviation DECD from 09/25/2020 to 10/04/2020

dateFMT	PODCode	ElectricPanelID	DECD
2020-09-25	IT001E04172906	QEID2	1.02
2020-09-25	UVAX	UVAXPANELID	1.05
2020-09-26	IT001E04172906	QEID1	1.33
2020-09-26	IT001E04172906	QEID2	1.02
2020-09-26	UVAX	UVAXPANELID	1.1
2020-09-26	IT000000000ID2	IT000000000ID2	0.86
2020-09-27	UVAX	UVAXPANELID	1.09
2020-09-28	UVAX	UVAXPANELID	1.06
2020-09-29	UVAX	UVAXPANELID	1.06
2020-09-30	UVAX	UVAXPANELID	1.06
2020-10-01	UVAX	UVAXPANELID	0.98
2020-10-02	UVAX	UVAXPANELID	0.97
2020-10-03	UVAX	UVAXPANELID	1.04
2020-10-04	UVAX	UVAXPANELID	1.06

evaluation.

5.5.3 Presentation

The resulting data frame described in the previous subsection is then available to be served to the GUI or dashboards. The general approach, described in section 5.3, is to serve the dataframe as parquet file, which is readable by a wide variety of libraries in the most common languages. In this specific use case, we here show the result using a simple data frame display function (see Fig. 5.3). As shown if the figure, a negative peak occurs during the first days of October. The reason is the change of the decade establishing the window of operating hours: in October the number of operating hours of the lighting plant, where the system is turned on, increase. The increase is not reflected in the actual measurements in first place, but only on the third and forth day of the month.

```
from static_data_manager import StaticDataManager
from dynamic_data_manager import DynamicDataManager
import yaml
import spark_session
import pyspark.sql.functions as psf

def evaluateDECD(self, dateFrom, dateTo, pod = None, qe =None):
        df_static = self.sdm.get_terminal_power(dateFrom, dateTo, pod, qe)
        df_static = df_static.withColumn('theoreticalDailyEnergyKwh',df_static['TerminalPower'] * ((df_static["hours"] ) +
              ↪ df_static["minutes"] /60))

        df_dynamic = self.ddm.load_jsons()
        df_dynamic = df_dynamic.withColumn('dateFMT',psf.to_date(psf.unix_timestamp(psf.lit(df_dynamic["start_period"]),
              ↪ format="yyyy-MM-dd'T'HH:mm:ss").cast("timestamp")))
        df_dynamic = df_dynamic.groupBy('dateFMT','PODID','ElectricalPanelID').agg(psf.sum('ActiveEnergy').alias('
              ↪ ConsumedDailyEnergyKwh'))

        df_merged = df_dynamic.join(df_static, (df_dynamic['PODID'] == df_static['PODCode']) & (df_dynamic['
              ↪ ElectricalPanelID'] == df_static['ElectricPanelID']) & \
        (df_dynamic['dateFMT'] == df_static['day']) ,'inner')
        df_merged = df_merged\
                .withColumn('kpiMaxDailyDeviation',psf.round(df_merged['ConsumedDailyEnergyKwh']/df_merged['
                      ↪ theoreticalDailyEnergyKwh'],2))\
                .drop(*['ConsumedDailyEnergyKwh', 'TerminalPower', 'hours', 'minutes', 'theoreticalDailyEnergyKwh','
                      ↪ SchedeCurrent'])\
                .sort(dateFMT)
        return df_merged
```

Listing 5.1: Example of KPI implementation in PySpark

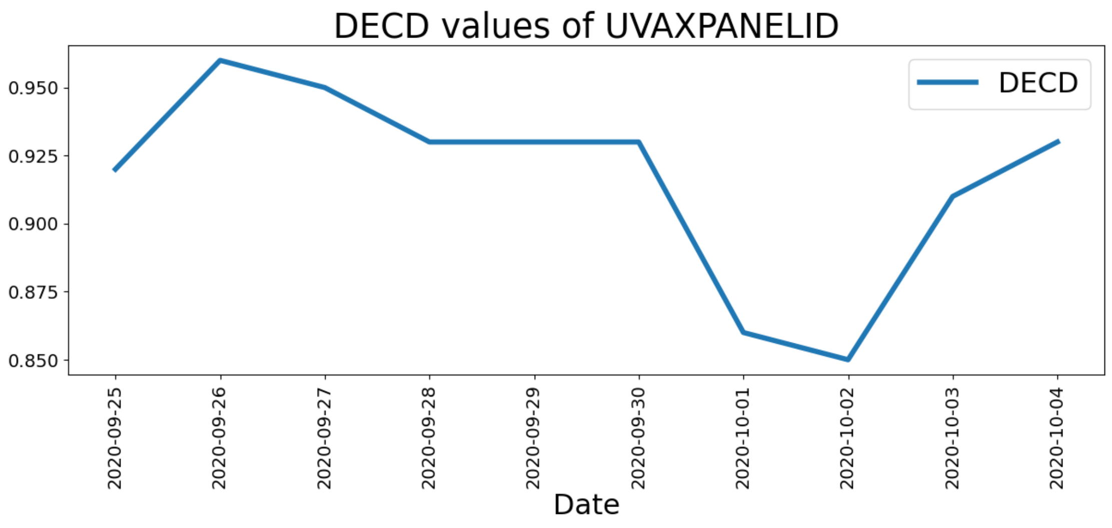


Figure 5.3: Line chart of DECD KPI calculation from 09/25/2020 to 10/04/2020

5.6 Lessons learned, development experience, and future directions

In addition to the domain analysis, the design of the software architecture, and the conceptualization and formulation of the KPIs, the concrete development of the proposed SCP required us a learning process for acquiring a wide spectrum of multi-disciplinary skills. We here try to distill the basic skills which are necessary to acquire by the development team on the base of our practical experience. We do not report details about the amount of human-resources and man-months of working time since we consider such project management issues out of the scope of this work.

From a computer system point of view, it is required to know how to set up the runtime environment according to the characteristics of the hosting platform (local or cloud-based virtual machines, virtual machine clustering, and operative system) and how to install Hadoop, Spark (including a stack library for SQL and data frames, streaming, and complex analytics) and all the other tools and software infrastructures of the Apache ecosystem required to build the proposed BD software architecture. The seamless connection of all these software solutions through a sequence of configuration steps require a fair amount of experience.

From a software developer point of view, in the first place acquiring basic programming skills in Python (or other similar programming languages like Java or Scala supported in Spark) and SQL are required, then the focus should be put on studying and applying the stack of libraries of Spark for large-scale data processing. In order to build a maintainable and scalable project in Python, a knowledge of the Object Oriented Programming (OOP) paradigm and of principles for developing small micro-services is highly suggested.

In the future we want to contribute to the design and development of an analytical framework based on machine learning algorithms that could be incorporated and used, for example, in the PELL SCP for discovering electrical energy consumption patterns, anomalies detection, and for providing energy efficiency strategies through predictive models. We also want to instantiate our Big-Data pipeline architecture for another energy-intensive public infrastructure, that is of public buildings.

5.7 Summary

In this chapter, we examined the main building blocks realizing a scalable and efficient BD software pipeline for processing and managing urban data in the public street lighting domain, which is a substantial part of the smart city concept. We also provided some insights on how we defined and implemented some KPIs for managing energy consumption as part of the BDA framework of the ENEA PELL SCP. It is important to underline that the main goal of these KPIs is to provide a feedback about the actual behavior of the lighting plant in respect of the expected behavior according to the declared information given in the static data. Thus, they are not designed to produce effective energy saving but only to improve the overall knowledge of the lighting plant behavior. This is precisely how smart data is produced in this specific context.

In the next chapter, we will focus on the third objective: devise a predictive analytics based approach for the detection of anomalies from time series of street lighting energy consumption data. The anomaly detection approach is developed as part of the PELL SCP, which is incorporated as an independent module inside the PELL SCP.

Chapter 6

Anomaly Detection in Public Street Lighting Data using Unsupervised Clustering

This chapter presents an anomaly detection approach for public street lighting that is investigated in collaboration with the Italian national research agency ENEA as part of the PELL SCP. This deals with the third objective of this thesis: Devise a predictive analytics based approach for the detection of anomalies from time series street lighting energy consumption data. We start by presenting some definitions of anomalies and outliers. After this, we discuss the problem domain, and then the proposed anomaly detection approach is discussed in great detail. Afterward, we discuss the experimental results to evaluate the effectiveness of the anomaly detection approach.

6.1 Anomalies and outliers

In the light of literature, there is no consent about the distinction of anomalies and outliers. There exist many studies which used two terms anomalies and outliers interchangeably. For example, in [115] anomalies and outliers are discussed as:

"Outliers are also referred to as abnormalities, discordants, deviants, or anomalies in the data mining and statistics literature."

On the other hand, there are some definitions which consider outliers as a generic concept and also include noise in addition to anomalies [116]. Some studies regard outliers as corruption in the data [117]. The most common definition of the anomalies is presented in [118] and is as following:

> *"Anomalies are patterns in data that do not conform to a well defined notion of normal behavior."*

In 1969, Grubbs [119] defined outliers as fallows:

> *"An outlying observation, or "outlier," is one that appears to deviate markedly from other members of the sample in which it occurs."*

Hawkins [120] discuss outliers as fallows:

> *"An outlier is an observation which deviates so much from the other observations as to arouse suspicions that it was generated by a different mechanism."*

6.2 Problem Domain: The PELL SCP

In this work, we proposed an anomaly detection approach for public street lighting. This approach is developed as part of the ENEA PELL SCP [121], and incorporated as an independent module in the PELL SCP as shown in Figure 6.1.

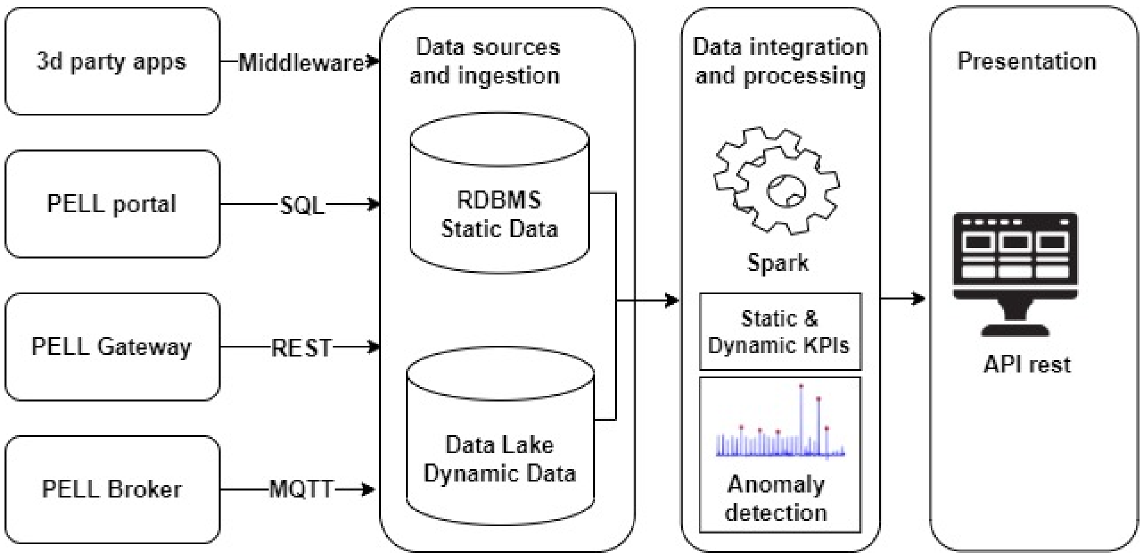


Figure 6.1: PELL SCP with anomaly detection module.

6.2.1 Anomaly Detection in Public Street Lighting

Anomaly detection is an important phase in the process of energy management which has been a research topic for a long time. It can be used to detect variety of issues such as power outage, abnormal consumption or changes in electricity usage. However in a world of digitization, the exponential growth of data has made it impossible to detect the anomalies manually. Hence, one of the most important data analysis task is to automatically detect anomalies in data. Anomalies are those data points which deviate from the normal distribution of the data [122]. In some cases, anomalies are considered as a noise that has a significant impact on the performance of prediction models [123]. The potential causes of the anomalies in the data could be error in the instruments, malicious activities, miscalculation of missing values and human error, therefore it is essential to detect the anomalies to maintain integrity and consistency in the data [124].

Generally, anomalies are divided into three categories such as (1) point anomalies, (2) collective anomalies, and (3) contextual anomalies [125, 122, 126].

1. *Point anomalies.* When a data point is significantly deviates from rest of the data, it could refer to as a point anomaly. It is the simplest type of anomaly and is very common. In the context of electricity consumption, a point anomaly could be that a faulty instrument or meter reports an hour's consumption measure as much higher than it actually is.

2. *Collective anomalies.* When a collection or sequence of data points deviates from rest of the data, it could refer to as a collective anomaly. In contrast to point anomalies that can occur in any data, collective anomalies only exist in the data where the data points are related, spatial data or temporal data [125]. An example of collective anomaly in the context of electricity consumption could be a power outage causing the consumption to drop and remain unavailable for several consecutive hours.

3. *Contextual anomalies.* When a point or sequence of points that may be considered anomalous in a specific context, it could refer to as a contextual anomaly. In other words, such type of anomalies often are identified in time-series and spatial data. Contextual anomalies can be identified by using both contextual and behavioral attributes. In context of electricity consumption, a contextual attribute is time of the year and a behavioral attribute could be the electricity consumption [125].

In this work, we have evaluated the performance of clustering algorithms to detect point anomalies and collective anomalies.

6.3 Related Work

Anomaly detection is also known as outlier detection which means that the problem of finding patterns in data which deviates from the expected normal behavior. It is an important research problem that has immense use in a wide variety of application domains such as intrusion detection, fraud detection, and eco-system monitoring [127]. In the literature, there have been several efforts focusing on the development of ML-based systems for anomaly detection through various supervised, semi-supervised, and unsupervised learning methods. Here, we first discuss in Section 6.3.1 the most notable efforts reported in the literature for anomalies detection in different application domains by using clustering techniques. Then in Section 6.3.2, we focus on methods for anomaly detection in the specific field of electricity consumption data.

6.3.1 Clustering-based techniques for Anomalies Detection

Celik et al. [128] performed a comparative analysis between DBSCAN and a statistical method to discover the anomalies in temperature data. According to the authors, DBSCAN is robust in discovering anomalies as compared to statistical model, because statistical model can only detect the anomalous data points that are below or above the threshold values but it is not able to detect those anomalies that are less frequent and exist in the threshold range.

In [129], three well known clustering algorithms including K-means, DBSCAN, and OPTICS are compared based on the performance measures such as accuracy, outliers formation and cluster size prediction. In the light of authors findings, K-means produced good quality clusters when apply on large scale data, but K-means is sensitive to outliers and also does not perform well with clusters of arbitrary shapes. The highlighted deficiencies of the K-means can be overcome by using DBSCAN, and OPTICS. DBSCAN has an ability to form the clusters of arbitrary shape and size. It also has an ability to determine which data points should be classified as noise point or outliers, in comparison to other clustering models it is very fast algorithm. However, the drawback of DBSCAN algorithm is that when clusters with different densities are located to each other then DBSCAN will not be able to distinguish them. These deficiencies are addressed by the OPTICS. It ensures good quality clustering by giving the priority to high-density clusters over low density clusters [130].

Another DBSCAN based anomaly detection model was proposed by jith et al. [131]. The objective of their work is to detect the anomalies from traffic data set, in which a path is labeled as anomaly if it does not match with the pre-trained model. In another study [132], authors proposed a K-means clustering based approach to detect the anomalies from traffic data set. In [133], authors employed K-means algorithm to detect the anomalies from Call Detail Records (CDR) data set. In order to evaluate the effectiveness of the proposed anomaly detection approach, authors train a neural network model on anomaly and anomaly-free data. During the model training, they observed the effects of anomalous activities and also noted the mean square error of anomaly and anomaly free data. In [134], authors proposed a K-means clustering algorithm based approach to detect the anomalies in software measurement data. In [135], an unsupervised learning based anomaly detection approach from system's log files is presented by using OPTICS algorithm.

In another study [136], authors proposed a K-means based approach to detect the anomalies from heart disease data. To determine the Optimal K values from heart disease data, they used Silhouette method to form the clusters for detecting the anomalies. After forming the clusters, authors used an Interquartile Range (IQR) based formula to detect the anomalies from each cluster. In order to demonstrate the effectiveness of the proposed anomaly detection model, authors evaluated the classification accuracy of various classifiers by using data set with and without of anomalies. In [137], authors proposed another K-means clustering based approach to detect the anomalies from network traffic data. In [122], authors implemented twenty different anomaly detection methods on five univariate time series data sets. These methods are categorized into three categories such as statistical, ML, and DL methods. The evaluation of the algorithms is conducted on publicly available real (Yahoo Services Network traffic, and New York City (NYC) Taxi Dataset) and synthetic benchmark time series data. Authors observed that the models performed differently depending on the anomaly type. Their experimental results demonstrated that statistical and ML methods performed better in detecting point anomalies and collective anomalies. Only in the case of contextual anomalies DL models outperformed the models in comparison. Authors also observed that unsupervised methods Local Outlier Factor (LOF), DBSCAN, isolation forest, and semi-supervised method

One-Class Support Vector Machine (OC-SVM) among the best performing models.

6.3.2 Anomaly Detection in Electricity Consumption Data

In literature, most of the work reported on electricity consumption data analysis consists of forecasting methods. These forecasting approaches are based on statistical, ML and DL models. The studies which deal with the problem of anomaly detection are mostly evaluated on small or synthetic data set due to the unavailability of annotated data [138]. In this section, we discuss the studies for anomaly detection in electricity consumption data.

Sharma and Sing [139] proposed a Density-Based Micro Spatial Clustering of Applications with Noise (DBMSCAN) clustering algorithm. This proposed algorithm is a variation of DBSCAN and applied on electricity consumption data set. In this study, authors managed to identify different types of irregular consumption behaviors such as abnormal peak demands, near zero demand, sudden change of demand, etc. In [140], authors proposed an approach to detect anomalies for power system generation control by using hierarchical density based clustering algorithm. They observed that proposed methods not only detect the anomalies it also has an ability to distinguish between different kinds of anomalies. Mao et al. [141] proposed an unsupervised approach to detect anomalies from power consumption data. The proposed method is based on isolated forest algorithm and has different modules for feature extraction, feature reduction, and isolated forest computing. Authors, claimed that proposed approach is computationally efficient to detect the anomalies from electricity consumption data, and can better meet the needs of the power sector.

In another study, Kedi et al [142]. proposed a density-based electricity theft detection approach to detect abnormal electricity consumption patterns via smart meter data. For experimental evaluation, authors created synthetic data set using six abnormal load profiles. In order to demonstrate the effectiveness of the proposed approach, a comparison with three unsupervised learning techniques including, K-means, DBSCAN, and GMM are conducted. Results show that proposed density-based approach outperformed other models and can precisely detect the electricity theft based on abnormal profiles. In [143], power data is processed by using mutual applications of KNN and K-means to minimize the number of measurement samples. Authors utilized seven different algorithms to detect abnormal consumption patterns. Then on the basis of consumption patterns, normal customers and malicious customers are identified. In [144], authors introduced a concept of collective anomaly, instead of a single event that refer to an anomaly. This collective anomaly is itemsets of events, which may be an anomalous behavior depending on their pattern of appearance. In order to detect the anomalous behavior, authors utilized frequent itemset mining and categorical clustering with clustering silhouette thresholding approaches on smart meters data streams.

In [145] authors investigate the performance of four classification techniques including SVM, LR, KNN, and MLP to detect the anomalies in power generation plant. The results of this study demonstrate that MLP outperformed other classifiers with 96% accuracy. In [146], authors proposed a genetic SVM model to detect abnormal consumption data and suspicious customers. In this study, authors combined genetic algorithm with SVM. In another study [147], authors combined SVM and particle swarm optimization for detecting abnormal power consumption in advanced metering infrastructures. In [148], authors proposed a decision tree based approach to analyse the energy consumption patterns and detect potentially fraudulent activity. In [149], authors proposed a decision tree regressor to detect abnormal power consumption using sensor data, while in [150] an improved decision tree model is presented to detect anomalous consumption data by making use of densities of the anomaly and normal classes. In [151], authors fused autoencoder and Long Short-Term Memory (LSTM) Neural Network (NN) to detect abnormalities in power consumption data set. In [152], Convolutional Neural Netwrok (CNN) and random forest are merged to track energy consumption anomalies due to energy theft attacks. This approach, helps the energy providers to cope with the issues related to irregular energy usage and inefficient electricity inspection. In [153], authors opted for combining Restricted Boltzmann Machine (RBM) along with a Deep Belief Networ (DBN) to develop a DNN-based abnormality detection framework. In [130], authors proposed an approach to detect the anomalies from IoT smart meter data. The objective of their study was to identify high consumption trends in domestic energy consumption. They employed density-based clustering algorithms i.e. DBSCAN, OPTICS, and LOF to identify clusters as normal consumption behavior and noise points as anomalous data points or outliers. The experimental evaluation is performed on the data without applying pre-processing steps and any feature extraction technique. In experimental analysis, authors observed that 53 anomalies identified by DBSCAN, 208 was detected by OPTICS, and 218 anomalies were caught by the LOF.

Based on the literature findings, combined with the promising results achieved in different application domains especially on electricity consumption data, the unsupervised ML-based methods K-means, DBSCAN and OPTICS are investigated in this study. To the best of author knowledge, the proposed approach is the first study in which clustering methods are investigated for anomaly detection on real street lighting data sets. The proposed approach is developed as part of the PELL SCP. In this work, we identified some anomalous scenarios, which are specific to the street lighting domain. Based on these anomalous scenarios, synthetic anomalies (point anomalies and collective anomalies) are introduced into the original electricity consumption data. These synthetic anomalies are injected by using the application of proposed novel algorithms. The performance of the clustering algorithms is measured on the basis of the detection of synthetic anomalies. This is a generic approach and can be applied on any street lighting data to detect anomalies.

6.4 Overview of the Anomaly Detection Approach

This section provides an overview of the overall approach we adopted for detecting anomalies in the PELL street lighting data. As discussed earlier, the proposed anomaly detection system is being developed as a part of the PELL SCP [121] (as already shown in Figure 6.1).

As shown in Figure 6.2, the proposed approach consists of three main stages: *data preprocessing*, *clustering model formulation*, and *model evaluation*. All these stages are discussed separately in the following subsections.

6.4.1 Data Preprocessing

This module is responsible for preparing the data for further processing and analysis. It includes three subsequent phases: data transformation, missing data treatment, and synthetic anomalies injection.

1. *Data Transformation:* PELL street lighting data refer to electrical measures collected by smart meters installed into the electrical panels. This data is retrieved through a MQTT broker and represented in the JSON format. In order to process the data for further analysis, it should be transformed into a usable format. Therefore, we transform the electrical measures available in the JSON files into a processable dataframe, and then perform the remaining

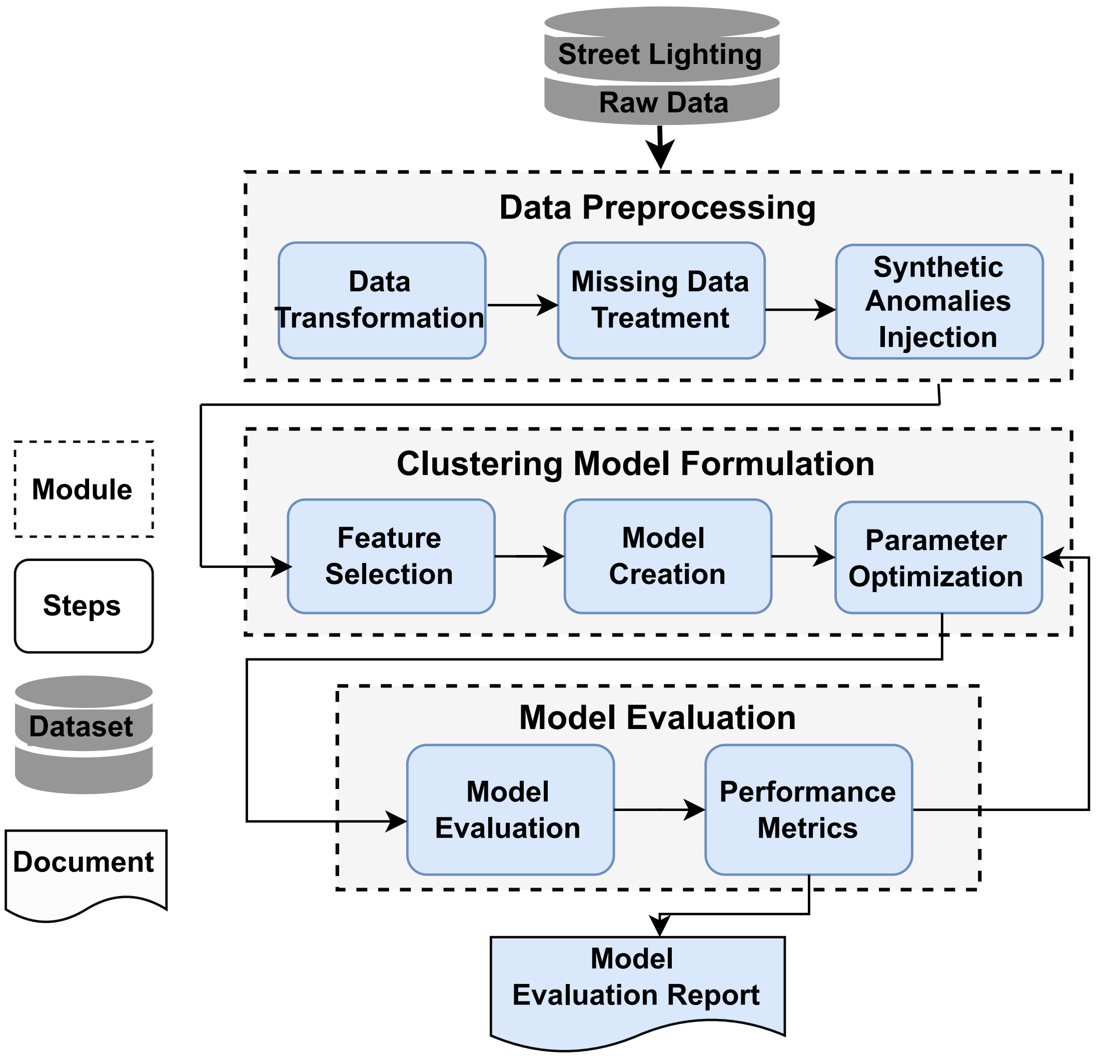
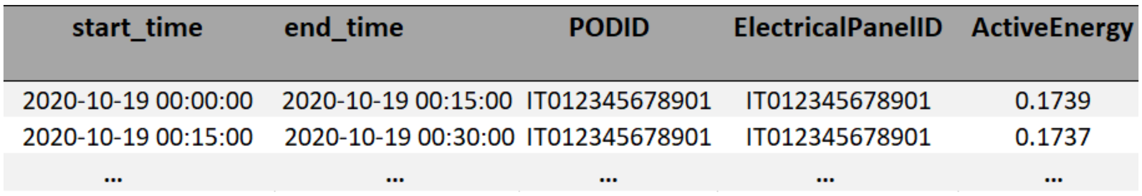


Figure 6.2: Overview of the proposed anomaly detection approach

preprocessing steps to prepare the data for the clustering algorithms. The dataset after preprocessing steps looks as shows in Figure 6.3. The PELL street lighting dataset contains the following fields.

- start_time: Time stamp indicating the start of the period.
- end_time: Time stamp indicating the end of the period.
- PODID: Representing the unique point of delivery.
- ElectricalPanelID: Denotes the unique electric panel.
- ActiveEnergy: representing the amount of energy consumed in a specific time interval.



start_time	end_time	PODID	ElectricalPanelID	ActiveEnergy
2020-10-19 00:00:00	2020-10-19 00:15:00	IT012345678901	IT012345678901	0.1739
2020-10-19 00:15:00	2020-10-19 00:30:00	IT012345678901	IT012345678901	0.1737
...	...	...	...	...

Figure 6.3: Dataset after preprocessing

2. *Missing Data Treatment:* PELL data pre-processing includes dealing with the irregularities in the data records in the form of missing data fields. Two well known approaches to deal with the missing data are the method *moving window* and *inference-based* methods [154]. The moving window method is an improved model of linear interpolation that is easy to implement particularly in the case when the duration of the missing value is brief. Instead, if the duration of missing value is long, then inference-based methods are highly recommended for analysis [155]. In this work, to deal with the missing data, we combined two different strategies: (1) when the number of missing data is less than or equal to 3 as adopted in [155] the moving window method is applied, (2) otherwise, we filled the missing data with the preceding value (the data registered at the same time the day before).

3. *Synthetic Anomalies Injection:* This phase is for evaluation purposes only. In order to evaluate the correctness of the clustering algorithms for the anomaly detection problem in street lighting data, we have introduced artificial anomalies in the original electricity consumption data set to be used as ground truth data. We propose six anomalous scenarios related to the public street lighting domain, and the anomalies are injected by following these anomalous scenarios. More details are given in Section 6.6.2.1.

6.4.2 Clustering Model Formulation

After the application of the pre-processing module, processed data is forwarded for clustering and anomaly detection phase. This module is responsible for selecting features and for building the clustering models.

1. *Feature Selection:* Feature selection is an important step to build the ML models. In this study, we select three features to cluster the PELL electricity consumption data and detect anomalies from it. The selected features are:

measuring time intervals `start_time` and `end_time` of the collected energy measures, and `active energy`, representing the amount of energy consumed in a specific time interval. The active energy is expressed in kilowatt-hours (kWh).

2. *Model Creation* To detect the anomalies by using selected features, we use three well known clustering algorithms, namely K-measn, DBSCAN, and OPTICS. We considered K-means because it is a simple and extensively used unsupervised algorithm, also for big data sets. We selected DBSCAN because of its ability to automatically identify the number of clusters, detect clusters with arbitrary shape, size and most importantly it can automatically detect the outliers available in the data. OPTICS is a generalized version of DBSCAN; we selected it to identify the clusters with varying density in which DBSCAN fails. More details on the configuration and use of these clustering algorithms are presented in Section 6.5.

3. *Parameter Optimization* The performance of clustering algorithms is highly affected by their input parameter settings. For example, K-means is essentially a partitioning method in which a data set of elements is partitioned into a fixed number k of clusters, where k is known a priori. Determining the optimal value of k into which the data may be clustered to perform the K-means clustering is a fundamental step. Similarly, the DBSCAN algorithm requires additional user-defined parameters, such as the optimal neighbourhood radius known as epsilon *(eps)* value, and the *MinPts* parameter, which is the minimum number of points within a group to be considered a cluster. For OPTICS, users have to specify the *eps* value and the cluster method to form clusters. Therefore, in this work we have performed an extensive analysis for tuning the clustering algorithms by using different parameter settings and methods for estimating them. We discuss them one by one in Section 6.5, while in Section 6.6.3, we report our findings with different parameter settings for the task analysis of anomaly detection.

6.4.3 Model Evaluation and Performance Metrics

In this phase we carry out the evaluation of the clustering algorithms for anomaly detection for the PELL electricity consumption data.

In order to evaluate the accuracy of the selected clustering algorithms, in this analysis we have used standard evaluation metrics: Precision, Recall, and F1-measure. These are the standard metrics adopted in information retrieval [156] and also used to evaluate the ML techniques for prediction analysis [157]. More details about our evaluation of the clustering models by using these metrics are provided in Section 6.6. In addition to the accuracy of the clustering models, we have also considered the computation cost by calculating the execution time of the algorithms using different hardware configurations in Apache Spark.

6.5 Adopted Clustering Algorithms

This section introduces the clustering methods used as part of the proposed approach to anomaly detection of the PELL street lighting data.

6.5.1 K-Means Algorithm

k-means [158] is one of the best-known and widely used methods in the literature for its efficiency and simplicity. The primary goal of k-means is identifying k subgroups in the data such that data points in the same subgroup (called *cluster*) are very similar, while data points in different subgroups are very different. The similarity among the data points can be measured by using different distance methods, such as euclidean-based distance or correlation-based distance [159, 160]. The decision of which similarity measure to use is application-specific; in our research we adopted euclidean-based distance measure. K-means takes the number of clusters k and the data set as input and returns a cluster partitioning. It adopts an iterative process, where each data point is assigned to one of the k clusters based on its similarity to the other data points of a cluster.

The outcome of k-means highly depends on the chosen value for the number of clusters k. In this work, to determine the optimal value of k, we used the well known *Silhouette analysis* [161] for a range of values of k (say 1 to 10). This method quantitatively evaluates the data fitness on existing clusters by measuring how similar a data point is to the cluster it belongs to (cohesion) compared to other clusters (separation). For a selected value of k, the silhouette value for a particular data point i of a cluster is calculated by using the formula:

$$S(i) = \frac{b(i) - a(i)}{max(a(i), b(i))} \tag{6.1}$$

where $a(i)$ is the average distance between i and all the other data points within the cluster, and $b(i)$ is the average distance from i to all the other data points of clusters to which i does not belong. Note that the Silhouette value varies from -1 to +1, where the best score +1 means the data point is very compact within the cluster and far away from the other clusters, while the worst score is -1. Values near 0 denote overlapping clusters. The average Silhouette is then calculated for every k as the mean of $s(i)$ over all data of the entire data set. The optimal value of k is then determined choosing the value of k for which the average Silhouette is maximized (the so called *silhouette coefficient*).

After determining k, the main partitioning task of the k-means algorithm is carried out to partition a given data set into k clusters. The steps can be summarized as follows: 1) choose randomly k data points from the data set as initial cluster's centroid or center; 2) assign each data point to the cluster with the closest centroid; 3) re-calculate the centroid of clusters by finding the mean value of the new cluster formed in step 2); 4) repeat steps 2) and 3) until convergence has been reached (no further changes) or a maximum number of iterations (when established) is reached.

After clustering through k-means the PELL street lighting data set, we analyzed the distance between the data points and their clusters's centroids. A huge distance may denote an anomaly data point in the cluster, as discussed in [136]. To this purpose, as proposed in [136], in our work we calculate a so called *Anomaly score* for every data point of the data set as follows:

$$A(i) = \frac{distance(i - C_i)}{L_{C_i}} \tag{6.2}$$

where distance$(i - C_i)$ is the distance between an instance i to its cluster center C_i, and L_{C_i} is the average distance of that cluster. Essentially, the Anomaly score is calculated as the ratio of each data point's distance from the cluster center and the average distance of that cluster. After this, we calculate the following anomaly thresholds based on the research in [162]:

$$Minimum\ threshold\ (MinT) = Q1 - 1.5 * \text{IQR} \tag{6.3}$$

$$Maximum\ threshold\ (MaxT) = Q3 + 1.5 * \text{IQR} \tag{6.4}$$

$$Interquartile\ range\ (\text{IQR}) = Q3 - Q1 \tag{6.5}$$

where $Q1$ is the 25th percentile of the data, and $Q3$ is the 75th percentile of the data.

6.5.2 Density-Based Spatial Clustering of Applications with Noise (DBSCAN)

DBSCAN [163] is another of the most common clustering algorithms. The basic idea behind the algorithm DBSCAN is to form clusters based on the density of data points instead of using the location of the centroid as in K-means [163]. DBSCAN requires two user-defined parameters: the minimum number of points (the threshold *MinPts*) clustered together for a region to be considered dense, and the neighborhood radius *eps* (ϵ), a distance measure used to locate the points in the neighborhood of any point. The steps for DBSCAN clustering can be summarized as follows: 1) the algorithm arbitrarily picks up a not yet visited point i in the data set (until all points have been visited); 2) if there are at least *MinPts* points within a radius of *eps* to the point i then all these points are considered to be part of the same cluster; 3) the clusters are then expanded by recursively repeating the neighborhood calculation for each neighboring point. Formally, the total number of data points within the neighborhood radius of a data point i is calculated by the following formula:

$$N\epsilon(i) = \{j \in D | dist(i, j) \leq \epsilon\} \tag{6.6}$$

where D is the data set, *dist* is an appropriate distance measure for the data set, and $\epsilon \in \mathbb{R}+$.

After clustering, DBSCAN identifies three types of data points: core points, border points, and noise points (outliers). Any data point i in the data set is called core point if $|N\epsilon(i)| \geq$ *MinPts*. A data point i is a border point if it is located at the edge of the dense cluster and the number of its neighbours is $<$ *MinPts*, but it is within the radius of some core point. Finally, a point i is labeled as a noise point or anomalous point if it is neither a core nor a border point [163]. All the data points available in the data set will create clusters with other data points that are reachable within the radius *eps*. Hence, two data points i and j are directly density reachable if the point i is a core point and the point j is within the radius *eps*.

For DBSCAN, the parameters *eps* and *minPts* need to be estimated. Some heuristics for choosing appropriate DBSCAN parameters are provided later in the text, in the evaluation Section 6.6.3.

6.5.3 OPTICS: Ordering Points to Identify the Clustering Structure

Ordering points to identify the clustering structure (OPTICS) [164] is another algorithm for finding density-based clusters. It is similar to DBSCAN, but it solves the DBSCAN issue of not being able to cluster data points of varying density. The key idea is to process the higher density data points first. Essentially, the data points of the data set are (linearly) ordered such that spatially closest points become neighbors in the ordering.

Similarly to DBSCAN, OPTICS needs two parameters: *eps*, which describes the maximum distance (radius) to consider, and *MinPts*, describing the number of points required to form a cluster.

In OPTICS, two additional parameters are also introduced: the *core distance*, which describes the distance to the *MinPts*th closest point, and the *reachability distance*, which is either the maximum of the distance between the two points or the core distance [165]. Formally, the core distance is defined as:

$$cd_{\epsilon,minPts}(j) := \begin{cases} \text{UNDEFINED} \quad if|\{i|d(i,j) \leq \\ \epsilon_m ax\}| < minPts \\ minPts - dist(j) \quad otherwise \end{cases} \tag{6.7}$$

where ***minPts-dist*** is the distance to the minPts nearest neighbor. Reachability distance represents the maximum distance between the point p and core point q within the radius *eps*, such that the reachability cannot be smaller than the core distance. The reachability distance of a point i from j is calculated by the following formula:

$$reachability(i \leftarrow j) := max\{dist(j,i), minPts - dist(j)\} \tag{6.8}$$

Both core distance and reachability distance are undefined if no sufficiently dense cluster (w.r.t. *eps*) is available. The *eps* parameter is therefore used to cut off the density of clusters that are no longer interesting. Hence, in contrast to DBSCAN, in OPTICS the *eps* parameter is not strictly necessary; it can simply be set to the maximum possible value to reduce the run-time complexity of the algorithm. More details on parameter estimation are presented later in the text, in the evaluation Section 6.6.3.

6.6 Experimental Evaluation

In this section we report on the empirical evaluation of the proposed anomaly detection approach on the PELL street lighting data. First, we introduce our research questions (Section 6.6.1). We present the design of the evaluation (Section 6.6.2), including a description of the PELL benchmark datasets, the synthetic anomaly injection, and the metrics we adopted to evaluate the performance of the clustering algorithms. Finally, we present the major results (Section 6.6.3) of our empirical evaluation.

To allow replication and verification of our study, a replication package (including the source code, and dataset with testing material) is publicly available as GitHub repository[1].

6.6.1 Research Questions

The purpose of our empirical evaluation is to study the extent to which the proposed configured clustering algorithms can detect anomalies, and the effectiveness and execution cost of the detection. In particular, we aim at answering the following three research questions:

- **RQ1**: What is the performance of the identified clustering algorithms for anomaly detection in the PELL street lighting data?
- **RQ2**: What is the cost of executing the clustering algorithms depending on their parameters setting and number of anomalies to detect?
- **RQ3**: What is the execution cost of the clustering algorithms and their ability to maintain effectiveness when running distributed at scale, i.e. when Spark runs on a cluster?

[1]`https://github.com/MubashirAliCheema/Anomaly-Detection`

6.6.2 Design of the Evaluation

To detect the anomalies from time series energy consumption data, in this work, we have used PELL street lighting data as discussed in Section 6.2. It is composed of electrical energy consumption measures collected by smart meters installed into the electrical panels located at Casaccia, Rome, Italy. We have used electric consumption measures which are collected from July 2020 to December 2020. This consumption is measured with the granularity of fifteen minutes, and each JSON file contains 96 values (4 values per hour and per 24 hours of a day). Figure 6.4 shows an example of electricity consumption series.

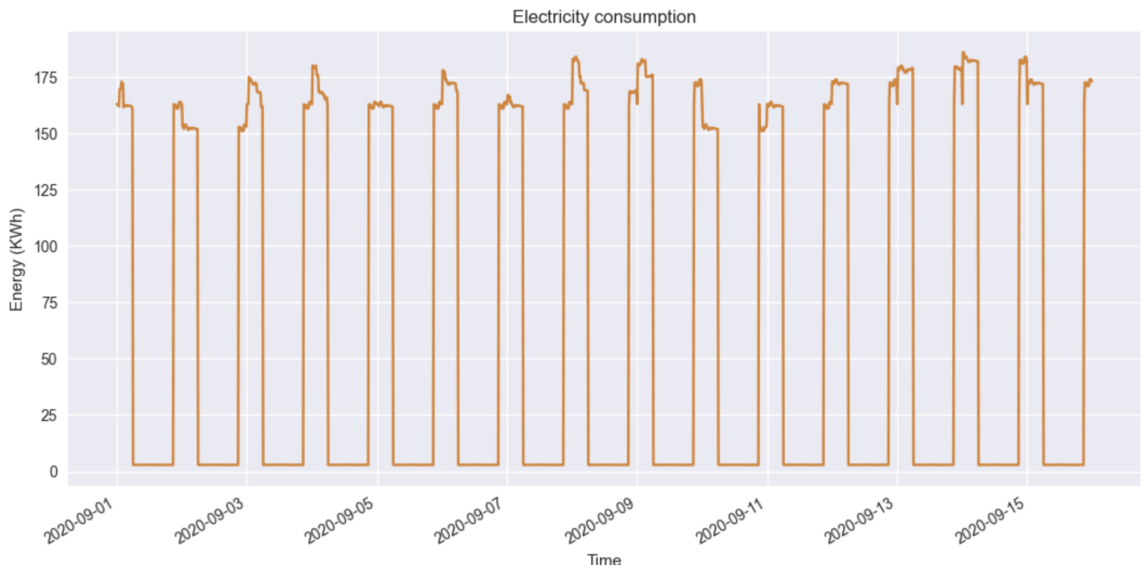
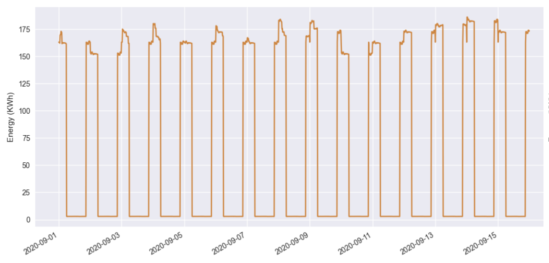


Figure 6.4: An example of electricity consumption series

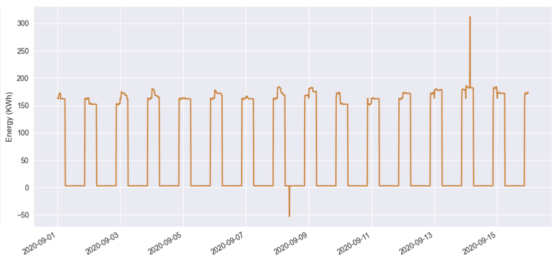
6.6.2.1 Artificial Anomalies

Since the models used in this study are unsupervised, no labeled data is required for training. However, in order to statistically evaluate the clustering algorithms, synthetic anomalies are inserted in the original data. As discussed in Section 6.2, anomalies could be point anomalies, collective anomalies, and contextual anomalies. In this study, point anomalies and collective anomalies are inserted manually in the original data. Figure 6.5 shows electricity consumption series before and after the synthetic anomalies.

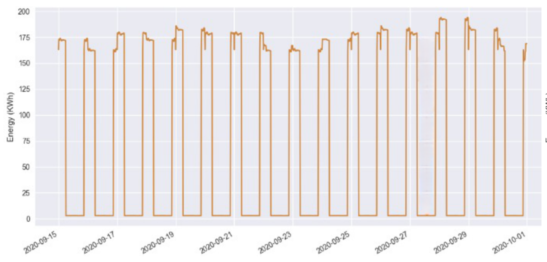
These synthetic anomalies are introduced by following the domain specific scenarios which are as follows:



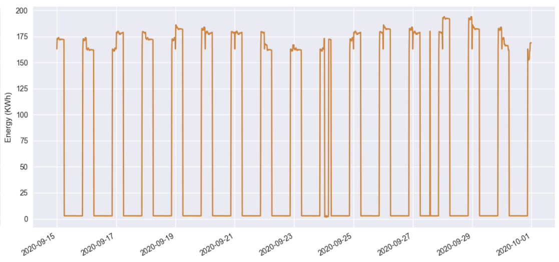
(a) Before point anomalies



(b) After point anomalies



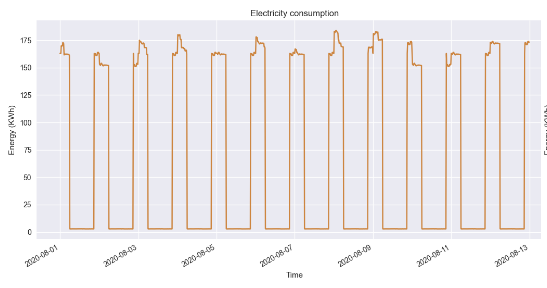
(c) Before collective anomalies



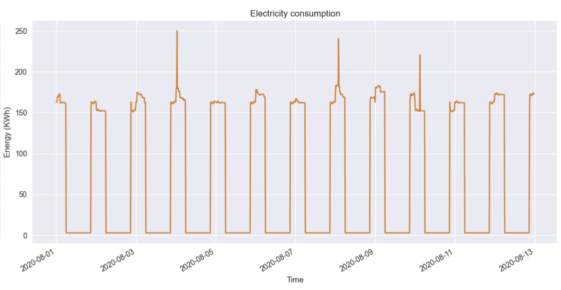
(d) After collective anomalies

Figure 6.5: Example of consumption series before and after the synthetic anomalies

- **Scenario 1 (S1)**: To handle random positive high peaks at the night time. For example, a terminal operating at the night time and some random high peaks encountered, which deviates from the normal consumption measures. An example of electricity consumption series of this scenario is shown in Figure 6.6.



(a) Before S1 anomalies

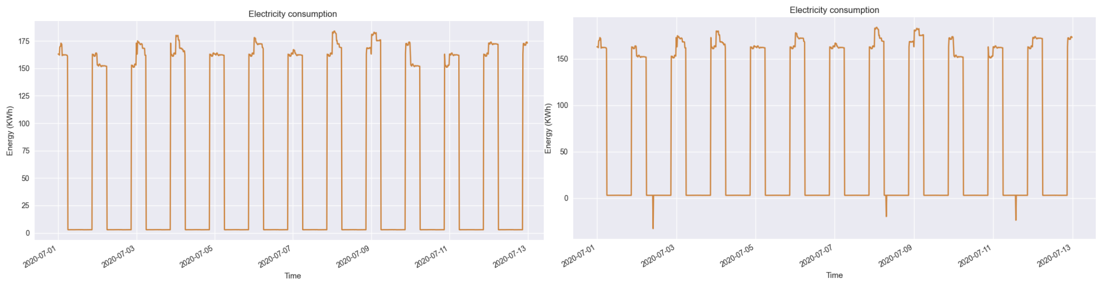


(b) After S1 anomalies

Figure 6.6: Example of electricity consumption series before and after S1 anomalies

- **Scenario 2 (S2)**: To handle random negative high peaks at the day time. For example, a terminal is switched off during the day time and some random negative peaks encountered. An example of electricity consumption series of this scenario is shown in Figure 6.7.

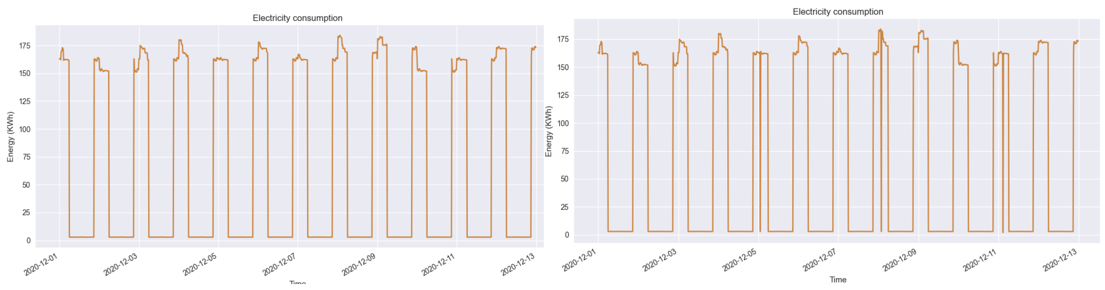
- **Scenario 3 (S3)**: To handle random low peaks at the night time. For ex-



(a) Before S2 anomalies **(b)** After S2 anomalies

Figure 6.7: Example of electricity consumption series before and after S2 anomalies

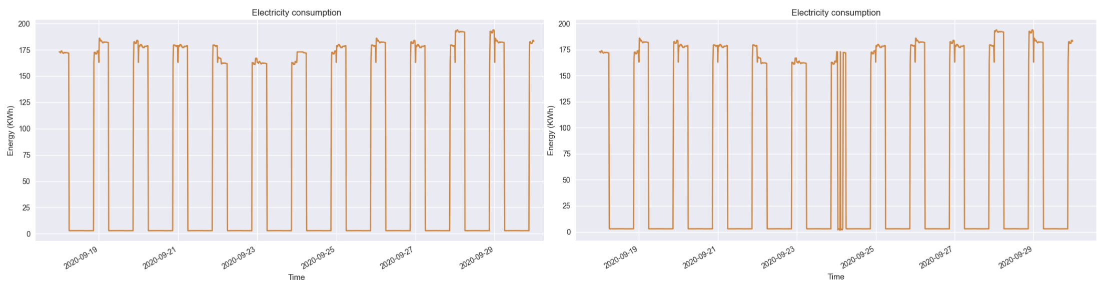
ample, a terminal is operating at the night and some random low peaks encountered, which are different from the normal consumption measures. An example of electricity consumption series of this scenario is shown in Figure 6.8.



(a) Before S3 anomalies **(b)** After S3 anomalies

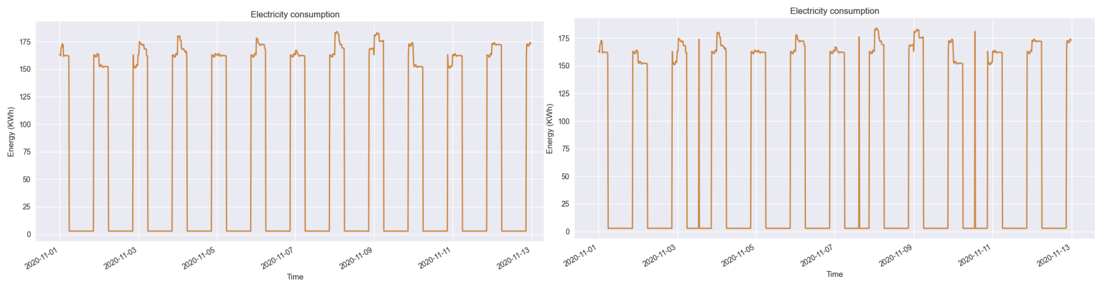
Figure 6.8: Example of electricity consumption series before and after S3 anomalies

- **Scenario 4 (S4)**: To handle continuous low peaks at the night. For example, a terminal is switched off for a duration of three to four hours at the night time. An example of electricity consumption series of this scenario is depicted in Figure 6.9.
- **Scenario 5 (S5)**: To handle random high peaks at the day time. For example, a terminal was switched off at the day time and some random high peaks encountered, which deviates from the normal consumption measures at the day time. An example of electricity consumption series of this scenario is presented in Figure 6.10.
- **Scenario 6 (S6)**: To handle continuous high peaks at the day time. For example, a terminal switched on for a duration of three to four hours at day



(a) Before S4 anomalies **(b)** After S4 anomalies

Figure 6.9: Example of electricity consumption series before and after S4 anomalies



(a) Before S5 anomalies **(b)** After S5 anomalies

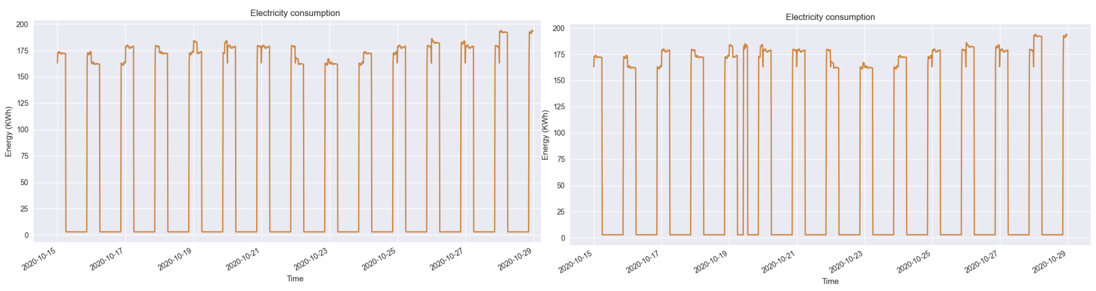
Figure 6.10: Example of electricity consumption series before and after S5 anomalies

time. An example of electricity consumption series of this scenario is shown in Figure 6.11.

6.6.2.2 Automatic Injection of Synthetic Anomalies

In this section, we have discussed the proposed algorithms that are used to automate the process of synthetic anomaly injection in the PELL street lighting data. **Algorithm 1** is proposed to automate the process of artificial anomaly injection for scenarios S1, S2, S3 and S5.

In the algorithm, at step 1, it is confirmed that the number of synthetic anomalies should be less than the total number of samples available in the data set. Then at step 3, random values are generated by following the minimum range value and maximum range value which will be used as synthetic anomalies. At step 6, $isNight(e.start\_ts)$ function determines the night operating hours of the electric panel based on the start time of the electric panel. Afterward at step 8, random indices equals to the number of synthetic anomalies are selected from the data set and



(a) Before S6 anomalies **(b)** After S6 anomalies

Figure 6.11: Example of electricity consumption series before and after S6 anomalies

at step 10, synthetic anomalies are replaced against the selected indices of ActiveEnergy column. Finally, at step 12, data set with synthetic anomalies is returned.

The **Algorithm 2** is proposed to automate the process of artificial anomaly injection for scenario S4 and S6.

In the algorithm 2, at step 1, it is confirmed that the number of synthetic anomalies should be less than the total number of samples available in the data set. Then at step 3, random values are generated by following the minimum range value and maximum range value which will be used as synthetic anomalies. At step 6, $isNight(e.start\_ts)$ function determines the night operating hours of the electric panel based on the start time of the electric panel. In order to ensure the indices range of continuous synthetic anomalies should not exceed the length of the data set, at step 8, we determine the maximum index for synthetic anomalies. At step 9, random index is selected from the data set by considering the minimum index value of the data set and maximum index value selected at step 8. Afterward, at step 10, random consecutive indices equals to the number of synthetic anomalies are selected and at step 12, synthetic anomalies are replaced against the selected indices of ActiveEnergy column. Finally, at step 14, data set with synthetic anomalies is returned.

6.6.2.3 Evaluation Metrics

To evaluate how the chosen clustering algorithms perform, we used three well known metrics that are *Precision*, *Recall*, and *F1-Score*. These are the standard metrics adopted in information retrieval [156]. These metrics are also used for prediction systems based on ML techniques [157].

Algorithm 1 Synthetic anomaly injection algorithm for PELL street lighting data (S1, S2, S3, & S5)

Input:
D: a data frame (two-dimensional array with heterogeneous data) containing n instances
N: numbers of synthetic anomalies to be injected
minR $\in \mathbb{R}$: minimum value for synthetic anomaly (SA)
maxR $\in \mathbb{R}$: maximum value for SA
Output: D' = D with N synthetic anomalies
Method:

```
 1: if $N < len(D)$ then
 2:     for i = 1 to N do
 3:         list_of_anomalies.append(random(minR, maxR))
 4:     end for
 5:     for each $e \in$ D do
 6:         $D' \leftarrow$ D - {e $\in$ D | $isNight(e.start\_ts)$}
 7:     end for
 8:     random_indices $\leftarrow$ sample($D'$, N)
 9:     for $i$ in random_indices do
10:         $D'$[i].ActiveEnergy $\leftarrow$ list_of_anomalies[i]
11:     end for
12:     return $D'$
13: else
14:     print("Reduce N number of synthetic anomalies")
15: end if
```

These metrics are usually adopted in ML to measure algorithms performance in classification problems. In our case, since in our experimentation we are going to test the performances of the clustering algorithm to detect anomalies in a scenario where we have a-priori information about the artificial anomalies manually introduced, we can actually define the task as a classification problem. So we adopted these metrics to have a significant comparison value among the anomaly detection algorithms we tested.

Precision determines the percentage of all detected anomalies which were true anomalies. It is computed as follows:

$$Precision = \frac{TP}{TP + FP} \tag{6.9}$$

Recall determines the percentage of all true anomalies which were detected by the system. It is computed as follows:

Algorithm 2 Synthetic anomaly injection algorithm for PELL street lighting data S4 & S6

Input:
D: a data frame (two-dimensional array with heterogeneous data) containing n instances
N: numbers of synthetic anomalies to be injected
minR $\in \mathbb{R}$: minimum value for synthetic anomaly (SA)
maxR $\in \mathbb{R}$: maximum value for SA
Output: D' = D with N synthetic anomalies
Method:

```
 1: if $N < len(D)$ then
 2:     for i = 1 to N do
 3:         list_of_anomalies.append(random(minR, maxR))
 4:     end for
 5:     for each $e \in$ D do
 6:         $D' \leftarrow$ D - {e ∈ D | $isNight(e.start\_ts)$}
 7:     end for
 8:     maxIndex $\leftarrow len(D')$ - N
 9:     random_index $\leftarrow$ random(0 , maxIndex)
10:     random_consecutive_indices $\leftarrow$ random_index + N
11:     for $i$ in random_consecutive_indices do
12:         $D'$[i].ActiveEnergy $\leftarrow$ list_of_anomalies[i]
13:     end for
14:     return $D'$
15: else
16:     print("Reduce N number of synthetic anomalies")
17: end if
```

$$Recall = \frac{TP}{TP + FN} \tag{6.10}$$

where true positive (TP) represents the number of predicted anomalies which were real anomalies, false positive (FP) represents the number of predicted anomalies which were not real anomalies, false negative (FN) represents the predicted normal windows which were anomalies.

Both metrics are combined in a single score, called *F1 score*, that gives both metrics an equal importance. F1-Score is the harmonic mean of Recall and Precision, and is computed as follows:

$$F1\text{-}Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{6.11}$$

6.6.3 Results

In this section we present experimental results of clustering algorithms with their exploratory settings. We developed the code for all the clustering algorithms in the Python programming language by exploiting scikit-learn[2] library.

6.6.3.1 Answers to RQ1 (Clustering Performance)

In order to evaluate the used clustering algorithms on the experimental data set, optimal parameter selection is performed for each algorithm. The attained results with different parameter settings are discussed here.

1. *K-means:* In this study, silhouette score is used to determine the optimum value of K. It is one of the most efficient and popular measures to determine the clustering quality. The Silhouette plot is very robust as it demonstrate how much data points are similar within one cluster as compared to the points in other cluster [166]. We plotted the K vs. silhouette score graph as shown in Figure 6.12.

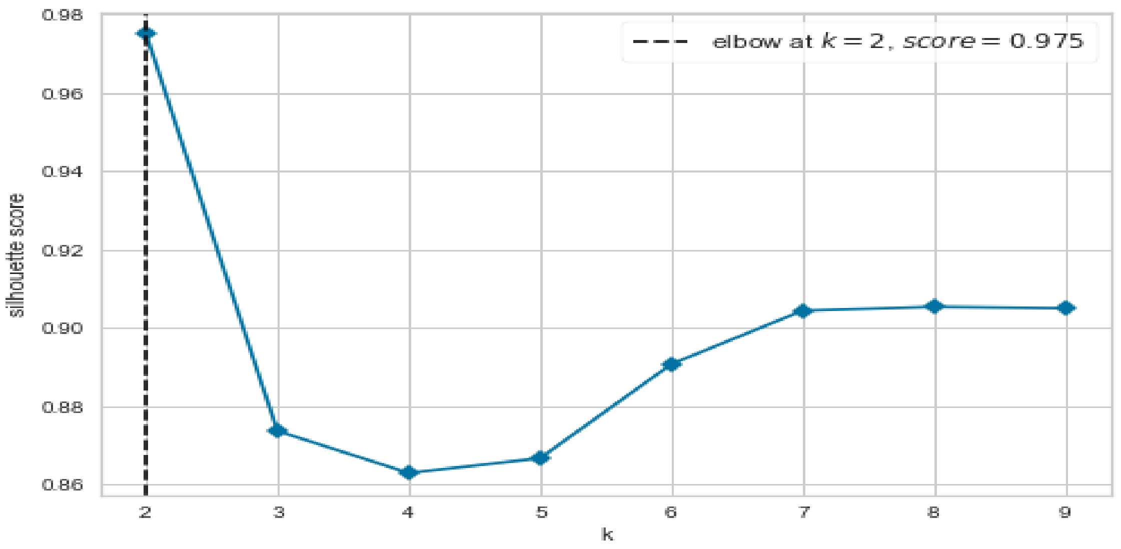


Figure 6.12: Silhouette plot for different K values

Higher silhouette score represents the optimal value of K for clustering. It is obvious from silhouette plot shown in Figure 6.12 that the cluster value K of 2 has the highest silhouette score. After getting an optimal K value, we applied

[2]https://scikit-learn.org/stable/

K-means algorithm on experimental data by using K = 2. Once the data is clustered into two clusters, we then applied the applications of formulas (discussed in Section 6.5.1) to determine the anomalies available in each cluster. To detect anomalies, we determine the centroid of each cluster to calculate the anomaly score (according to Eqn. 2) for each data point. Afterward, we calculate the values of minimum threshold *(MinT)*and maximum threshold *(MaxT)* for each cluster by using the applications of Eqn. 3 and Eqn. 4 respectively. Finally, each data point with a score greater than *(MaxT)* or less than *(MinT)* is labeled as anomalous data point. In Table 6.1, we have presented the minimum and maximum threshold values for K = 2. We also repeat the experiment by taking the cluster values as: K = 3, and K = 4 in order to observe the effect of higher cluster value for anomaly detection. The results of this experiment with different K values are presented in Table 6.2.

Table 6.1: Threshold values of MaxT and MinT with K=2.

Cluster	MinT	MaxT
One	0.9985	1.0009
Two	0.6136	1.3664

Table 6.2: K-means evaluation with different K values.

No. of clusters	Precision	Recall	F1-Measures
K = 2	1	0.52	0.68
K = 3	0.02	0.52	0.04
K = 4	0.03	0.52	0.07

During the experimental evaluation, it is observed that K-means detected (24 out of 46) artificial anomalies by using two clusters. It is obvious also from Table 6.2 with the K value 3 and four results are not promising and there are many false positives. The anomalies detected by the K-means by using *K=2* are shown in Figure 6.13, and Figure 6.14 presents the results according to the evaluation metrics.

2. *DBSCAN:* To apply DBSCAN, we need to estimate two user-defined parameters, which are the minimum points threshold *MinPts* and the neighborhood

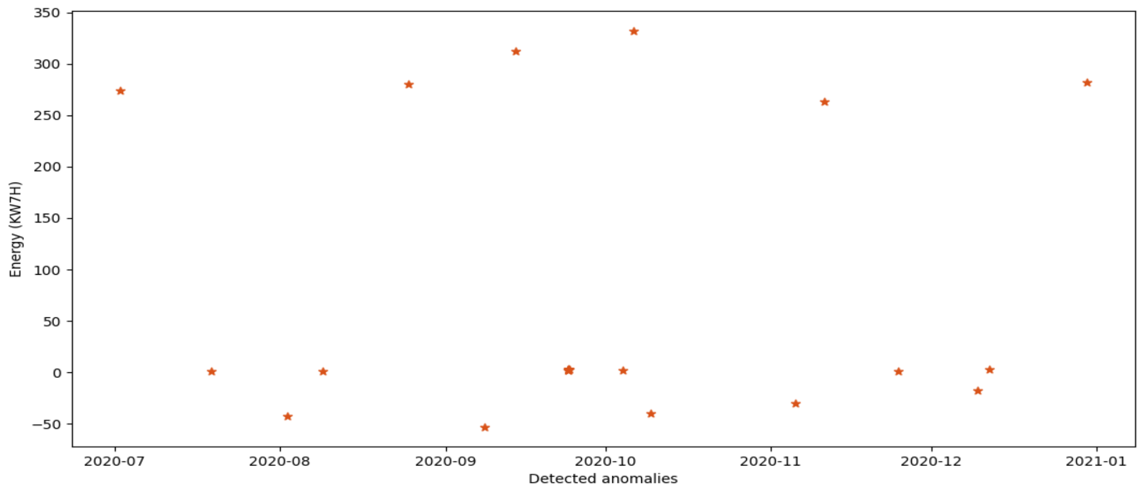


Figure 6.13: Distribution of anomalies correctly detected by K-means

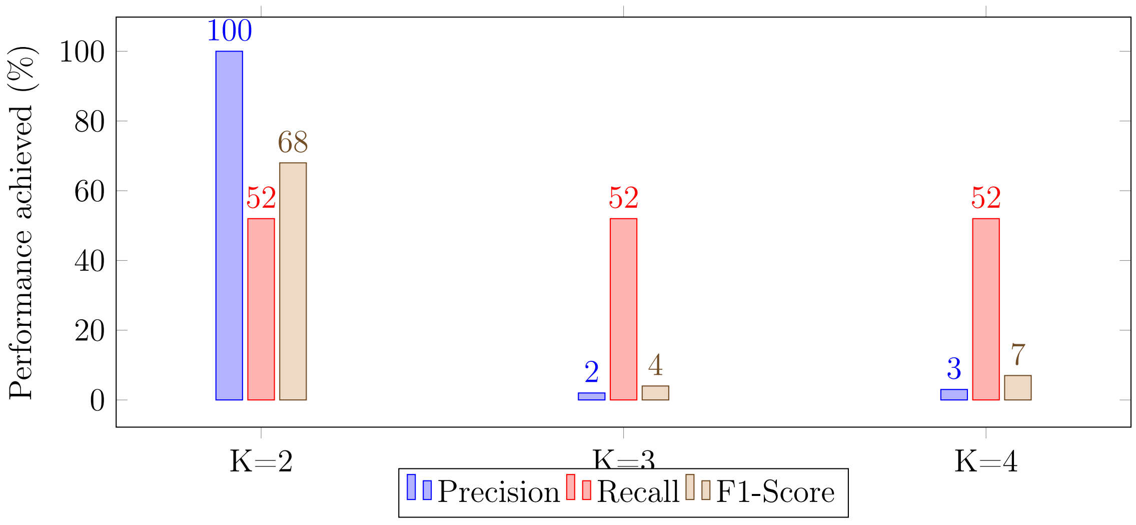


Figure 6.14: Results achieved by using K-means with different K values

radius *eps*. As a rule of thumb [167], *MinPts* should be at least the number of dimensions of the data set plus one. Usually, *MinPts* must be chosen at least 3. However, larger values are usually better for data sets with noise to obtain more significant clusters. Therefore in this work, we set it to 4.

To find an optimal value for *eps*, we can compute the *k-nearest neighbour* (KNN) distance and then plot the K-distance in ascending order and look for the knee in the plot [167]. This is called *elbow method* that corresponds to the optimal value of *eps*. The idea behind is that the data points located inside clusters will have a small KNN distance, because they are close to other data points in the same cluster, while the outliers are far away and will have a large

KNN distance [167]. Therefore, we have computed and plotted the K-nearest neighbour distance which is shown in Figure 6.15.

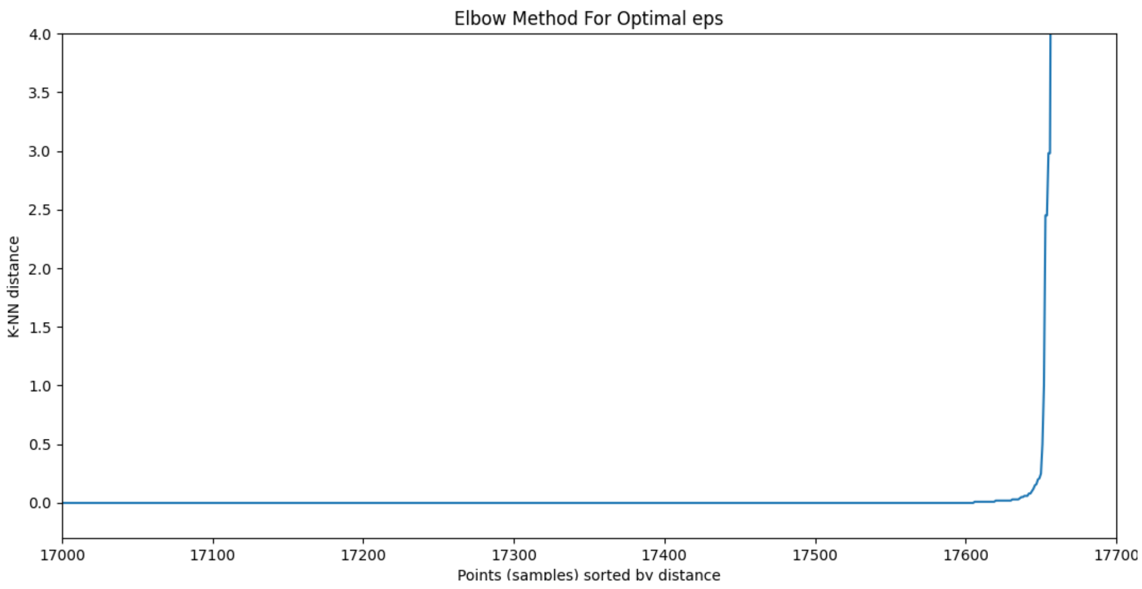


Figure 6.15: Optimal *eps* calculation

In order to detect the maximum numbers of artificial anomalies, we have tested different value for both the *eps* and *MinPts*. The used values with their effects on clustering performance are presented in Table 6.3.

Table 6.3: DBSCAN evaluation with different settings of *(eps)* and *(MinPts)*.

Eps	Min Sample	Clusters	Precision	Recall	F1-Measure
0.2	4	53	0.91	0.65	0.76
	6	51	0.93	0.83	0.88
	8	48	1	0.75	0.86
	9	48	1	0.75	0.86
	10	48	1	0.75	0.86
	12	48	1	0.75	0.86
0.25	4	29	1	0.61	0.76
	6	27	1	0.80	0.89
	8	26	0.86	0.80	0.83
	9	25	0.88	1	0.94

Continued on next page

Table 6.3 – continued from previous page

Eps	Min Sample	Clusters	Precision	Recall	F1-Measure
	10	25	0.88	1	0.94
	12	25	0.88	1	0.94
0.28	4	25	1	0.61	0.76
	6	23	1	0.80	0.89
	8	22	0.86	0.80	0.83
	9	21	0.88	1	0.94
	10	21	0.88	1	0.94
	12	21	0.88	1	0.94
0.3	4	22	1	0.61	0.76
	6	20	1	0.80	0.89
	8	19	0.86	0.80	0.83
	9	18	0.88	1	0.94
	10	18	0.88	1	0.94
	12	18	0.88	1	0.94
0.35	4	19	1	0.61	0.76
	6	18	1	0.78	0.88
	8	18	1	0.80	0.89
	9	18	1	0.80	0.89
	10	17	1	1	1
	12	17	1	1	1

From Table 6.3, it can be seen that excellent results are obtained by using the *Eps = 0.35* and *MinPts = 10*. With these parameter settings, all the artificial anomalies were detected from the PELL street lighting data with zero false positive. The anomalies detected by the DBSCAN by using *eps=0.35*, and *MinPts=10* are presented in Figure 6.16, and Figure 6.17 presents the results according to the evaluation metrics.

3. *OPTICS:* It solves one of the DBSCAN issue of not being able to cluster data points that have varying density in the data set. In order to visualize and understand how data is structured and clustered, we have plotted the reachability plot as shown in Figure 6.18. The clustering order produced by the OPTICS

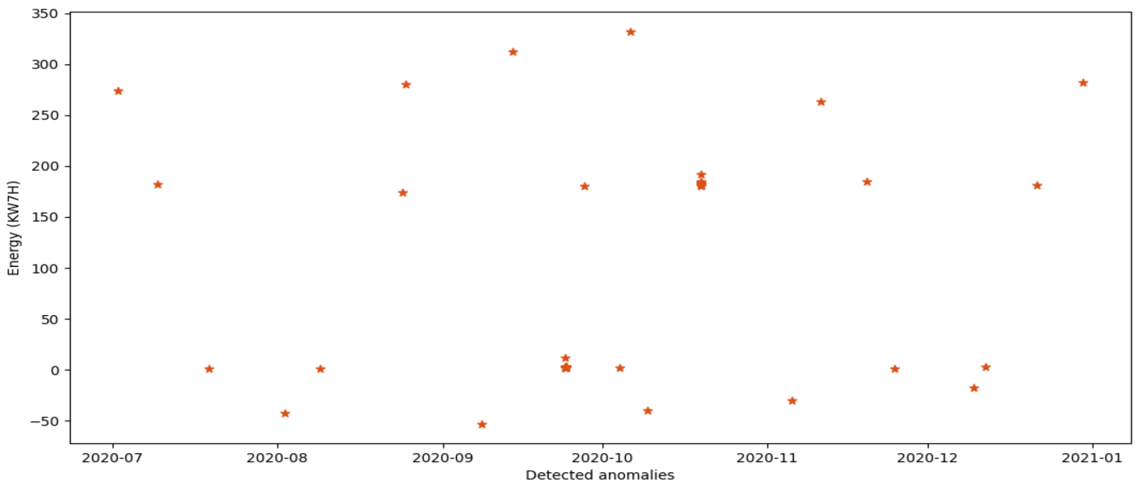


Figure 6.16: Distribution of anomalies correctly detected by DBSCAN

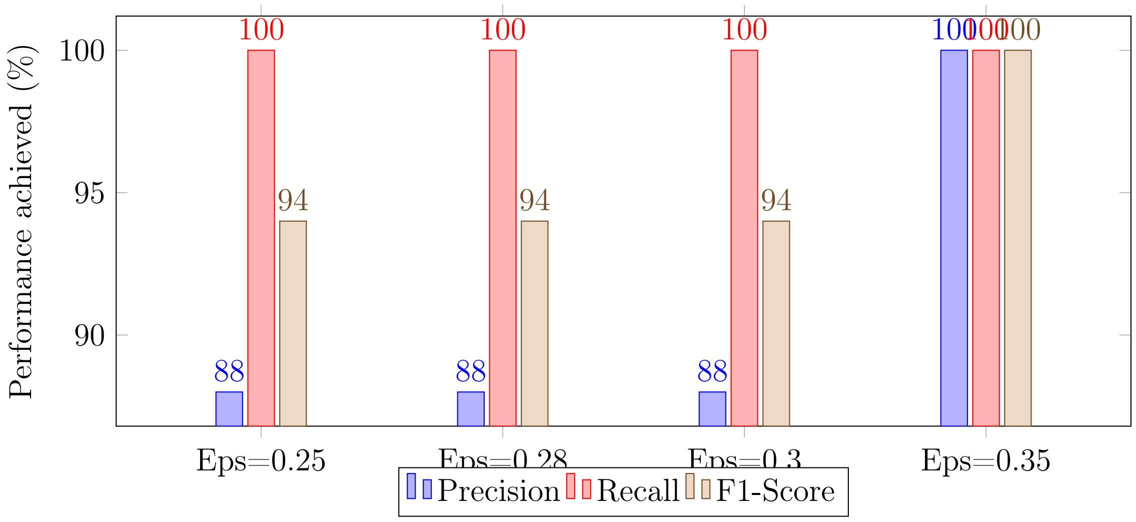


Figure 6.17: Results achieved by using DBSCAN with different Eps values

algorithm is plotted across the x-axis and reachability distance plotted along the y-axis. In reachability plot, the points with lower reachability distance indicate that they are more similar as compared to the points having higher reachability distance [125]. The points with higher reachability distance are labelled as noise points or outliers. In this study, in order to detect maximum numbers of artificial anomalies, we have tested different values for *eps* by considering reachability plot. The results obtained by using different *eps* values with *extractDBSCAN* clustering method are presented in Table 6.4.

By setting the *eps* value 0.1, OPTICS correctly detected 40 artificial anomalies out of 46 with 86 cluster count. The anomalies detected by the OPTICS with

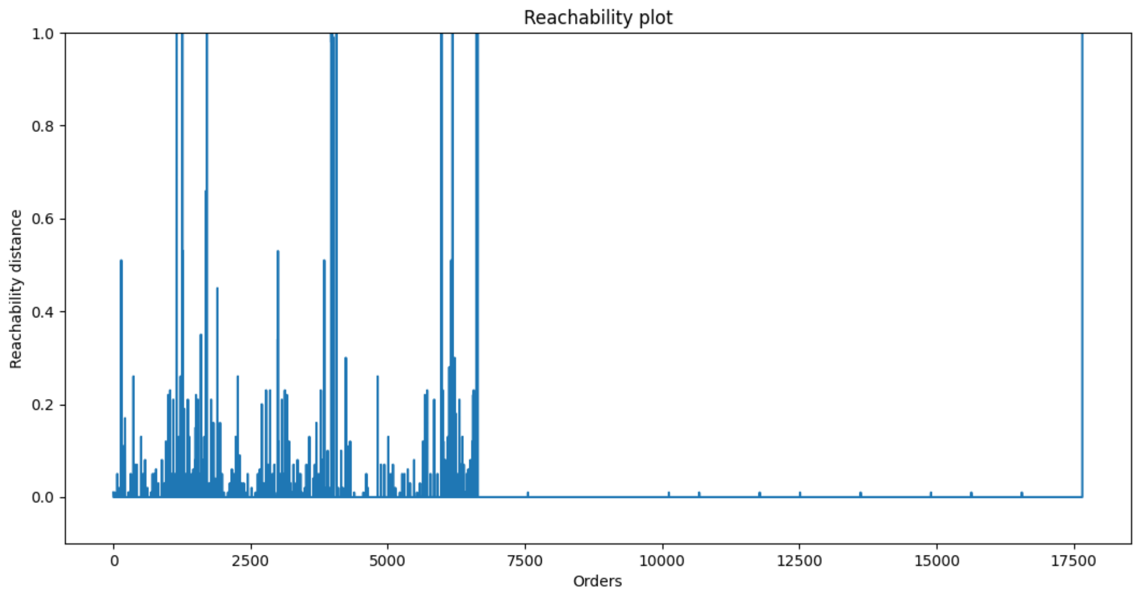


Figure 6.18: OPTICS: Reachability plot

Table 6.4: OPTICS Output

Eps Value	Cluster Count	Precision	Recall	F1-Measure
0.1	86	89.13	89.13	89.13
0.13	60	92.68	82.6	87.35
0.15	60	92.85	84.78	88.63
0.2	52	92.68	82.6	87.35

eps=0.1 as a parameter setting are shown in Figure 6.19, and Figure 6.20 shows presents the results according to the evaluation metrics.

In order to determine which clustering algorithm is more suitable to detect the anomalies from PELL street lighting data with anomalous scenarios, we have made a comparative analysis of the attained accuracy of each algorithm with optimal parameter settings. During the experimental evaluation, it is observed that K-means with cluster value 2, correctly detected all the artificial anomalies for Scenario S2, and also detected some anomalies of Scenarios S3 and S4, but it completely failed to detect the anomalies for scenarios S1, S5, and S6. In the case of DBSCAN, it is observed that with *Eps = 0.35* and *MinPts = 10* as parameter settings, DBSCAN correctly detected all the artificial anomalies for all the S1, S2, S3, S4, S5, and S6 scenarios. In the case of OPTICS, with *Eps = 0.1* it correctly detect all the anomalies of scenarios S1, S2, S5, S6 and some false positives are noted for scenario S3, and S4. Table 6.5 reports the

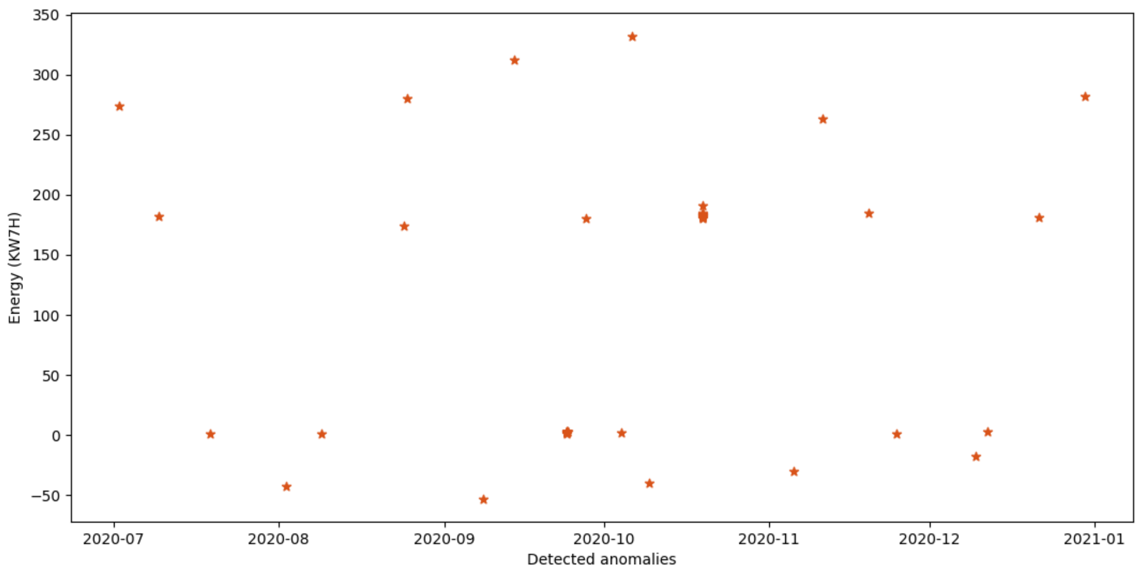


Figure 6.19: Distribution of anomalies correctly detected by OPTICS

results of K-means, DBSCA, and OPTICS with optimal parameter settings. In Figure 6.21, we have plotted the results in term of precision, recall, and F1-score for all three clustering algorithms.

Table 6.5: Algorithm comparison

Algorithm	**Parameters**	**Clusters**	**Detected Anomalies**	**Precision**	**Recall**	**F1-Score**
K-means	*k=2*	2	24	1	0.52	0.68
DBSCAN	*Eps=0.35, MinPts=10*	17	46	1	1	1
OPTICS	*Eps = 0.1*	86	49	0.89	0.89	0.89

RQ1 summary: According to our experience, DBSCAN outperformed K-means and OPTICS in terms of prediction accuracy. It correctly detected all the synthetic anomalies for all six anomalous scenarios. OPTICS also produced satisfactory results, but K-means badly failed to detect synthetic anomalies especially in the case when we increased number of clusters.

6.6.3.2 Answers to RQ2 (Clustering Execution Cost Locally)

To determine the computation cost of used clustering algorithms is also another important objective of our study. Therefore, we have collected the execution time of

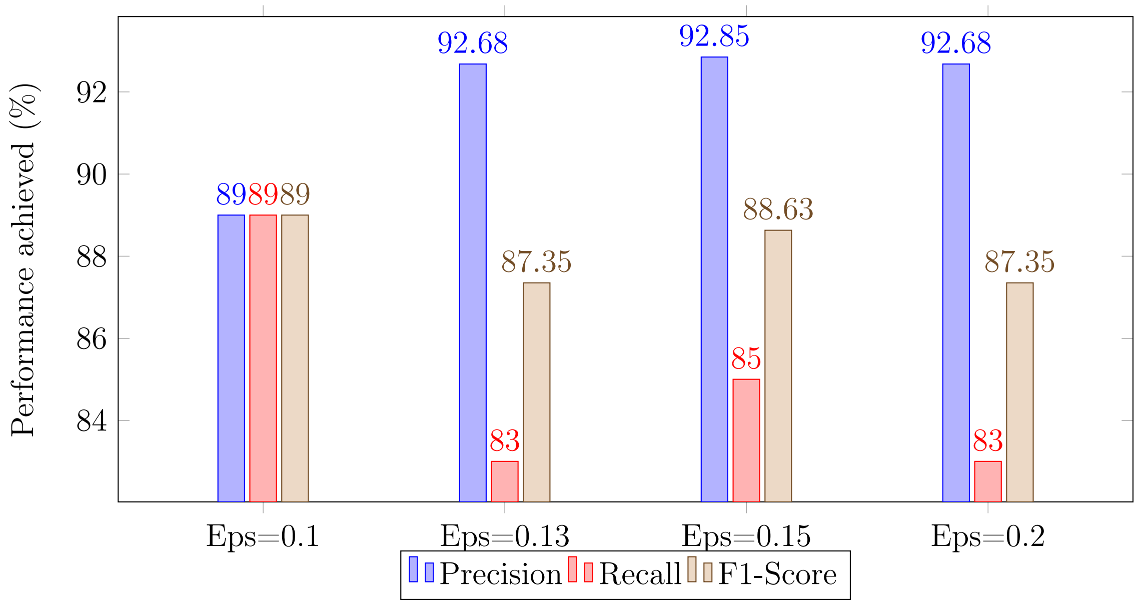


Figure 6.20: Results achieved by using OPTICS with different Eps values

clustering algorithms by using three different hardware settings. We executed each algorithm five times against every single parameter setting and then average time is noted. Table 6.6 presents the hardware specification used for this analysis.

Table 6.6: Hardware specifications for the analysis of computation cost.

	Machine 1	**Machine 2**
CPU	Core(TM) i7-10510U, CPU @1.80 GHz	Xeon(R) Silver 4114, CPU @2.20 GHz
Memory (GB)	8	15
Operating System	Windows 10 pro (64-bit)	CentOS (64-bit)

In the very first analysis, we determine the computation cost by using a local machine. The Apache Spark is configured with single node and local file system is used to read the data set. The specifications of this machine 1 are given in Table 6.6. Table 6.7, Table 6.8 and Table 6.9 reports the execution times taken by the K-means, DBSCAN, and OPTICS respectively.

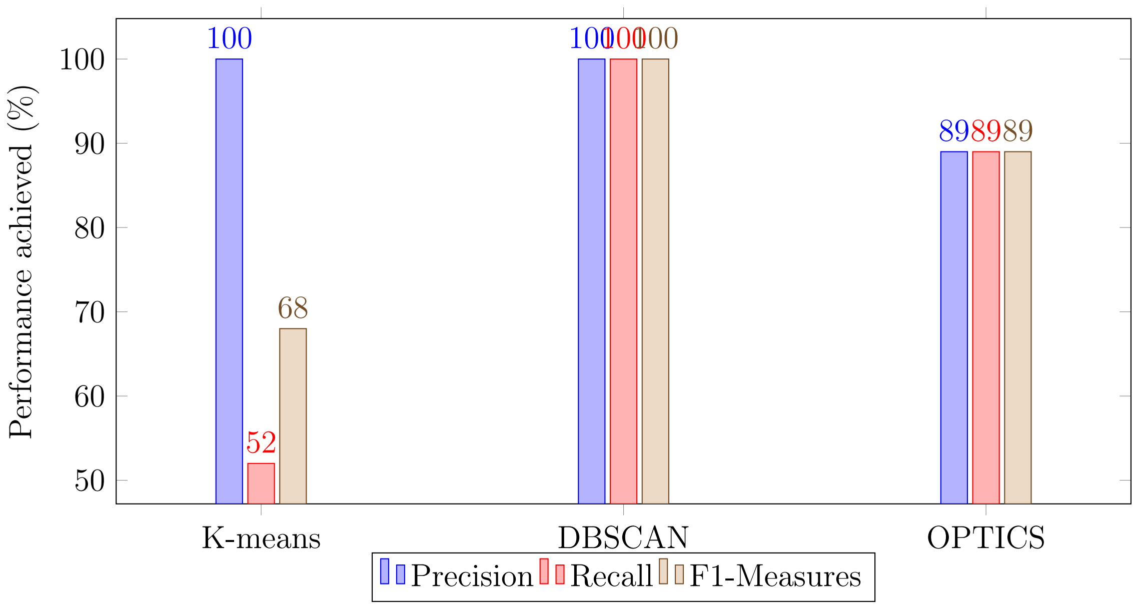


Figure 6.21: Performance comparison of anomaly detection models with optimal parameters

Table 6.7: K-means execution time (in seconds) on machine 1.

No. of cluster	Execution Time	Anomalies
K = 2	0.07	24
K = 3	0.10	924
K = 4	0.12	877

Table 6.8: DBSCAN execution times (in seconds) on machine 1.

Eps	Min Sample	Clusters	Execution Time	Anomalies
0.2	4	53	1.74	33
	6	51	1.75	41
	8	48	1.74	61
	9	48	1.74	61
	10	48	1.76	61
	12	48	1.81	61
0.25	4	29	1.83	28
	6	27	1.83	37

Continued on next page

Table 6.8 – continued from previous page

Eps	Min Sample	Clusters	Execution Time	Anomalies
	8	26	1.84	43
	9	25	1.86	52
	10	25	1.77	52
	12	25	1.82	52
0.28	4	25	1.75	28
	6	23	1.78	37
	8	22	1.72	43
	9	21	1.77	52
	10	21	1.80	52
	12	21	1.80	52
0.3	4	22	1.78	28
	6	20	1.74	37
	8	19	1.76	43
	9	18	1.76	52
	10	18	1.76	52
	12	18	1.79	52
0.35	4	19	1.71	28
	6	18	1.74	36
	8	18	1.77	37
	9	18	1.76	37
	10	17	1.76	46
	12	17	1.80	46

Table 6.9: OPTICS execution times (in seconds) on machine 1.

Eps	Detected Clusters	Execution Time	Anomalies
0.1	86	23.4	46
0.13	60	23.8	42
0.15	60	23.8	42
0.2	52	24.7	41

In this analysis, it is observed that k-means is very fast as compared to DBSCAN and OPTICS as shown in Table 6.7. The only problem observed with the K-means is the large numbers of false positive especially in the case when we increase the cluster value. DBSCAN is also performed well to finish its job, we observed excellent results with DBSCAN in the case of accuracy as well as efficiency. In the case of OPTICS, it took more time to complete its job in comparison to K-means and DBSCAN as shown in Table 6.9.

In the second analysis, we used a Linux based local machine and the Spark configuration is similar to the first analysis i.e. single node but for data management Hadoop distributed file system (HDFS) is used. The specifications of this machine 2 are given in Table 6.6. Table 6.10, Table 6.11, and Table 6.16 presents the execution times of all three clustering algorithms.

Table 6.10: K-means execution times (in seconds) on machine 2.

No. of cluster	Execution Time	Anomalies
K = 2	0.40	24
K = 3	0.60	924
K = 4	0.57	877

Table 6.11: DBSCAN execution times (in seconds) on machine 2.

Eps	Min Sample	Clusters	Execution Time	Anomalies
0.2	4	53	1.68	33
	6	51	1.67	41
	8	48	1.66	61
	9	48	1.67	61
	10	48	1.68	61
	12	48	1.67	61
0.25	4	29	1.68	28
	6	27	1.67	37
	8	26	1.67	43
	9	25	1.67	52
	10	25	1.68	52
	12	25	1.67	52

Continued on next page

Table 6.11 – continued from previous page

Eps	Min Sample	Clusters	Execution Time	Anomalies
0.28	4	25	1.67	28
	6	23	1.67	37
	8	22	1.69	43
	9	21	1.68	52
	10	21	1.67	52
	12	21	1.67	52
0.3	4	22	1.67	28
	6	20	1.69	37
	8	19	1.68	43
	9	18	1.67	52
	10	18	1.67	52
	12	18	1.68	52
0.35	4	19	1.68	28
	6	18	1.69	36
	8	18	1.68	37
	9	18	1.68	37
	10	17	1.68	46
	12	17	1.69	46

Table 6.12: OPTICS execution times (in seconds) on machine 2.

Eps Value	Detected Clusters	Exe. Time	Anomalies
0.1	86	43.51	46
0.13	60	43.35	42
0.15	60	43.40	42
0.2	52	43.05	41

In this analysis, we observed that K-means took more time as compared to the time taken by using machine 1. Same behavior is observed for OPTICS, but we noticed a little improvement in the time taken by DBSCAN. The overall performance of clustering algorithms on these hardware settings is observed to be slow as compared to the first analysis.

RQ2 summary: In term of computation speed, K-means observed to be a very fast algorithm as compared to DBSCAN and OPTICS. DBSCAN also completed its process of clustering and anomaly detection with the highest accuracy in less time. But OPTICS took more time and was observed a bit slower to complete its process of anomaly detection.

6.6.3.3 Answers to RQ3 (Clustering Execution Cost at Scale)

In this analysis, we configured Apache Spark in a cluster scale and HDFS is used for data management. Spark cluster is comprised of one master node and four worker nodes. The specifications of the cluster components are presented in Table 6.13. Table 6.14, Table 6.15, and Table 6.16 reports the execution times taken by the K-means, DBSCAN, and OPTICS respectively.

Table 6.13: Hardware specifications for the analysis of computation cost.

	Master	**Worker 1**	**Worker 2**	**Worker 3**	**Worker 4**
CPU	12 vCPU (12 vCore)	6 vCPU (6 vCore)	6 vCPU (6 vCore)	6 vCPU (6 vCore)	6 vCPU (6 vCore)
Memory (GB)	96	64	64	64	64
Hard Disk (TB)	2	3	3	3	3
Operating System	CentOS (64-bit)	CentOS (64-bit)	CentOS (64-bit)	CentOS (64-bit)	CentOS (64-bit)

Table 6.14: K-means execution times (in seconds) on cluster scale.

No. of cluster	**Execution Time**	**Anomalies**
K = 2	0.34	24
K = 3	0.57	924
K = 4	0.54	877

Table 6.15: DBSCAN execution times (in seconds) on cluster scale.

Eps	**Min Sample**	**Clusters**	**Execution Time**	**Anomalies**
0.2	4	53	1.63	33

Continued on next page

Table 6.15 – continued from previous page

Eps	Min Sample	Clusters	Execution Time	Anomalies
	6	51	1.63	41
	8	48	1.62	61
	9	48	1.62	61
	10	48	1.62	61
	12	48	1.62	61
0.25	4	29	1.64	28
	6	27	1.63	37
	8	26	1.63	43
	9	25	1.63	52
	10	25	1.62	52
	12	25	1.63	52
0.28	4	25	1.64	28
	6	23	1.63	37
	8	22	1.63	43
	9	21	1.64	52
	10	21	1.63	52
	12	21	1.63	52
0.3	4	22	1.64	28
	6	20	1.63	37
	8	19	1.63	43
	9	18	1.63	52
	10	18	1.63	52
	12	18	1.63	52
0.35	4	19	1.63	28
	6	18	1.63	36
	8	18	1.62	37
	9	18	1.64	37
	10	17	1.64	46
	12	17	1.63	46

In this analysis, we observed a very similar behavior of clustering algorithms

Table 6.16: OPTICS execution times (in seconds) on cluster scale.

Eps Value	Detected Clusters	Exe. Time	Anomalies
0.1	86	42.94	46
0.13	60	43.08	42
0.15	60	43.16	42
0.2	52	43.05	41

that was noted in the second analysis. K-means and OPTICS almost took the same time as noted with the second machine. In the case of DBSCAN, we again observed a little improvement in comparison to machine 1 and machine 2. The computation speed of clustering algorithms observed very slow on cluster scale. There could be certain possible factors behind these results, for example, the data set used for the experimentation purpose is not large enough to take the benefits of Spark distributed computing. Furthermore, for the implementation of clustering algorithms, we used Panda data frame with Scikit-learn[3] (ML library for Python programming language) instead of MLib [4] (Spark's ML library) with spark data frame, because MLib does not support the implementation of DBSCAN and OPTICS. So these could be the possible factors that hinder to take full benefits of Spark distributed computing at cluster scale.

RQ3 summary: In this analysis, it is observed that the computation speed of the clustering algorithms could not improve by using Spark at cluster scale, and it could be due to certain factors. For example, the experimental data set is not large enough, the implementation of clustering algorithms is not based on MLib.

6.7 Threats to Validity

Construct validity threats may arise due to assumptions made when we introduced artificial anomalies in our data set. To mitigate these threats, we used a PELL data set based on an established benchmark from ENEA and considered typical and well-know anomalies that usually are observed in the PELL street lighting data.

Internal validity threats may be caused by bias in establishing cause-effect relationships in our experiments. To limit these threats, we performed extensive exper-

[3]`https://scikit-learn.org/stable/`
[4]`https://spark.apache.org/mllib/`

iments for each clustering algorithm by setting different parameter settings. We not only evaluated the effectiveness of parameter setting in term of algorithm accuracy but also its effect on computation cost of the algorithm is also determined. We also enable replication by making the experimental results publicly available.

External validity threats may exist if the characteristics of the data set in our benchmark are not indicative of the characteristics of data produced by other street lighting systems. This type of threat applied to our context at some extent since we rely on the PELL data model that is aimed at collecting and treat in an uniform way all different characteristics related to heterogeneous street lighting plants. Moreover, we conducted the evaluation by running the algorithms both locally and at scale with Spark running on a local machine and on cluster mode, respectively.

To avoid threats to *conclusion validity*, we repeated our experiments multiple times. In particular, performance metrics (precision and recall) have been measured calculating them with a data set of 17664 observations. Furthermore, to measure the computation cost, each experiment has been repeated five times both locally and at scale.

Regarding the *scalability analysis*, we do not perform any scalability study in terms of high volume of data (to gain the full benefits of Spark's distributed computing with HDFS) and high number of anomalies (to observe the accuracy of clustering algorithms by introducing large number of artificial anomalies). We postpone scalability analysis as future work such a study with the availability of a very larger data set from different municipalities.

6.8 Implementation details and lesson Learned

This anomaly detection module is developed as part of the PELL SCP which is discussed in Section-II. In addition to the study of the peculiarities of the public street lighting domain, the development and evaluation of the clustering algorithms on BD required us a learning curve for gaining a wide spectrum of multi-disciplinary skills. We do not report about the amount of time taking to develop this anomaly detection module since we consider such project management issues out of the scope of this work.

From a software engineer perspective, it is required to know how to set up the runtime environment, i.e. how to install Hadoop, Apache Spark, and all other Python libraries including ML libraries such as pandas, scikit-learn, Matplotlib, etc. The seamless connection of all these software solutions through a sequence of configuration steps requires a certain amount of experience.

From a software development point of view, python programming skills are required, then the focus should be put on applying the stack of libraries of Spark for large scale data processing, and Python ML library (scikit-learn) for the implementation of clustering algorithms. We used PySpark to transforms the data set from the JSON file to Spark DataFrame for BD processing. After data processing we transformed Spark DataFrame to the pandas DataFrame because MLib Apache Spark's ML library does not support the implementation of DBSCAN and OPTICS. After this conversion, we exploited scikit-learn for the implementation of clustering algorithms. To build a maintainable and scalable project in Python, knowledge of the OOP paradigm and of principles for developing small micro-services is highly suggested.

6.9 Summary

In this chapter, we presented an anomaly detection approach that is developed as part of the PELL SCP. Specifically, we investigated three well-known clustering algorithms: Kmeans, DBSCAN, and OPTICS. The performance of these clustering algorithms is examined on a data set made of real electricity consumption data from the PELL SCP. For the public street lighting domain, we proposed six anomalous scenarios which could lead to a possible anomaly in the platform. Two types of synthetic anomalies point anomalies, and collective anomalies, are introduced. These synthetic anomalies helped us to evaluate the clustering algorithms by using standard evaluation metrics such as precision, recall, and F1-measure. We performed extensive experiments to evaluate the performance of clustering algorithms by using different parameter settings. We not only tested the clustering algorithms to measure their accuracy but we also evaluated the computation cost of every algorithm. In a comparative analysis, it was observed that DBSCAN produced better accuracy results as compared to K-means and OPTICS. It correctly detected all the synthetic anomalies for all six anomalous scenarios.

In the next chapter, we will focus on the fourth objective: an empirical investigation of predictive models to select a suitable forecasting model for street lighting energy consumption. The forecasting approach is developed as part of the PELL SCP, which is integrated as an independent module inside the PELL SCP.

Chapter 7

Energy Consumption Forecasting in the PELL SCP

This chapter address the final objective of this dissertation: empirical investigation of predictive models to select a suitable forecasting model for street lighting energy consumption forecasting. To achieve this objective, we present an energy consumption forecasting approach for street lighting energy data based on machine learning models. In addition, a novel rule-based predictor for street lighting domain is developed to forecast street lighting energy consumption. A comparative analysis is performed on two different street lighting data sets to compare the performance and determine a suitable forecasting model for street lighting energy consumption.

7.1 Background Concepts

7.1.1 What is a Time Series?

A time series is a collection of data points that are measured sequentially over successive times. Mathematically, it is described as a set of vectors, for example $p(t)$, $t = 0, 1, 2, ...$ where t denotes the time elapsed [168, 169, 170], and the variable $p(t)$ is treated as a random variable. In a time series, the measurements collected during an event are organized in a chronological order.

A time series that contains the records of a single variable is known as univariate, whereas if it holds the records of more than one variable then it is termed as multivariate. A time series can be classified into two categories such as continuous or discrete time series. A continuous time series contains measurements at every point in time, whereas a discrete time series only includes measurements at specific points in time. The examples of a continuous time series are temperature readings, concentration of a chemical process, flow of a river etc. On the other hand, discrete time series may be represented by the population of a particular city, production of a company, or the exchange rates between two different currencies.

7.1.2 Components of a Time Series

A time series in general is composed of four components: *Trend*, *Cyclical*, *Seasonal*, and *Irregular* [171].
Trend: Trend also known as Secular Trend is the general tendency of a time series to increase, decrease or stagnate over a long period of time. Thus, trend can be a long term movement in a time series [171]. For instance, a series contains the population growth, number of houses in a city show upward trends, whereas a downward trend can be observed in a series that contains the epidemics information, mortality rates etc [171]. A trend can be linear as well as non-linear.
Seasonal Variation: A seasonal variation is a periodic movement in a time series that recurs within a year. The seasonal variation is largely influenced by climate and weather conditions, traditional habits, customs, etc. For instance, ice cream sales rise in the summer and wool clothing sales rise in the winter.
Cyclical Variation: A cyclic variation in a time series refers to medium-term changes in the series caused by circumstances which repeat in cycles. A cycle lasts for a longer period of time, typically two years or longer [171].
Irregular Fluctuation: Irregular or random Fluctuations in a time series occur due to unpredictable influences, which are purely random and unpredictable and do not repeat in a particular pattern. These fluctuations are caused by incidences such as war, earthquakes, floods, strikes, etc. [171].

7.2 Problem Domain: The PELL SCP

In this work, we proposed an energy consumption forecasting approach for public street lighting. The proposed approach is developed as part of the ENEA PELL SCP [121]. The proposed energy consumption forecasting approach is integrated as an independent module in PELL SCP as shown in the Figure 7.1.

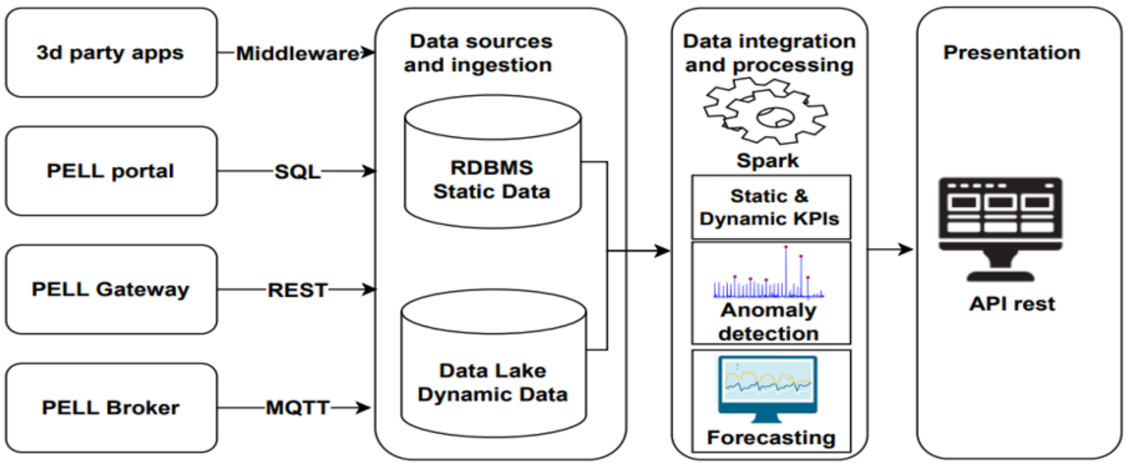


Figure 7.1: PELL SCP with forecasting module

7.2.1 PELL Street Lighting Data

PELL street lighting data is composed of electrical energy consumption measures collected by smart meters installed into the electrical panels located at Casaccia, Rome, Italy. This data is retrieved through a MQTT broker and represented in the JSON format. We have used electric consumption measures, which are collected from 1st September 2020 to 30th November 2020. This consumption is measured with the granularity of fifteen minutes, and each JSON file contains 96 values (4 values per hour and per 24 hours of a day). Figure 7.2 shows an example of PELL street lighting electricity consumption series.

7.2.2 Street Light Energy Usage Data set

In this study, we have also used a publicly available dataset known as the Street Light Energy Usage data set[1]. This data set is used for street lighting energy consumption forecasting. It contains two years (from Oct 2014 to Oct 2016) worth of energy consumption of 15 street light smart meters. The consumption is measured with the granularity of fifteen minutes, and data is collected from the streets of city of Las Vegas, USA. Figure 7.3 represents the distribution of energy usage for each smart meter. We used two smart meters (CC029790952, CC030050742) for empirical evaluation. Figure 7.4 and Figure 7.5 shows the energy consumption series for the used smart meters. We utilised the street light energy usage data for the selected smart meters without applying any preprocessing step, because we want to

[1]https://data.world/lasvegasnevada/street-light-energy-usage/

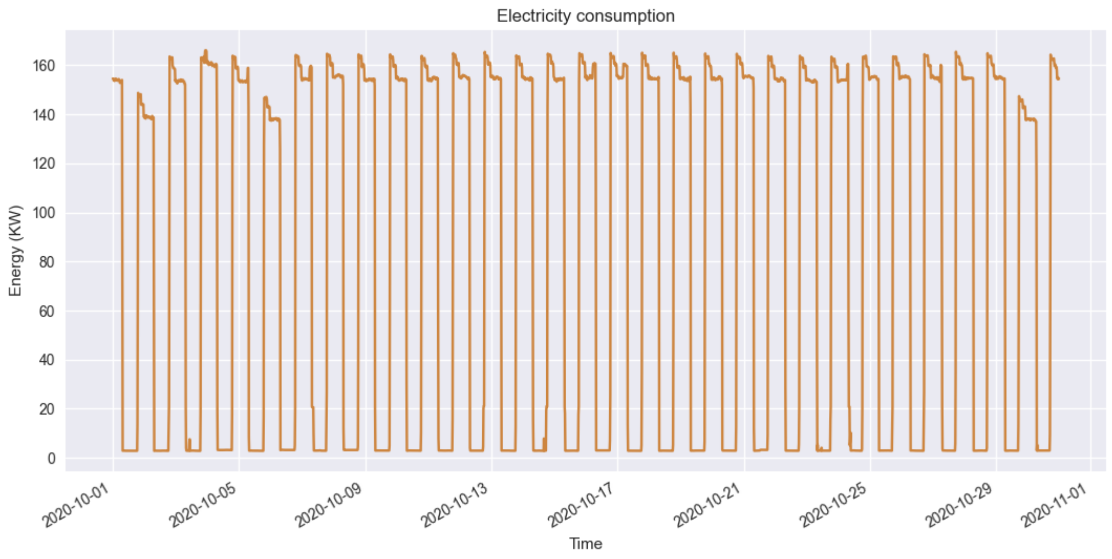


Figure 7.2: An example of electricity consumption series of PELL data

demonstrate the effectiveness of ML forecasting models on both preprocessed and raw data.

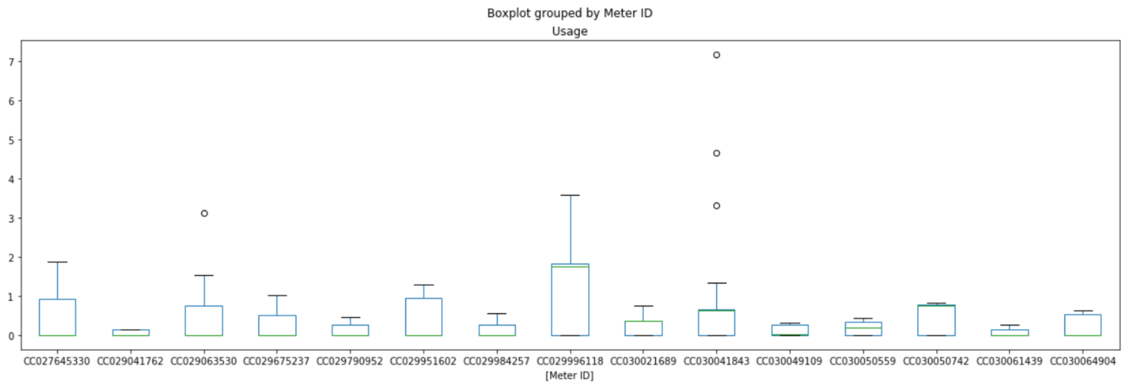


Figure 7.3: Energy usage w.r.t. smart meter

7.3 Related Works

Various forecasting models have been reported in the literature to accurately forecast the electrical energy consumption because of economic, environmental, and technical reasons. These models can be categorized into three groups: data driven [172, 173, 174, 175, 176], AI [177, 178, 179, 180, 181], and classical methods. Data driven models utilize past observations and predict the desire outcome. Classical models

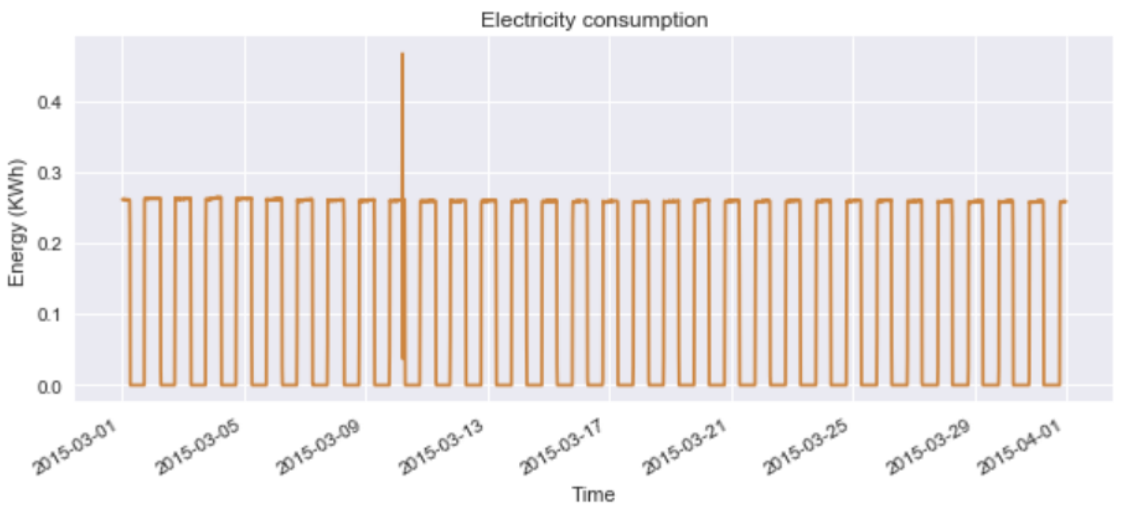


Figure 7.4: An example of electricity consumption series of smart meter with ID: CC029790952

comprise of statistical and mathematical models like Autoregressive Integrated Moving Average (ARIMA) [182, 183], Seasonal ARIMA (SARIMA) [184, 185], RF [186, 187] etc. In contrast, AI techniques like Feed Forward Neural Network (FFNN), CNN, LSTM, etc. mimic the behavior of biological neurons. In this section we have discussed the some notable efforts reported in the literature for time series energy consumption forecasting by using data driven, classical and AI based techniques.

Pedro et al. [188] utilized regularized ML models i.e. Lasso, Lars, Lasso Lars, Ridge, Elastic Net, and RF, with the intention to forecast Brazilian power power electricity consumption for short and medium terms. Authors performed predictions for 6 different forecast horizon: 1, 7, 15, 30, 60, and 90 days ahead. They divided the horizons into three groups such as (1) very short-term forecast group (VSTFG) that includes 1 and 7 days, (2) short-term forecast group (STFG) containing 15 and 30 days and (3) medium-term forecast group (MTFG) including 60 and 90 days. Beside the power electricity consumption its lagged values, authors also used calendar variables, weather variables, price of electrical energy, and several economic variables. After experimental evaluation, they concluded that ML methods, especially RF and Lasso Lars more accurate results for all horizons. Chou et al. [189] performed a comprehensive analysis of ML techniques for time series energy consumption forecasting using actual data. The primary objective of their study was to investigate the performance of single, ensemble, and hybrid techniques for predicting the energy consumption of buildings. They used five models such as Artificial Neu-

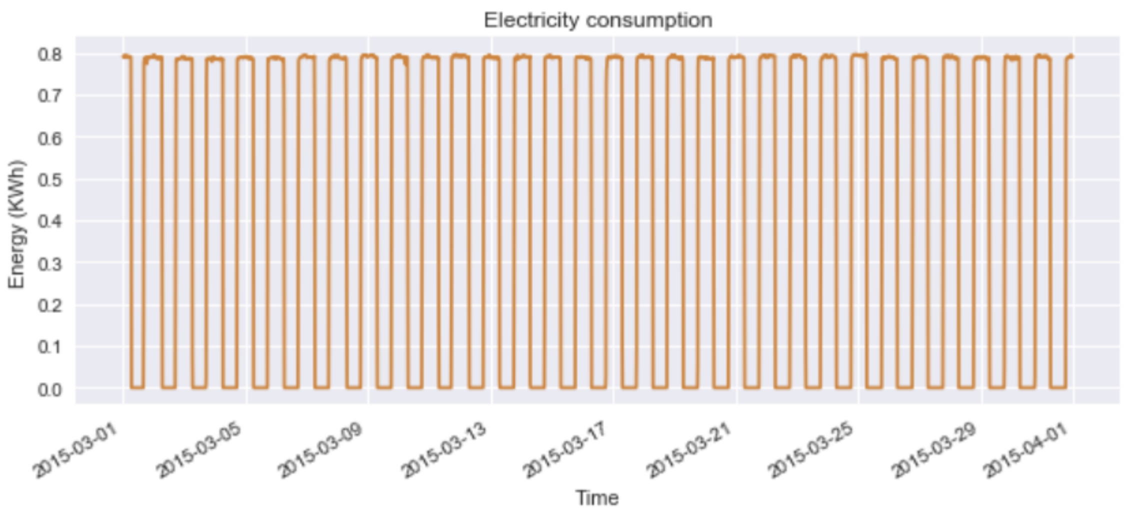


Figure 7.5: An example of electricity consumption series of smart meter with ID: CC030050742

ral Networks (ANNs), SVR, Classification and Regression Tree (CART), LR, and SARIMA to build single models, whereas author used the combinations of these models to produce two ensemble models, involving voting and bagging. Authors also proposed two hybrid models: SARIMAMetaFA-LSSVR and SARIMA-PSO-LSSVR. They employed three analytic platforms - RapidMiner Studio, IBM SPSS Modeler, and WEKA to implement the used models. Experimental evaluation of the used models is performed on a real-time energy consumption dataset obtained from smart grid network of Taiwan. Authors concluded that the best AI single model was ANNs, whereas the best ensemble model was the ANN-based bagging model, and The better effective hybrid model was SARIMA-MetaFA-LSSVR which is implemented in MATLAB.

In another study [190], Evolutionary Hybrid System (EvoHyS) is proposed for energy consumption forecasting data in smart meters. The proposed system EvoHyS is composed of three stages. The first stage performs linear modeling by using the applications of the SARIMA model, and at the second stage of the system, an evolutionary optimization based on a genetic algorithm is utilized to determine the best hyper-parameter of the SVR model. In the final stage of the proposed system, a combination of linear and non-linear models is performed using an MLP optimized by a genetic algorithm. The experimental evaluation of the proposed forecasting system is performed on a dataset of smart grid network installed in a residential building, and performance is measure by using well know evaluation metrics i.e.

Correlation Coefficient (R), Mean Absolute Percentage Error (MAPE), Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and Maximum Absolute Error (MaxAE). The experimental results demonstrate that the proposed approach outperformed the existing state-of-the-art approach [189]. In [191], authors investigated the usage of DL techniques in order to perform energy demand forecasting. They proposed a mixed architecture comprising of a CNN coupled with an ANNs. The objective of using this mixed architecture was to take the advantage of the virtues of both structures: the regression capabilities of the artificial neural network and the feature extraction capabilities of the CNN. The evaluation of the proposed architecture was performed on the historical data of French energy consumption. The experimental results show that proposed approach outperforms the reference system. Moreover, authors also achieved better forecasting results with comparison to ARIMA and tradition ANN models. In another study [192], authors performed a comparative analysis between traditional ML approaches and DL methods to accurately predict the aggregated energy consumption. The used ML models were LR, Multiple Linear Regression (MLP), SVR with RBF kernel, SVR with polynomial kernel, SVR with linear kernel, Regression Tree (RT), Ensemble regression tree, Ensemble Tree LS Boost, Ensemble Tree Bag, Gaussian process (linear), Gaussian process and DL model was MLP. In order to explore the prediction capabilities of MLPs different parameter settings were employed. Experimental evaluation was performed on a real-world dataset provided by Scholt Energy Control B.V., The Netherlands. The obtained results demonstrate that MLPs enhanced with DL capabilities present better prediction accuracy.

Jihoon et al. [193] authors conducted a comparative analysis of diverse Artificial Neural Network (ANN) models for short-term load forecasting (STLF). Authors constructed different ANN models by tuning two hyperparameters such as number of hidden layers and activation functions. They considered rectified linear unit (ReLU), leaky rectified linear unit (LReLU), parametric rectified linear unit (PReLU), exponential linear unit (ELU), scaled exponential linear unit (SELU) as activation functions, and the number of hidden layers from 1 to 10. In order to compare the prediction performance with two hyperparameters for the STLF model, authors used electric load data collected from five different types of buildings for 2 years, and two performance metrics such as coefficient of variation of the root mean square error (CVRMSE) and MAPE. The authors concluded that SELU-based model with five hidden layers produced better average performance than other ANN-based models for short-term load forecasting. In [194], authors proposed kCNN-LSTM, a DL framework for robust and reliable building energy consumption. kCNN-LSTM uses (i) k-means clustering to perform cluster analysis for trend characterization; (ii) CNN – to extract complex energy related features with non-linear interactions; and (iii) LSTM neural networks – to handle long-term dependencies and model temporal information in energy consumption data. The effectiveness of the proposed kCNN-LSTM framework was demonstrated by using a real-time building energy consumption data acquired from a four-storeyed building in IIT-Bombay, India. The ability to learn the spatio-temporal dependencies in the energy consumption data makes the kCNN-LSTM a suitable DL model for energy consumption forecasting. Pavlicko et al. [195] authors utilized and compared different forecasting models to predict the maximum hourly electricity consumption per day. Authors categorized the used model into two groups. The first model group is is based on the transverse set of Grey models and Nonlinear Grey Bernoulli models. The second group is based on a multi-layer feed-forward back-propagation network. Furthermore, the authors also proposed a new potential hybrid model by combining these approaches, which is used to forecast the maximum hourly electricity consumption per day. Experimental results show that the hybrid model offered the best results according to the used performance metrics.

Khan et al. [196] proposed a spatial and temporal ensemble forecasting model for short-term electric consumption forecasting. The developed ensemble forecasting model comprises of two DL models such as LSTM, and Gated Recurrent Unit (GRU). The proposed ensemble forecasting model was capable to forecasts the electric consumption at three spatial scales apartment, building, and floor level for hourly, and daily. The evaluation of the proposed approach is performed on the consumption data belonging to four multifamily residential buildings in South Korea. Experimental results proved that the proposed model surpassed the counterpart solutions and attained superior performance. In [197], authors proposed a ranger-based online learning approach called RABOLA for electricity consumption forecasting for buildings with complex energy consumption patterns. For model training, they utilized publicly available electricity consumption data from office buildings in Richland, WA, U.S. Experimental data set was divided into the training and test set and applied the most famous ensemble learning approaches such as bagging, boosting, and stacking. On training data, the authors employed RF (bagging), Gradient Boosting Machine (GBM) (boosting), and Extreme Gradient

Boosting (XGB) (boosting) to develop STLF models. Whereas, on the test, they used the stacking ensemble technique to develop an online learning-based multistep-ahead STLF model using the predicted values of the RF, GBM, and XGB models. The authors performed extensive comparative experiments to demonstrate that the proposed experimentation RABOLA model outperforms the prediction performance of state-of-the-art stacking ensemble and DL approaches.

Henzel et al. [198], proposed an approach to forecast the day-ahead energy consumption by utilizing unique data obtained from a digital twin model of a building. Authors employed diverse forecasting models such as naive method, LR, LSTM, and Prophet method by using various set of features. After an extensive analysis, they observed that Prophet model using information about the total energy consumption and real data about the energy consumption of the top 10 energy-consuming devices produced the best forecast accuracy. In [199], authors came with a novel hybrid AI-empowered forecasting model, which combines singular spectrum analysis (SSA) and parallel long short-term memory (PLSTM) neural networks. They observed that the decomposition with the SSA enhanced the performance of the PLSTM network. Experimental results show that the proposed novel hybrid model outperforms the state-of-the-art models at different time intervals in terms of prediction accuracy as well as computational efficiency. In [200], authors proposed an ensemble approach for the estimation of electricity in Agartala, Tripura in India. The developed ensemble model is based on RF and XGBoost. The authors also evaluated the performance of individual RF and XGBoost with their ensemble. The proposed model is capable of accurately predicting the next 24 h of load with an estimation of load for 1 week to 1 month. Rodrigues et al [201], used ANN to predict daily and hourly short-term load and household electricity consumption. They consider apartment location, occupants' numbers, electric appliance consumption, and hourly meter system to train the model. In this work, a feed-forward ANN and the Levenberg-Marquardt algorithm produced better forecasting results.

Ahmad et al. [202] performed a comparative analysis of two ML models namely ANNs and RF. to predict hourly Heating, Ventilation, and Air Conditioning (HVAC) energy consumption of a hotel in Madrid, Spain. According to the experimental results, it was observed that ANN produced better results than RF a root-mean-square error of 4.97 and 6.10 respectively. Nevertheless, it was also observed that both of the ML models are feasible and effective in building energy prediction. In [203], authors demonstrated that the forecasting granularity and one-day-ahead load forecasting accuracy for residential customers can be affected by the calendar variables and the scaling of the training set. Statistical analysis has demonstrated that the regression trees approach significantly surpasses ANN, and SVR techniques despite the similarity of average RMSE for all techniques. Deng et al. [204] perform a comparative analysis of six data mining techniques including SVM and RF for estimating Energy Use Intensity (EUI) for commercial office buildings in the US. They also estimated the plug loads, and lighting loads of HVAC, based on the 2012 CBECS microdata. According to experimental results, SVM and RF provided better predictive accuracy based on a large number of outliers in the CBECS dataset.

Divina et al. [205] performed a comparative analysis to analyze the performance of statistical and ML models in predicting energy consumption in non-residential smart buildings by utilizing the electrical energy consumption data collected from thirteen smart buildings located on a university campus in Spain. The authors demonstrated that highly accurate prediction accuracy can be reached in favor of strategies based on ML approaches and the historical window's optimal size optimization. Wang et al. [206] proposed a RF-based prediction model for predicting hourly building energy. To evaluate the performance of the proposed model, the hourly electricity consumption of two educational buildings in North Central Florida was predicted. The training of the RF was performed by considering different input variables with the intention to search the feature space that has a critical impact on the prediction model's performance. According to experimental results, RF showed better performance in comparison with RT, and SVR models.

Based on the literature findings, combined with the promising results achieved in energy consumption forecasting domain, the ML-based forecasting models are investigated in this work. To the best of author knowledge, the proposed forecasting approach is the first study in which ML forecasting models are investigated on real street lighting data. In addition, in this work, we developed a rule-based predictor for street lighting energy consumption. It is a generic predictor and can be applied on any street lighting energy data. Furthermore, an extensive set of experiments are performed to make a comparative analysis between rule-based predictor and ML forecasting models.

7.4 Overview of the Energy Consumption Forecasting Approach

This section provides an overview of the overall approach we adopted for energy consumption forecasting for the PELL data and street light energy usage data[2]. As discussed earlier, the proposed energy consumption forecasting system is being developed as a part of the PELL SCP [121] (as already shown in Figure 7.1).

As shown in Figure 7.6, the proposed approach consists of three main stages: *data preprocessing*, *data splitting*, and *forecasting model building and evaluation*. All these stages are discussed separately in the following subsections.

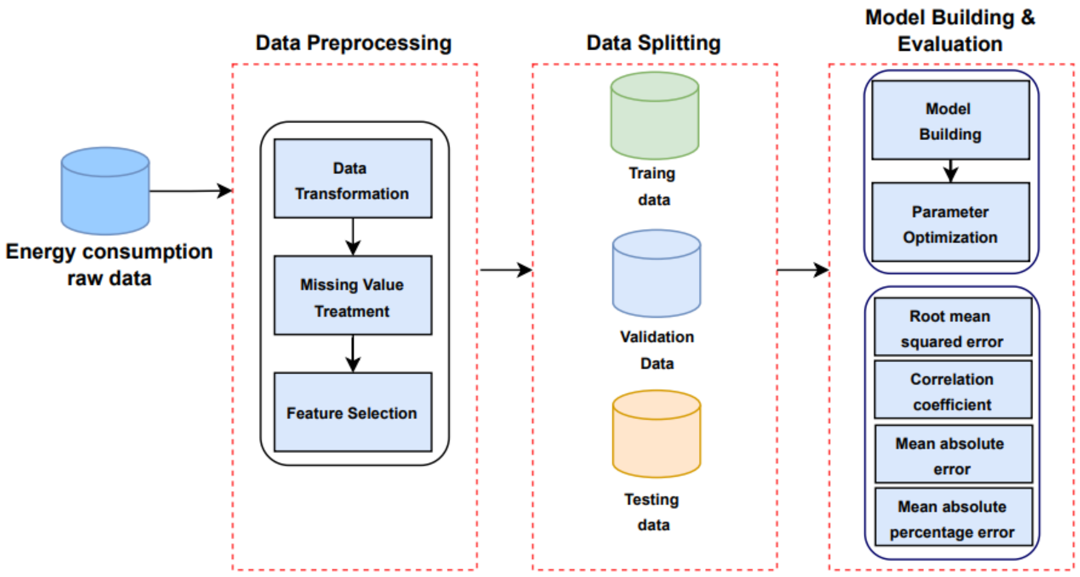


Figure 7.6: Overview of the proposed energy consumption forecasting approach

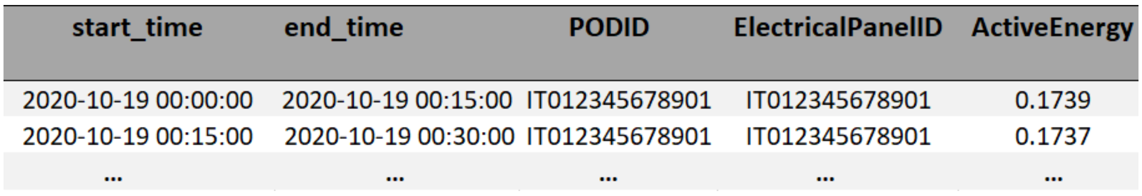
7.4.1 Data Preprocessing

This module is responsible for preparing the data for further processing and analysis. It includes three subsequent phases: data transformation, missing data treatment, and feature selection.

[2]https://data.world/lasvegasnevada/street-light-energy-usage/

7.4.1.1 Data Transformation

The PELL data is retrieved through a JSON broker and represented in the JSON format. In order to process the data for further analysis, it should be transformed into a usable format. Therefore, we transform the electrical measures available in the JSON files into a processable dataframe, and then perform the remaining pre-processing steps to prepare the data for the training and testing of forecasting algorithms. The data set after pre-processing steps looks as shows in Figure 7.7. The PELL street lighting data set contains the following fields.



start_time	end_time	PODID	ElectricalPanelID	ActiveEnergy
2020-10-19 00:00:00	2020-10-19 00:15:00	IT012345678901	IT012345678901	0.1739
2020-10-19 00:15:00	2020-10-19 00:30:00	IT012345678901	IT012345678901	0.1737
...	...	...	...	...

Figure 7.7: PELL Dataset after preprocessing

- start_time: Time stamp indicating the start of the period.
- end_time: Time stamp indicating the end of the period.
- PODID: Representing the unique point of delivery.
- ElectricalPanelID: Denotes the unique electric panel.
- ActiveEnergy: Representing the amount of energy consumed in a specific time interval.

The street light energy usage data[3] is already available in CSV format, so we only read the CSV and transform it into a processable dataframe.

7.4.1.2 Missing Data Treatment

PELL data pre-processing includes dealing with the irregularities in the data records in the form of missing data fields. Two well known approaches to deal with the missing data are the method *moving window* and *inference-based* methods [154]. The moving window method is an improved model of linear interpolation that is easy to implement particularly in the case when the duration of the missing value is

[3]https://data.world/lasvegasnevada/street-light-energy-usage/

brief. Instead, if the duration of missing value is long, then inference-based methods are highly recommended for analysis [155]. In this work, to deal with the missing data, we combined two different strategies: (1) when the number of missing data is less than or equal to 3 as adopted in [155] the moving window method is applied, (2) otherwise, we filled the missing data with the preceding value (the data registered at the same time the day before).

7.4.1.3 Feature Selection

Feature selection is an important step to build the ML models. It has a significant impact on the performance of ML models. The considered PELL street lighting data is a ***nxm*** dimensional data, where n denotes the rows (collected energy consumption measurements) and m represents the columns (m = 5; *start_ time*, *end_ time*, *PODID*, *ElectricalPanelID*, *ActiveEnergy*). The *ActiveEnergy* representing the amount of energy consumed in a specific time interval, and it is expressed in kWh. As a feature selection step, in this study, we used the Autoregressive (AR)(1) Feature of *ActiveEnergy* variable and time feature (hours of the day) from the PELL electricity consumption data to forecast the energy consumption. The time feature (hours of the day) extracted from *start_ time*, and it represents a number from 0 to 23.

Regarding the Street Light Energy Usage[4] data, we consider the same features such as Autoregressive Feature AR(1) of *Usage* variable and time feature (hours of the day) to forecast the street energy consumption.

7.4.2 Data Splitting

The PELL street lighting data contains consumption measures which are collected from 1st September 2020 to 30th November 2020. This consumption is measured with the granularity of fifteen minutes, and each JSON file contains 96 values (4 values per hour and per 24 hours of a day), totaling 8736 data points of three months consumption. The data set is split into training data and testing data, each one comprising 5876 (two months data), and 2880 (one month data).

7.4.3 Model Building and Evaluation

In literature, energy consumption forecasting methods falls into three categories: (1) engineering methods, (2) statistical methods, and (3) AI-based methods [194].

[4]https://data.world/lasvegasnevada/street-light-energy-usage/

In this work, we have used AI-based methods such as LR, SVR, DT, RF, KNN, and MLP. AI-based methods learn consumption patterns from historical energy consumption data, i.e. find out the non-linear relationship between the input (historical data) and output (target consumption) [207, 208]. More details on the configuration and use of these forecasting models are presented in Section 7.5.

In order to evaluate the predictive accuracy of the selected forecasting models, in this analysis, we have used well-known quality metrics: RMSE, R, MAE, and MAPE. More details about our evaluation of the forecasting models by using these metrics are provided in Section 7.6.

7.5 Adopted Forecasting Models

7.5.1 Rule-Based Predictor

In this work, we have developed a novel rule-based predictor for street lighting energy consumption. This model is developed as a baseline predictor for street lighting domain. The energy consumption pattern of street lighting is different than the consumption pattern of buildings. Street lights operate under the control of municipalities by following predefined operating policies and schedule. The street lights operate at night and are switched off during the daytime. The observed energy consumption pattern of street lighting is shown in the Figure 7.8.

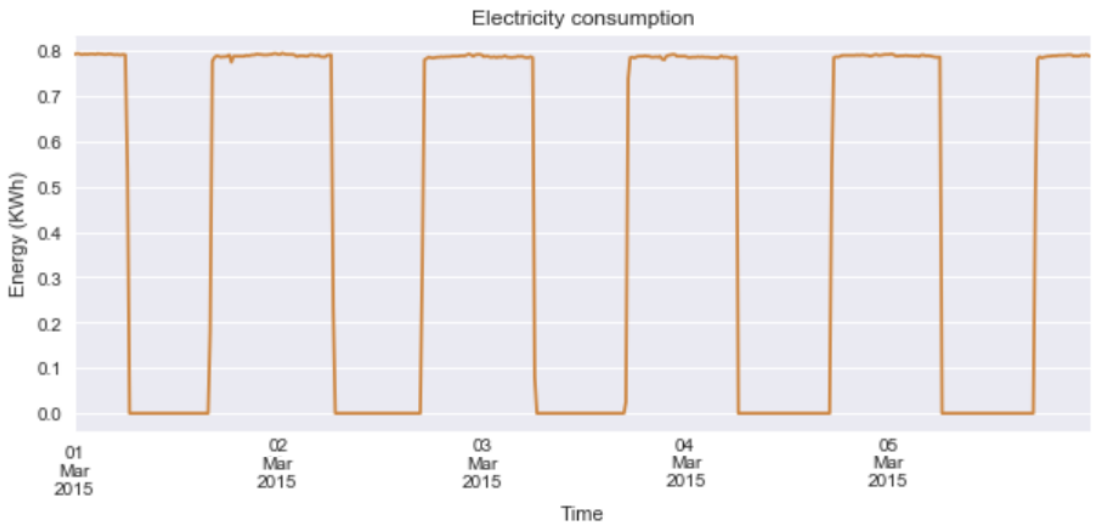


Figure 7.8: Street lighting energy consumption pattern.

In this consumption pattern, we can see that energy consumption is reported at night when the terminal was operating, and in the daytime, we have zero consumption. We developed a rule-based predictor by following this day and night consumption pattern. The rule-based predictor is composed of two modules: training module and prediction module. The working of the RBP requires training on the previous year data and testing should be performed on the next year or current year data of the same time period. The training module of the RBP is based on **Algorithm 3**.

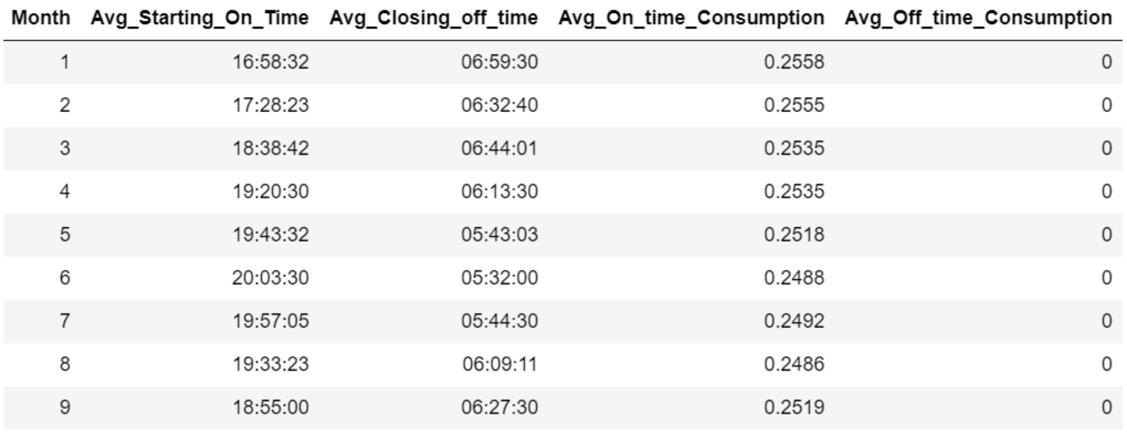
Algorithm 3 Rule-based predictor training module

INPUT: D: a data frame (three-dimensional array such as [Time, Month, EnergyConsumptionValue] with heterogeneous data) containing n instances
OUTPUT: $D^\star = [ASO_NT, ACO_{FF}T, AO_NTC, AO_{FF}TC]$
Method:

```
1:  for $i = 1$ to $len(M)$ do
2:      if $EnergyConsumptionValue > threshold$ then
3:          state ← ON
4:      else
5:          state ← OFF
6:      end if
7:      D.append(state)
8:      on_off_time_list.append($D[time].state$ != $D[time].state.shift()$)
9:      starting_on_time_list.append(on_off_time_list[$state == ON$])
10:     closing_off_time_list.append(on_off_time_list[$state == OFF$])
11:     $ASO_NT$ ← starting_on_time_list.mean()
12:     $ACO_{FF}T$ ← closing_off_time_list.mean()
13:     on_off_consumption_list.append ($D[EnergyConsumptionValue].state$ !=
        $D[EnergyConsumptionValue].state.shift()$)
14:     starting_on_time_consumption_list.append(on_off_consumption_list[$state$
        $== ON$])
15:     closing_off_time_consumption_list.append(on_off_consumption_list[$state$
        $== OFF$])
16:     $AO_NTC$ ← starting_on_time_consumption_list.mean()
17:     $AO_{FF}TC$ ← closing_off_time_consumption_list.mean()
18:     return $D^\star = [ASO_NT, ACO_{FF}T, AO_NTC, AO_{FF}TC]$
19: end for
```

In the algorithm, at step 1, it is checked that whether the value of consumed energy is greater than or less than the threshold value. If the the value of consumed energy is greater then a variable *state* with ON value is added to the dataframe at step 3. If the value is smaller than threshold then variable *state* with OFF value is added to the dataframe at step 5. Then at step 8, on-time and off-time of the electric panel is determined for each day of the month. At step 9, starting on-time of the panel is determined for each day of the month. Similarly at step 10, closing off-time of the panel is determined for each day of the month. Afterward at step 11, average staring on-time (ASO_NT) is calculated by taking the mean of starting on-time of the whole month, and at step 12 average closing off-time (ACO_FFT) is calculated by taking the mean of closing off-time of the whole month. Then at step 13, consumed energy during on-time and off-time is calculated for each day of the training month. At step 14, starting on-time energy consumption is calculated for the whole month, and similarly, at step 15, closing off-time energy consumption is calculated for the whole month. Then at step 16, average on-time consumption (AO_NTC) is calculated for the whole month, and at step 17, average off-time consumption ($AO_{FF}TC$) is calculated for the whole month. Finally, at step 18, training matrix ($D^\star$) is returned.

After the applications of training module, we will have a training matrix ($D^\star$) as shown in Figure 7.9.



Month	Avg_Starting_On_Time	Avg_Closing_off_time	Avg_On_time_Consumption	Avg_Off_time_Consumption
1	16:58:32	06:59:30	0.2558	0
2	17:28:23	06:32:40	0.2555	0
3	18:38:42	06:44:01	0.2535	0
4	19:20:30	06:13:30	0.2535	0
5	19:43:32	05:43:03	0.2518	0
6	20:03:30	05:32:00	0.2488	0
7	19:57:05	05:44:30	0.2492	0
8	19:33:23	06:09:11	0.2486	0
9	18:55:00	06:27:30	0.2519	0

Figure 7.9: An example of RBP training matrix.

Once the training matrix of RBP is computed, we can make prediction for any time period of testing data. We can make prediction for one instance, for one day, for one week, for one month, and also for a range of time periods. The prediction module of the RBP is based on **Algorithm 4**.

The prediction module requires a training matrix $D^\star$, which is the output of the training module, and test data that will be used to make predictions. At step 3, time of the testing instance is determined, and at step 4, month of the testing instance is found. Then at step 5, it is checked whether the testing month matches with the months in the training matrix or not. If the training matrix contains the

Algorithm 4 Rule-based predictor prediction module

INPUT: $D^\star$, test data
OUTPUT: $predictedEnergy \mid time\&month$
Method:

```
1:  for $j = 0$ to $len(D^\star)$ do
2:      for $k = 0$ to $len(testData)$ do
3:          list_of_prediction_time.append(testData.time)
4:          list_of_prediction_month.append(testData.month)
5:          if $D^\star$[Month][j] $==$ $list\_of\_prediction\_month[k]$ then
6:              if $list\_of\_prediction\_month[k]$ in $Range(ASO_NT, ACO_{FF}T)$ then
7:                  predictedEnergy.append($D^\star(AO_NTC)$)
8:              else
9:                  predictedEnergy.append($D^\star(ACO_{FF}T)$)
10:             end if
11:             return $[predictedEnergy \mid time\&month]$
12:         else
13:             Testing month should be in the range of months available in $D^\star$.
14:         end if
15:     end for
16: end for
```

training parameters for the testing month, then at step 6, it is determined whether the time of the testing instance falls in the range of ASO_NT and $ACO_{FF}T$ or not. If testing time exist within the range of ASO_NT and $ACO_{FF}T$, then AO_NTC of the same month available in the training matrix is assigned to the testing instance at step 7, and this is the predicted value. If the time of testing instance does not exist in the range of ASO_NT and $ACO_{FF}T$, then at step 9, $AO_{FF}TC$ available in the training matrix is assigned to the testing instance and this is predicted value. Finally, at step 11, predicted energy for the given time and month is returned.

7.5.2 Linear Regression (LR)

Regression is a statistical method for determining the relationship between two or more variables. It can be of two types, i.e., linear or nonlinear. The relationship is modeled in linear regression by functions that are linear combinations of variables. Nonlinear regression models the relationship using functions of nonlinear variable combinations [209]. In general, we use one or more factors to predict another variable in regression. The most basic type of regression contains two variables: the explanatory or independent variable (typically denoted X) and the response or dependent variable (commonly denoted Y).

Thus, the simplest regression issue has two variables: the variable being predicted (Y), which serves as a response from the model (thus its name), and the explanatory variable (X), which explains the prediction (reason why it is called the explanatory variable). We assume in this scenario that the explanatory variable (X) and the response variable (Y) are connected by [210];

$$\text{“}Y = \beta_0 + \beta_1 X + \varepsilon\text{”} \tag{7.1}$$

In above equation 7.1, β_0 and β_1 are model parameters which are unknown must be estimated whereas ε is error which cannot be modeled. Meanwhile, β_1 gives the slope and direction of the line of regression, whereas, β_0 gives the Y-intercept. We leave out the error term (since it cannot be modeled) and model the relationship as [210];

$$\text{“}\hat{Y} = b_0 + b_1 X\text{”} \tag{7.2}$$

In equation 7.2, $\hat{Y}$ is predicted or theoratical value as an output by the model, whereas, b_0 and b_1 are estimates of β_0 and β_1 respectively. The residual e represents the discrepancy between the model's observed and anticipated values. It is computed as follows [210]:

$$\text{“}e = Y - \hat{Y} = Y - (b_0 + b_1 X)\text{”} \tag{7.3}$$

This means that regression can tell us how much change in one variable we should expect if we modify the other by a specific amount. However, it is unknown what causes the transformation. This implies that regression does not reveal causation. To instance, if one runs a regression on the prices of houses in a neighborhood versus the incomes of home owners, the regression analysis will yield a strong positive coefficient of association; however, this does not imply that high wages create high home prices. The high correlation coefficient and total lack of causation indicate that correlation is concerned with quantifying the strength of the relationship between two variables, whereas regression is more concerned with explaining how much change one can expect in one variable when the other changes by a given amount.

It is critical to understand the distinction between correlation and regression analysis. Regression seeks to find the link between two or more variables, whereas correlation assesses the strength of that association (See Hood and Green (2006)). We tend to choose the line that best suits our data collection because no single straight line can travel through all of the data points in most real-world circumstances. This is known as the "best fit" line. The Ordinary Least Squares (OLS) regression process will determine the optimal values for the parameters $b0$ and $b1$ (the intercept and slope of the regression line, respectively) based on the observations. The residual is the vertical gap between each observation and the line of "best fit"—the regression line.

7.5.2.1 Linear Regression Model Assumptions

To develop a linear regression model using the ordinary least squares technique, certain requirements must be met [211].

- The linear model proposed is the appropriate one. This means that if the model is not correctly specified, the approach may not generate its best estimates of the parameters $\beta_j, j \in 0...n$. We can misspecify the model by leaving out (a) variable(s) that are strongly related to the independent variable. For example, modeling the frequency of a pendulum without taking into account its length.
- The mean error term should remain independent of observed dependent variables. If this is not the case, it is likely that at least one variable is missing from the model and is indicated by the errors.
- The error terms are uncorrelated with one another and have constant variance that is independent of the observed X variables, such as $Var(Y_i|X_i) = \sigma^2$. If this criteria is not met, the error terms will be connected to the independent variables such as $Var(Y_i|X_i) = \sigma_i^2$.
- There is no independent variable that exactly predicts another. When an independent variable predicts another independent variable in a model, the model gives the variable more weight. It would be as if the variable was utilized twice in the model. This restriction could be broken by, for example, putting a person's age and birth date in the same model.

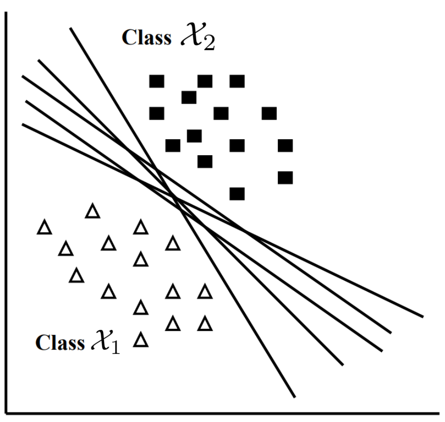
7.5.3 Support Vector Regression (SVR)

SCM! (**SCM!**) was developed by Vapnik et. al. at AT&T Bell labs in 1995 [212, 213, 214] and it has been receiving attention to perform classification and forecasting

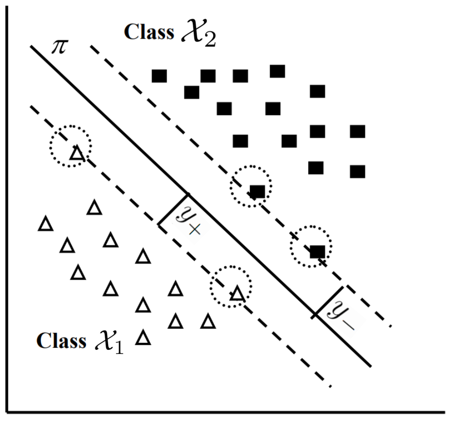
[212, 215, 216, 170] for some time now. It is based on Structural Risk Minimization (SRM) principle [212, 213, 216]. SVMs were first created to handle pattern classification issues such as optimal character recognition (OCR), face recognition, and text classification, among others. However, they quickly found widespread use in other domains, including function approximation, regression estimation, and time series prediction problems [212, 170, 217]. The goal of SVM is to create a decision rule with strong generalization ability by selecting a specific subset of training data known as support vectors [213, 170, 214]. After non-linearly mapping the input space onto a higher dimensional feature space, an optimal separation hyperplane is built. As a result, the quality and complexity of an SVM solution are not directly dependent on the input space [215, 218]. Another distinguishing feature of SVM is that the training procedure is analogous to solving a linearly constrained quadratic programming problem. As a result, unlike other networks, the SVM solution is always unique and globally optimal. However, when the training size is high, SVM demands a significant amount of computing, which raises the time complexity of the solution [212, 156].

When used to perform binary classification problems, the primary principle of SVM is to discover a canonical hyperplane that separates the two provided classes of training data the most [212, 214, 213, 215, 170]. To understand this concept, let us consider two sets of linearly separable training data points in $\mathcal{S}^n$ that must be classified using linear hyperplanes into one of two classes i.e., $\mathcal{X}_1$ and $\mathcal{X}_2$. The one with the greatest margin should be chosen from an infinite number of separating hyperplanes for the best classification and generalization. This concept is visualized in figure 7.10 below:

The training data points are separated by an infinite number of hyperplanes, as shown in Fig. 7.10a. As illustrated in Fig. 7.10b, y_+ and y_- are the perpendicular distances from the separating hyperplane to the nearest data points of $\mathcal{X}_1$ and $\mathcal{X}_2$, respectively. Then for each distance i.e., either y_+ or y_-, it is called margin and the total margin can be computed as $M = y_+ + y_-$. The *Maximum Margin Hyperplane* is regarded the optimal one for accurate classification and best generalization since it maximizes the total margin [213, 214]. And for this hyperplane, we have $y_+ = y_-$ [213]. Furthermore, *Support Vectors* are the data points from either of the two classes that are closest to the maximum margin hyperplane [213, 214]. Meanwhile in Fig. 7.10b, π represents the optimal hyperplane and circulated data points of both classes.



(a) Infinite number of linearly seperating hyperplanes



(b) The maximum margin hyperplane

Figure 7.10: Support vectors for linearly separable data points

To further understand how this work, let us consider a training set composed of the input-output pairs represented as $\{\mathbf{j}_i, k_i\}_{i=1}^{N}$, where $\mathbf{j}_i \in \Re^n$, $k_i \in \{-1, 1\}$. The purpose is to find a maximum margin canonical hyperplane that divides the data points into two classes. The hypothesis space is the set of functions $f(\mathbf{j}, \mathbf{w}, b) = \text{sgn}(\mathbf{W^T}\ \mathbf{j} + b)$ where $\mathbf{w}$ represents weight vector, b represents bias and $\mathbf{j} \in \Re^n$. The set of separating hyperplanes is given by $\{\mathbf{j} \in \Re^n : \mathbf{W^T}\ \mathbf{j} + b = 0$, where $\mathbf{w} \in \Re^n$, $b \in \Re\ \}$. The challenge of determining the largest margin hyperplane is reduced to a non-linear convex quadratic programming problem (QPP) using SVM [213, 214, 219].

7.5.3.1 SVM for Linearly Seperable Data

The equivalent quadratic optimization problem for linearly separable data is [18, 29, 33]:

$$\left.\begin{array}{rl} \text{Minimize} & l(\mathbf{w}) = \dfrac{1}{2}\mathbf{w}^T\mathbf{w} = \dfrac{1}{2}||\mathbf{w}||^2 \\ \text{Subject to} & k_i(\mathbf{w}^T\mathbf{j}_i + b) \geq 1; \forall_i = 1, 2, ..., N \end{array}\right\} \tag{7.4}$$

It is convenient to transform to the dual space to solve the QPP Eq. 7.4. Then, to determine the best solution, Lagrange multipliers and Kühn-Tucker complementary conditions are applied. Assume that the solution to the QPP produces the optimal Lagrange multipliers. $\boldsymbol{\beta} = (\beta_1, \beta_2, ..., \beta_N)^{\mathrm{T}}$ where $\beta_i \geq 0$ ($\forall_i = 1, 2, ..., N$) and the optimal bias b_{opt}. The support vectors are the data vectors for which $\beta_i > 0$, and assume that there are a total of N_s support vectors. The ideal weight vector can then be represented as [213, 214].

$$\mathbf{w}_{opt} = \sum_{i=1}^{N_s} \beta_i k_i \mathbf{j}_i \tag{7.5}$$

The optimal hyperplane decision function is given by [29, 33]:

$$kj = sgn\left(\sum_{i=1}^{N_s} k_i \beta_i (\mathbf{j}^T \mathbf{j}_i + b_{opt})\right) \tag{7.6}$$

Unknown data is sorted into one of two classes based on the sign of $k(\mathbf{j})$.

7.5.3.2 SVM for Non-inearly Seperable Data

In practice, the training data points are frequently not linearly separable; consider the XOR classification problem. In these circumstances, a *Soft Margin Hyperplane* [29] classifier is built. Then the corresponding QPP is given by [213, 214, 215]:

$$\left.\begin{aligned} \text{Minimize} \quad l(\mathbf{w}, \xi) = \frac{1}{2}||\mathbf{w}^2|| + C\left(\sum_{i=1}^{N} \xi_i\right) \\ \text{Subject to} \quad k_i(\mathbf{w}^T \mathbf{j}_i + b) \geq 1 - \xi_i; \forall_i = 1, 2, ..., N \\ \xi_i \geq 0 \end{aligned}\right\} \tag{7.7}$$

Slack variable ξ_i are used to relax the hard-margin requirements, and $C > 0$ is the regularization constant that penalizes misclassification. Similarly, Eq. 7.7 represents a QPP by applying Lagrange multipliers and Kühn-Tucker conditions, its solution can be found. The optimal weight vector and decision function are the same as in the case of linearly separable weights. The sole difference is that the Lagrange multipliers have an upper restriction on C, such as $0 \leq \beta_i \leq C$ and for support vectors $0 < \beta_i \leq C$, in this scenario.

7.5.3.3 Suppoer Vector Kernels

In SVM applications, it is better to map the points of the input space to a high dimensional Feature Space using some non-linear mapping [213, 215, 218], and then construct the best separating hyperplane in this new feature space. This technique also solves the problem of training points that cannot be separated by a linear decision boundary. Because, with the right transformation, the training data points may be separated linearly in the feature space. This concept is visualized in figure 7.11.

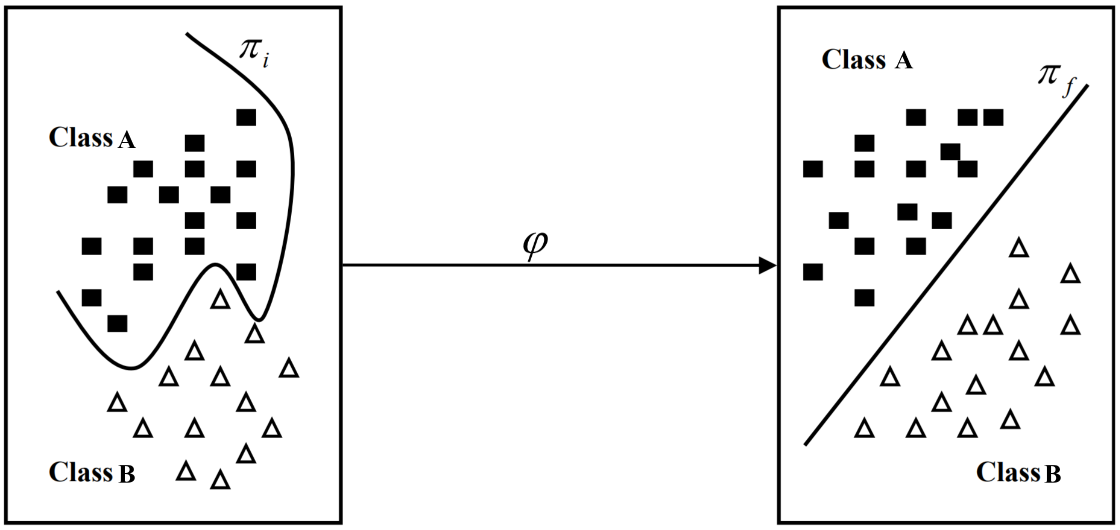


Figure 7.11: Non-linear mapping of input space to the feature space where left hand side is the low dimensional input Space 'X' and right hand side is the high dimensional feature space 'H'.

By utilizing the nonlinear mapping $\varphi : X \longrightarrow H$, the data points of the input space $X \subseteq \Re^n$ are mapped to the feature space H. Because of this transformation, the linearly non-separable input points in the feature space can now be separated by a linear decision boundary π. Meanwhile, the corresponding linear decision function H can be written as [213, 214, 215]:

$$y(\mathbf{j}) = sgn\left(\sum_{i=1}^{N_s} k_i\beta_i(\varphi\mathbf{j}^T\varphi(\mathbf{j}_i) + b_{opt})\right) \tag{7.8}$$

If we are able to find a function such that $K(\mathbf{j}_i, \mathbf{j}_j) = \varphi(\mathbf{j}_i)^T\varphi(\mathbf{j}_j)$, it can be utilized directly in the training equations of SVM without even having the information about the mapping $\varphi(j)$. We will call this function K as *Kernel Function* [213, 215, 216, 217]. The kernel function K must satisfy *Mercer's Condition* [213, 216, 217] to avoid the computation of non-linear mapping $\varphi(j)$. Following are some of the well known kernels used in literature of SVM:

- Linear Kernel [217] where $K(j,k) = j^Tk$
- Polynomial Kernel [212, 217] where $K(j,k) = (1 + j^Tk)^d$

- Radial Basis Function Kernel [213, 217] where $K(j,k) = exp(-||\textbf{j-k}||^2/2\sigma^2)$
- Neural Network Kernel [213] where $K(j,k) = tanh(a\mathbf{j}^T\mathbf{y} + b)$ and *a,b* are constants

7.5.3.4 SVM for Regression (SVR)

We've talked briefly about the SVM techniques used for classification. To apply SVM to regression problems, Vapnik developed a generalized technique [213, 216, 214]. All other variables will have similar meaning in case of classification except the ouputs $k_i \in \Re$.

In SVR, Vapink used the *ϵ-insensitive loss function* defined by [212, 218, 213, 216, 214]:

$$L\epsilon(k, f(\mathbf{j},\mathbf{w})) = \begin{cases} 0 \quad if \quad |k - f(\mathbf{j},\mathbf{w})| \le \epsilon \\ |k - f(\mathbf{j},\mathbf{w})| - \epsilon \quad \text{otherwise.} \end{cases} \tag{7.9}$$

The empirical risk to be reduced is then given as usual by:

$$R_{emp}(f) = \frac{1}{N}\sum_{i=1}^{N} L_\epsilon(k_i, f(\boldsymbol{j}_i, \boldsymbol{w})) \tag{7.10}$$

The linked QPP is denoted as [215, 213, 219, 220, 221]:

$$\left.\begin{aligned} \text{Minimize} \quad l(\mathbf{w},\xi,\xi^*) &= \frac{1}{2}||\mathbf{w}^2|| + C\sum_{i=1}^{N}\left(\xi_i + \xi_i^*\right) \\ \text{Subject to} \\ k_i - \mathbf{w}^T\varphi(\mathbf{j}_i) - b &\le \epsilon + \xi_i; \forall_i = 1,2,...,N \\ \mathbf{w}^T\varphi(\mathbf{x}_i) + b - k_i &\le \epsilon + \xi_i^* \\ \xi_i &\ge 0 \\ \xi_i^* &\ge 0 \end{aligned}\right\} \tag{7.11}$$

Two sets of Lagrange multipliers are used to get the solution of Eq. 7.11 which are $\boldsymbol{\beta} = (\beta_1, \beta_2, ..., \beta_N)^{\mathrm{T}}$ and $\boldsymbol{\beta^*} = (\beta_1^*, \beta_2^*, ..., \beta_N^*)^{\mathrm{T}}$ where $0 \le \beta_i, \beta_i^* \le C$. For support vectors $0 < \beta_i, \beta_i^* \le C$. In the end, the ideal decision hyperplane can be obtained as [212, 214, 216, 218]:

$$k(\mathbf{j}) = \sum_{i=1}^{N_s}(\beta_i - \beta_i^*)k(\boldsymbol{j}, \boldsymbol{j}_i) + b_{opt} \tag{7.12}$$

7.5.4 Decision Tree (DT)

The DT algorithm is introduced in this section for both multidimensional and data series forms. Decision trees are one of the most frequent supervised learning techniques for multidimensional data. The appeal of decision trees stems mostly from their ability to provide predictive models that are both trustworthy and intelligible. [222], [222], and [223] all independently described decision tree induction.

Decision trees' success in the ML community can be linked to four significant advantages over alternative techniques. First, because decision trees are non-parametric, they can simulate arbitrarily complex learning tasks given enough training data [224]. Second, they can handle heterogeneous input vectors with numerical, ordinal, and category values. They are also resistant to noisy, irrelevant, and missing attribute values. Fourth, decision trees in various forms serve as the foundation for several cutting-edge predictive models, including random forest [225], boosting, and, in the context of data series categorization, shapelet trees [225].

A DT is a tree-structured model where each node reflects a split of the input space. Although decision trees can be created for various partitions at each node in the general case, this article focuses on binary decision trees, in which each internal node has exactly two offspring. In a decision tree, all internal nodes t are associated with a splitting criterion S_t , which divides the input space into disjoint subspaces. For binary decision trees, one of the subspaces is made up of input instances that satisfy the criterion, while the other is made up of examples that do not satisfy the criterion. As a result, the root node t_0 partitions X, and X_t is the partitioning at node t for each node. A decision tree's terminal nodes are labeled with the most likely output $\hat{y}_t \in \mathbf{Y}_t$ of the output value, among the examples in $\mathbf{Z}_t$ reaching t. The most likely class label is written as $\hat{y}_t$ $(\mathbf{x})$. In this manner, the terminal nodes are labeled with the best guess among the examples reaching the leaf. Formally, the label of the leaf reached by traversing the tree according to the splits st is the prediction of an instance x.

7.5.4.1 Induction of decision trees

To minimize overfitting, it is probably conceivable for smaller datasets to enumerate all potential decision trees and choose the shortest possible tree that describes the data, as directed by the principle of Occam's Razor [226, 223]. A more typical technique, however, is to greedily learn a sufficiently good proxy by employing one or more of the heuristics given in the literature [227, 228]. [227] developed an impurity measure $I(s,t)$ that evaluates the goodness of a split s at node t, with the assumption that the lower the impurity value, the better the final leaf node $\hat{y}(\mathbf{x})$ predicts the cases $\mathbf{x} \in \mathbf{X}_t$ arriving the node. The goal of the decision tree induction algorithm in this framework is to maximize the impurity decrease measure, i.e., to make each split as pure as possible. The impurity decrease for a binary split is defined by [227] as:

$$\text{“}Im(s,t) = I(t) - \frac{n_t^R}{n_t} I(t^R) - \frac{n_t^L}{n_t} I(t^L)\text{”} \tag{7.13}$$

In equation 7.13, n_t^R and n_t^L represents the number of children on right and left side of node t respectively, whereas, n_t represents the number of instances reaching t and $I(t)$ measures the goodness of a particular node t. When the local class distribution is maximally dissimilar or maximally similar, the most prevalent impurity metrics achieve their minimum and maximum [227]. Both the entropy [228], defined as:

$$\text{“}I_{entropy}(t) = -\sum_{c \in \mathbf{Y}} p(c|t) log_2 p(c|t)\text{”} \tag{7.14}$$

$p(c|t)$ estimates the probability of c based on the class frequencies in $\mathbf{Y}_t$ and Gini index [227] which is defined as:

$$\text{“”}I_{gini}(t) = 1 - \sum_{c \in \mathbf{Y}} p(c|t)^2 \tag{7.15}$$

fulfill these criteria.

Because a decision tree's resubstitution error is a decreasing function of tree depth, it appears that expanding a decision tree to maximum depth is preferable. However, because increasing the model's complexity results in decision trees that collect noise in the training data, it often leads to overfitting. As a result, there is a trade-off between trees that are too deep and trees that are too shallow. The most popular pruning procedures are to cease induction if the number of samples reaching the node falls below a certain threshold, if the depth of the tree exceeds a certain threshold, or if the drop in impurity falls below a certain threshold. Because pruning has been found to be detrimental to random forest performance.

Furthermore, given a numerical attribute X^k, the optimal split s'_k of X^a is X_t's binary partitioning into subsets.

$$\mathbf{X}_{t_L}^{v} = (\mathbf{X}, y)|(\mathbf{X}, y) \in \mathbf{Z_t}, x^k \leq v \tag{7.16}$$

$$\mathbf{X}_{t_R}^{v} = (\mathbf{X}, y)|(\mathbf{X}, y) \in \mathbf{Z_t}, x^k > v \tag{7.17}$$

where v is a numerical value representing the threshold to split. The conventional approach for determining the threshold v is to sort the examples by attribute value and then choose the midpoint between two examples with different output labels to maximize impurity decrease [223].

7.5.5 Random Forest (RF)

Classic decision tree induction algorithms select the best split dimension and location from all candidate splits at each node by greedily optimizing some appropriate quality criterion (e.g., information gain). Individual trees in a random forest are randomized to de-correlate their predictions. The following are the most frequent methods for introducing randomness:

- Bootstrap aggregation, often known as bagging [229], is a technique in which each decision tree is trained on a slightly different training dataset.
- Subsampling the set of candidate splits within each node at random.

A random forest forecast is typically an average of individual tree predictions which can be represented as follows [230]:

$$\text{``}g(x; \{T_m, \theta_m\}_{m=1}^{M}) = \sum_{m=1}^{M} \frac{1}{M} g(x; T_m.\theta_m)\text{''} \tag{7.18}$$

It is also possible to utilize majority voting for classification if the individual trees output discrete class labels rather than probability distributions. While uniform weights ($w_m = M^{-1}$) are commonly used, the weights can also be optimized, for example, by stacking [231].

Using the bias-variance tradeoff, [232] discuss the advantage of random forests over decision trees. Individual decision trees are low in bias but large in variance (as tree induction algorithms produce different trees on slightly different versions of the dataset.) Individual trees in a random forest are randomized in order to decorrelate their forecasts. Individual trees in the forest may be slightly biased as a result of the randomization technique. The second reason is computational:

if the training technique is prone to local optima, the ensemble may provide a better approximation to the genuine unknown function by combining the results of several random searches. The third argument is representational: while decision trees can theoretically represent any function, the greedy training technique limits the useful hypothesis space. While training on finite data with a greedy local search, an ensemble can represent weighted combinations of trees, increasing its effective representational capacity.

Breiman-RF [225] and Extremely randomized trees (ERT) are two popular random forest versions [232]. Breiman-RF employs bagging, and a random k-dimensional subset of the original D features is sampled at each node. ERT selects a k-dimensional subset of the features and then randomly selects one split location for each of the k features (unlike Breiman-RF which considers all possible split locations along a dimension). ERT, unlike Breiman-RF, does not employ bagging.

Because the separate trees do not interact with one other, RF allows them to be trained in parallel. [233] examined a suite of ML algorithms on a variety of datasets and discovered that random forests are regularly among the top-performing techniques. As a result, random forests continue to be one of the most popular black-box prediction methods.

While the random forest architecture is quite strong, it does have a few drawbacks. For starters, random forests do not define predictive uncertainty in a consistent manner. Methods like Gaussian processes, in particular, offer the enticing attribute of increasing uncertainty as we move away from the training data. Predictions from a random forest, on the other hand, can be overly confident even in locations where no training data has been observed. The major difference is that Gaussian processes are probabilistic, but random forests are not. In a probabilistic framework, we first provide a prior that represents our uncertainty about the underlying function's parameters, and then introduce a likelihood function that quantifies how well the parameters explain the observed training data. Finally, we use the Bayes theorem to obtain the prediction posterior. The observed data constrains the function by down-weighting improbable parameters, resulting in a less uncertain predicted posterior in regions near to the observed training data. However, in regions remote from the observed training data, the observed training data does not constrain the function, so the predictive posterior drops to the prior distribution and exhibits more uncertainty[230].

Another downside of random forests is that they are not well suited for incremental or online learning scenarios in which additional data points are observed on the fly (unlike the batch learning setting where the training dataset does not grow with time.) Stochastic gradient techniques are frequently used to train large-scale ML systems with streaming data. Random forests with hard splits, on the other hand, are not receptive to gradient-based updates. Because splits in decision trees are difficult to undo, contemporary online random forests wait until they have seen enough data to reliably decide the split. As a result, they are significantly less data efficient than the corresponding batch random forest.

7.5.6 k-Nearest Neighbours (KNN)

The KNN) is based on the premise that if the majority of the k closest samples in a feature space belong to a given category, and the sample also belongs to this category and has the features of the samples in this category. To determine the category of the sample to be classed in a classification decision, the KNN) approach simply relies on the category of one or many nearby samples. Fix and Hodges Jr [234] devised the KNN), which was later formalized for classification tasks by Cover and Hart [235]. In general, the KNN) is a modified version of an instance-based learning method that is based on the differences between features in a labeled dataset. It looks for a set of k samples that are the closest to unknown samples based on distance functions. A tagged dataset bunch is utilized in this case to locate the k most similar examples that are closest to the new data point. As a result, the algorithm predicts the new incoming data based on how similar they are to the training observations. This algorithm keeps the entire training set during the learning phase.

From these k samples, the label (class) of unknown samples (the new input data) are compared to each instance in training set and determine the prediction of that unknown samples by calculating the average of the response variables [236]. Therefore, for a regression case, this can be the mean output variable, and for the classification task, it can be the most common class value. KNN) model employs the k closest instances in the training set to compute $\hat{Y}$ for a given instance X, and the prediction is the average of the corresponding targets. The model can be written in the most basic form:

$$\hat{Y}(x) = \frac{1}{k} \sum_{x_i \in N_k(x)} y_i \tag{7.19}$$

Where $N_k(x)$ denotes the collection of nearest points in the x_i training sample.

Determining the k value is tricky because as the parameter k approaches 1, the error on the training set decreases while the error on the test set increases. This is owing to the fact that the KNN) model has a low bias and a high variance. The similarity metric (certain distance functions) is used to determine the closeness and quantify the distance d between two data points. The most commonly used function in this context is Euclidean distance, which measures the distance between x and y points as follows:

$$d(x, y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \tag{7.20}$$

The fundamental advantage of KNN) for classification is that it is non parametric [234], which means that we do not need to know the distribution of data. Because it is based on distances, it is multiclass, it does not assume linear separability of the data [234, 237], it is very stable (i.e., small changes in the training data do not result in significantly different classification results [229]), it can learn from a small set of objects, and it can incrementally add new information and provide competitive performance [238].

The drawbacks of KNN) include that it does not perform well if the classes are unbalanced, that is, if the quantity of objects in the training classes varies greatly from one class to the next, because it increases the likelihood of discovering nearest neighbors belonging to the class with the most objects. Also, it is sensitive to the k value, which must be optimized. Although several probabilistic approaches are known for KNN), they are not used to provide the reliability of the classification for a particular object [239]. Another significant disadvantage of the KNN) approach is the *"curse of dimensionality"*, which proposes the peaking phenomena, which states that for a constant number of objects, the peak of classification accuracy drops as the number of variables grows [240, 241, 242]. This can be avoided by employing a high number of objects or lowering the data's dimensionality [243, 244, 245].

7.5.7 Multilayer Perceptron (MLP)

From the input layer to the output layer, feedforward neural networks process data in a single direction. This implies that the activation signals always go ahead during each training cycle and never go back to nodes they have already encountered [246]. Additionally, while utilizing feedforward neural networks, it is expected that every occurrence of the input data is independent of every other instance, i.e., that every training data point is handled separately of every other training data point. This is a presumption because the mass and temperature of the leftover steel from the prior batch impact the present batch of steel.

MLP consists of one or more hidden layers between the input and output layers. However, the complexity of the model also means that the weight-update procedure gets more complicated. An effective weight-update technique is called the back-propagation algorithm. The back-propagation algorithm enables the MLP to update all its weights with the condition of minimizing the output error. A commonly defined error function is the sum of squared errors shown in Eq. 7.21.

$$E = \frac{1}{2}\sum_{i=1}^{n} \|\mathbf{p}_i - \mathbf{t}_i\|^2 \tag{7.21}$$

where $\mathbf{p}_i$ represents the value of predicted output and $\mathbf{t}_i$ represents the value of target output of the training set instance i.

The chain rules of differential calculus may be used to determine the derivation process since the error function, E, depends on an intricate web of activation functions. Only if the activation functions are continuous and differentiable does this hold true. Finding the minimum of the error function by permitting $\nabla E = 0$ requires computing the gradient in Eq. 7.22 for all weights, l, in the network [247].

$$\nabla E = \left(\frac{\partial E}{\partial w_1}, \frac{\partial E}{\partial w_2}, \dots, \frac{\partial E}{\partial w_l}\right) \tag{7.22}$$

Meanwhile, weight w_i is updated using the increment mentioned in Eq. 7.23.

$$\Delta w_i = -\gamma \frac{\partial E}{\partial w_i} \tag{7.23}$$

Whereas, γ is a constant that represents the learning rate to decide the step lengths towards the minimum for each iteration. It is possible to choose how quickly the network should converge toward a solution, or an error minimum, by altering the learning rate constant. A modest learning rate constant is preferable to a big learning constant if the error-space is very irregular [247].

Below is a presentation of the back-propagation algorithm for an MLP [247].

- *Feed-forward computation*: The input layer receives an input vector x_i with an arbitrary number of values (dimensions). Each node in the input layer as well as the subsequent nodes in the hidden layers assess the activation function and its derivatives with respect to each input value. Each node's output values and derivatives are maintained. The network can accurately calculate the weight changes in the back-propagation stage thanks to the derivatives and output values.

- *Back-propagation to the output layer*: The output nodes are sent the constant 1 to start the back propagation. The derivative values in the initial stage of the back-propagation computation are not changed by a multiplication by unity. The output nodes' stored derivatives are then used to execute the network backward. Each output node undergoes this process, and node j generates a backpropagated error value, δ_j.

- *Back-propagation to the hidden layers*: There are connections between every hidden node in the current layer and nodes in the previous layer and the successor layer. Here, the letters s, c, and p stand for the successor node, the current node, and the predecessor node, respectively. Every backpropagated error value from the successor nodes, δ_s, must be taken into account when computing the backpropagated error for each hidden layer. Consequently, the backpropagated error for current node c is computed as:

$$\delta_c = o_c(1 - o_c) \sum_{s=1}^{S} w_{cs} \delta_s \tag{7.24}$$

where o_c denotes the output value obtained after the feed-forward step evaluation of the current node. The link between the successor node, s, and the current node, c, uses the weight, w_{cs}. The partial derivatives can be derived as follows by utilizing the backpropagated error, c:

$$\frac{\partial E}{\partial w_{pc}} = \delta_c o_p \tag{7.25}$$

where o_p is the input value that the preceding node provided, and p is both the predecessor node's output value and its input value.

- *Weight updates*: The weights are modified in the direction of the negative gradient when all the nodes' backpropagated error values have been determined.

$$w_{pc}^{i+1} = w_{pc}^{i} - \gamma \delta_c o_p + \alpha w_{pc}^{i} \tag{7.26}$$

After calculating all backpropagated error values, it is crucial to update the weights. Otherwise, the weight updates do not match the current iteration's gradient direction.

- *Stopping criterion*: When the difference between the error value's prior and current values is less than a certain threshold.

7.6 Experimental Evaluation

In this section, we report on the empirical evaluation of the proposed energy consumption forecasting approach on the PELL data and street light energy usage data[5]. We present the design of the evaluation (Section 7.6.1), including a description of the PELL data and street light energy usage data, and the evaluation metrics (Section 7.6.1) adopted to evaluate the performance of the forecasting models. Finally, we present the results of our empirical investigation on the both the data sets.

7.6.1 Design of the Evaluation

To forecast the energy consumption from time series energy data, in this work, we have used PELL street lighting data and energy usage data as discussed in Section 7.2.1 and Section 7.2.2 respectively. The PELL data is composed of electrical energy consumption measures collected by smart meters installed into the electrical panels located at Casaccia, Rome, Italy. We have used electric consumption measures which are collected from September 2020 to November 2020. Whereas, street light energy usage data[6] is composed of energy usage measures collected by fifteen smart meters from the streets of city of Las Vegas, USA. For this data, we have used the energy usage measures which are collected from October 2014 to October 2016. Figure 7.2 and Figure 7.4 show examples of electricity consumption series of both the experimental data sets.

7.6.2 Evaluation Metrics

To evaluate the predictive accuracy of the used forecasting models, we used four well known error metrics such as R, RMSE, MAE, and MAPE.

[5]https://data.world/lasvegasnevada/street-light-energy-usage/
[6]https://data.world/lasvegasnevada/street-light-energy-usage/

1. Correlation Coefficient (R): It measures the linear dependencies between the actual y_t and predicted value $\hat{y_t}$. Equation 7.27 presents the formula to calculate R.

$$R = \frac{n \sum y_t.\hat{y_t} - (\sum y_t)(\sum \hat{y_t})}{\sqrt{n(\sum y_t^2) - (\sum y_t)^2}\sqrt{n(\sum \hat{y_t}^2) - (\sum \hat{y_t})^2}} \tag{7.27}$$

2. Root Mean Squared Error (RMSE): It is the standard deviation of the differences between actual and the predicted value. It promotes a heavier penalty on higher errors and also presents the results on the same scale as the data. Equation 7.28 presents the formula to calculate RMSE.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y_t} - y_t)^2} \tag{7.28}$$

3. Mean Absolute Error (MAE): It measures the absolute difference between the actual and the predicted value. Equation 7.29 presents the formula to calculate MAE.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_t - \hat{y_t}| \tag{7.29}$$

4. Mean Absolute Percentage Error (MAPE): It is the measure of the amount of deviation of the predicted value from the actual value. Equation 7.30 presents the formula to calculate MAPE.

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_t - \hat{y_t}}{y_t}\right| \tag{7.30}$$

7.6.3 Results on PELL Data

In this section, we present experimental results of forecasting models with their exploratory settings on PELL data. We have performed the implementation of all the selected forecasting models in the Python programming language by utilizing scikit-learn[7] ML library.

[7]https://scikit-learn.org/stable/

7.6.3.1 Naive Regressor Results

The very first model that is used as a baseline model is the Naive Predictor (NP). The Naive method simply assumes that the energy consumption of the desired day will equal to the previous day:

$$P(t) = (t - \delta t) \tag{7.31}$$

where $\delta t = 24\text{h}$

This approach is only applicable to make a raw prediction. In our work, we made forecasting for PELL data by using fifteen minutes granularity and therefore, in this case $\delta t = 15\text{min}$. Table 7.1 shows the forecasting errors of NP with $\delta t = 15\text{min}$.

Table 7.1: Naive predictor forecasting errors.

Model	$\boldsymbol{\delta t}$	**RMSE**	**MAE**	**MAPE**	**R**
NP	15min	0.0191	0.0055	0.4955	0.9530

7.6.3.2 LR Forecasting Results

LR model tries to model the relationship between a scalar dependent and one or more independent variable [189] (details in sec. 7.5.2). LR models have been reported in literature to predict the electricity consumption for public lighting data because of their ease of use. In this work, we have used two different versions of the LR model. The only difference in both versions is which features are used as input to train the model. The first version contains the HC with autoregressive AR(1) value, and the second version of the LR model employs more sophisticated features. In addition to HC, the second version of the model also contains time information (hour of the day). Table 7.2 shows the forecasting errors obtained by using two different versions of linear regression models.

Table 7.2: Linear regression forecasting errors.

Model	**Input Features**	**RMSE**	**MAE**	**MAPE**	**R**
LR	HC	0.0176	0.0058	0.5387	0.9721
LR	HC & time (hour)	0.0174	0.0055	0.4926	0.9726

The results in Table 7.2 demonstrate that the LR model has produced better forecasting results than the naive predictor. The prediction errors show that the addition of time feature (hour of the day) does not significantly improve the forecasting accuracy. Figure 7.12 and Figure 7.13 shows the comparison of actual and predicted energy consumption values of the LR model for both the used versions.

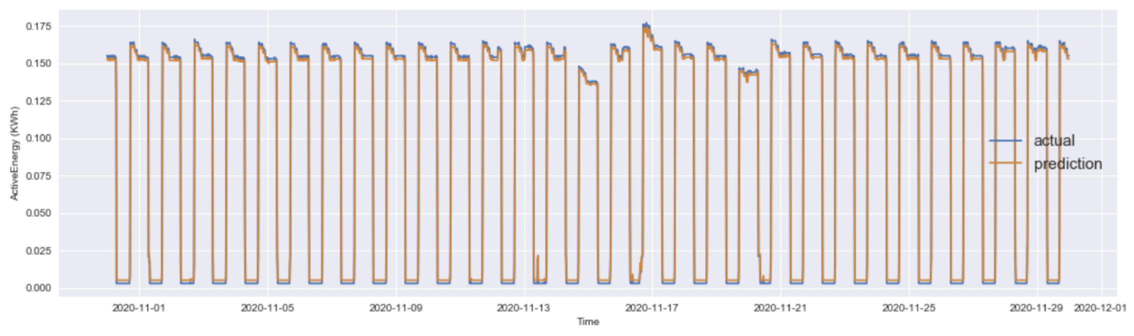


Figure 7.12: Actual and predicted energy consumption forecasting of LR with HC feature.

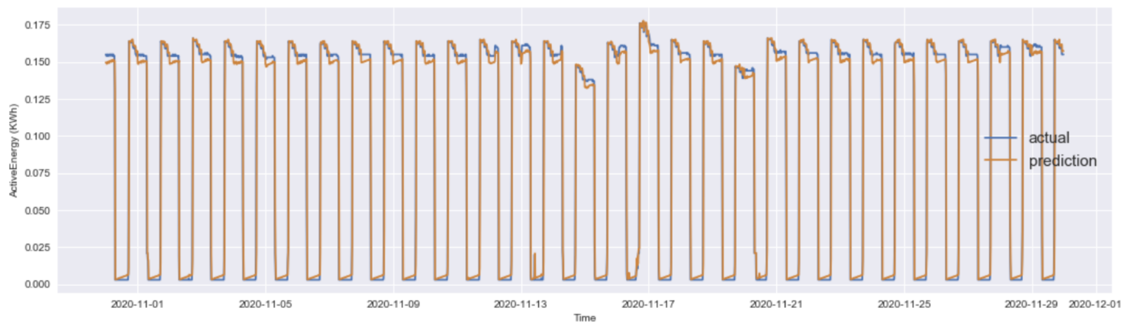


Figure 7.13: Actual and predicted energy consumption forecasting of LR with HC and time feature (hour of the day).

7.6.3.3 SVR Forecasting Results

The SVR is the conversion of Support Vector Machine (SVM), it uses the same principle as the SVMs. It is used to solve the regression problems [189] (details in sec. 7.5.3). Similar to LR model, in this work, we used two different versions of SVR model. In the first version, we trained the model by using HC with AR(1) value. In the second version, we added the time information (hour of the day) to train the model. The performance of the SVR model is dependent on the configuration settings of the model. For this, we used grid search method to select the optimal configurations for the SVR. It is a commonly used approach for hyper-parameter tuning. It attempts to methodically build and evaluate a model for each possible combination of algorithm parameters provided in a grid. In Table 7.3, we provided the used configurations selected by using grid search approach. Table 7.4 shows the forecasting results achieved by using two different versions of SVR model.

Table 7.3: Parameters configuration of SVR model.

Parameters	**Used value**
Kernel	RBF
gamma	0.5
C	5
epsilon	0.001

Table 7.4: SVR forecasting errors.

Model	**Input Features**	**RMSE**	**MAE**	**MAPE**	**R**
SVR	HC	0.0174	0.0041	0.4365	0.9721
SVR	HC & time (hour)	0.0151	0.0037	0.3039	0.9795

The results in Table 7.4 demonstrate that SVR model produced better forecasting results as compared to naive predictor and LR. We can see an improvement in the forecasting quality, especially in the case when we trained the model by using HC and time feature (hour of the day). Figure 7.14 and Figure 7.15 shows the comparison of actual and predicted energy consumption values of the SVR model for both the used versions.

7.6.3.4 KNN Forecasting Results

K-nearest neighbors is one of the straightforward regressor (details in sec. 7.5.6). The performance of KNN model is strongly dependent on the number of neighbors to be used for prediction. This value is represented by K. The performance of the model can be improved by carefully selecting the optimal K value. In this work, we considered different K values and observed better forecasting results with K value to 1. Similar to earlier models, we trained two different versions of KNN model. In the first version, we trained the model by using HC with AR(1) value and in the

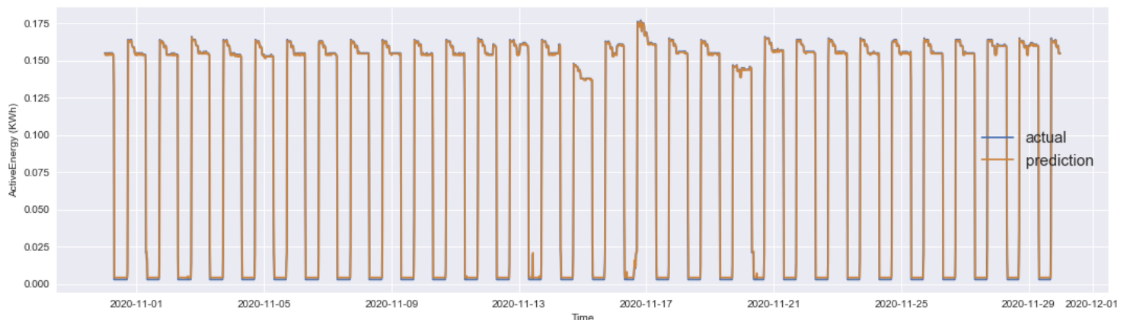


Figure 7.14: Actual and predicted energy consumption forecasting of SVR with HC feature.

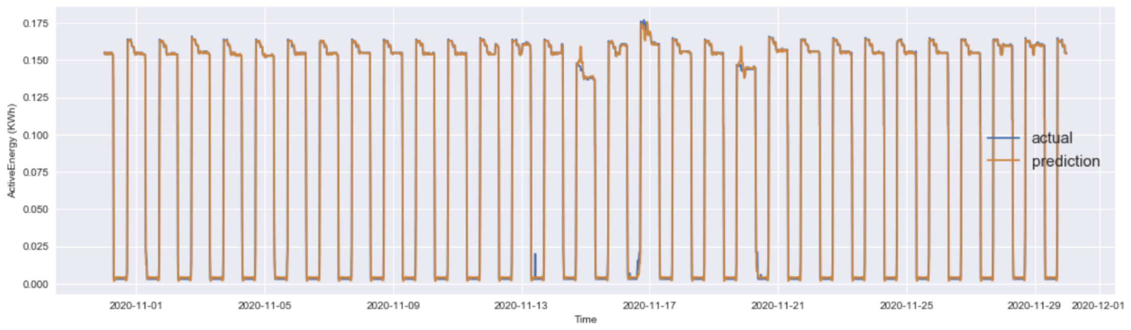


Figure 7.15: Actual and predicted energy consumption forecasting of SVR with HC and time feature (hour of the day)

second version we added time information (hour of the day) with the HC. Table 7.5 shows the forecasting results obtained by using two different versions of KNN regressor.

Table 7.5: KNN forecasting errors.

Model	**Input Features**	**RMSE**	**MAE**	**MAPE**	**R**
KNN	HC	0.0222	0.0046	0.2932	0.9566
KNN	HC & time (hour)	0.0110	0.0021	0.0761	0.9892

Results in Table 7.5 show that the forecasting errors of the KNN model are even higher than the NP when we made predictions by training the model on historic consumption as a feature. But on the other hand, we can observe a significant improvement in the forecasting quality when we trained the model by using both the HC and time information (hour of the day). Figure 7.16 and Figure 7.17 shows the comparison of actual and predicted energy consumption values of KNN model with both the used versions.

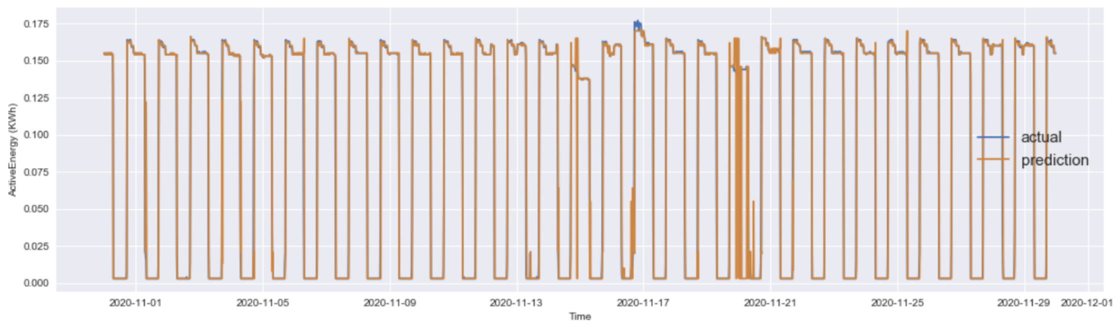


Figure 7.16: Actual and predicted energy consumption forecasting of KNN with HC feature.

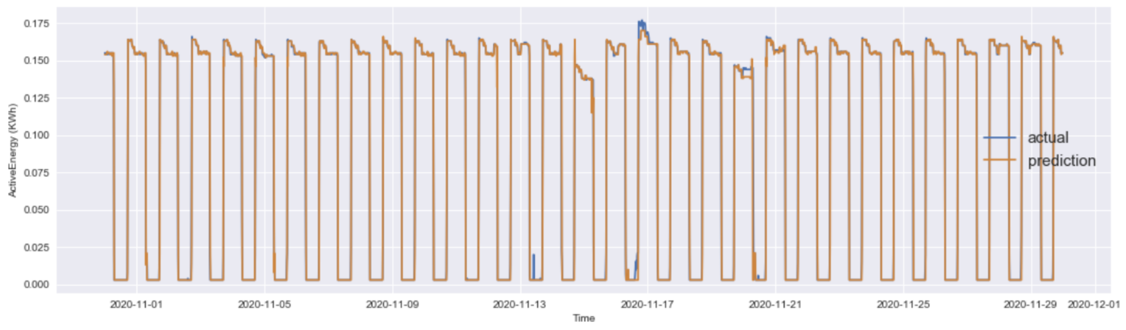


Figure 7.17: Actual and predicted energy consumption forecasting of KNN with HC and time (hour of the day) features.

7.6.3.5 DT Forecasting Results

The DT is one of the most commonly used model in ML [223]. It divides the data into categories by using a tree like flowchart (details in sec. 7.5.4). Similar to the earlier models, we trained two different versions of the DT model. In the first version, model is trained by using HC with AR(1) value, and in the second version time information (hour of the day) is added with the HC. The performance of DT model is extremely dependent on the configuration of the model. Therefore, in this work we used grid search method to select the optimal parameters settings. Table 7.6 presents the parameters for DT model that are selected by using grid search method. Table 7.7 shows the forecasting results obtained by using two different versions of DT regressor.

Table 7.6: Parameters configuration of DT model.

Parameters	Used value
criterion	mae
splitter	best
max_depth	5
min_samples_split	300
min_samples_leaf	5

Table 7.7: DT forecasting errors.

Model	Input Features	RMSE	MAE	MAPE	R
DT	HC	0.0185	0.0037	0.3486	0.9694
DT	HC & time (hour)	0.0153	0.0031	0.2319	0.9791

The results in Table 7.7 demonstrate that an improvement is observed in the forecasting quality when we trained the model by using HC and time feature (hour of the day). Figure 7.18 and Figure 7.19 shows the comparison of actual and predicted energy consumption values of the model for both versions.

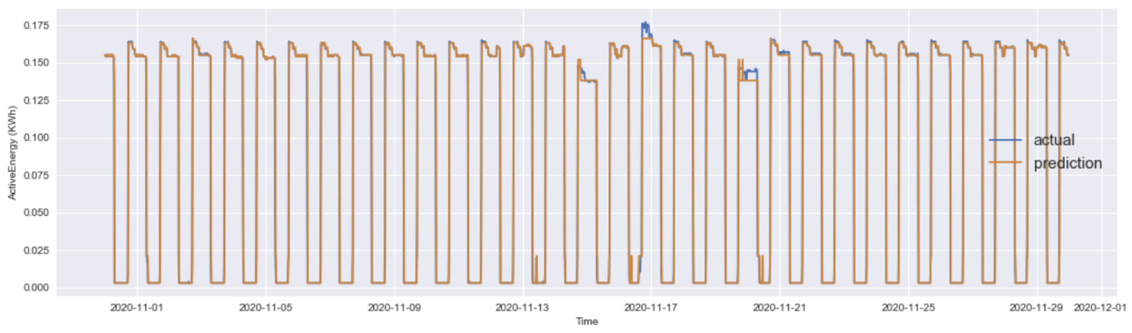


Figure 7.18: Actual and predicted energy consumption forecasting of DT with HC feature.

7.6.3.6 RF Forecasting Results

RF is an ensemble model that consists of a bagging framework and an independent decision tree. In ensemble learning, models combine the output from various models. An ensemble model typically produces better results than a single model(details in

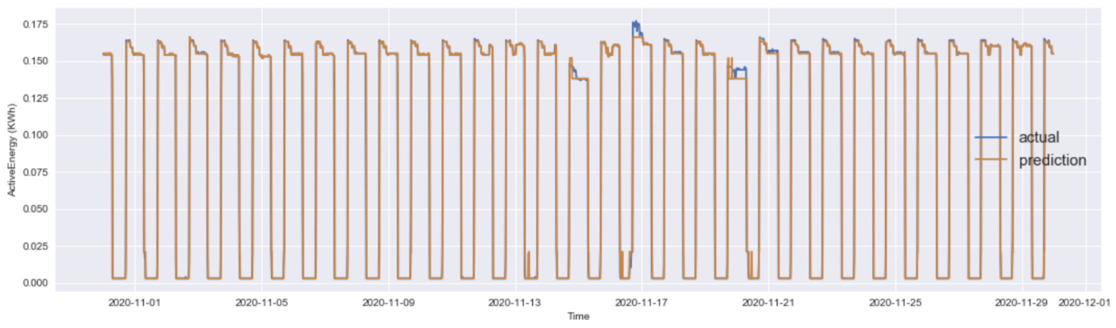


Figure 7.19: Actual and predicted energy consumption forecasting of DT with HC and time (hour of the day) feature.

sec. 7.5.4). To evaluate the effectiveness of the random forest model, we used two different versions of the RF model. In the first version, model is trained by using HC with AR(1) value. In the second version, model is trained by employing HC and time information(hour of the day). The performance of RF model is very sensitive to the model configuration. Therefore, in this work we used grid search approach to select the optimal parameters to configure the model. Table 7.8 shows the parameters for RF that are selected by using grid search method. We only tune two hyperparameter such as number of trees and number of maximum depth. Table 7.9 shows the forecasting results obtained by using two different versions of RF regressor.

Table 7.8: Parameters configuration of RF model.

Parameters	**Used value**
max_depth	10
n_estimators	300

Table 7.9: RF forecasting errors.

Model	**Input Features**	**RMSE**	**MAE**	**MAPE**	**R**
RF	HC	0.0163	0.0041	0.3378	0.9760
RF	HC & time (hour)	0.0105	0.0022	0.0837	0.9902

The results in Table 7.9 show that RF produced improved forecasting accuracy results than the earlier discussed models. We can see that forecasting errors are significantly improved by using the time information with the HC. Figure 7.20 and Figure 7.21 shows the comparison of actual and predicted energy consumption values of the model for both versions.

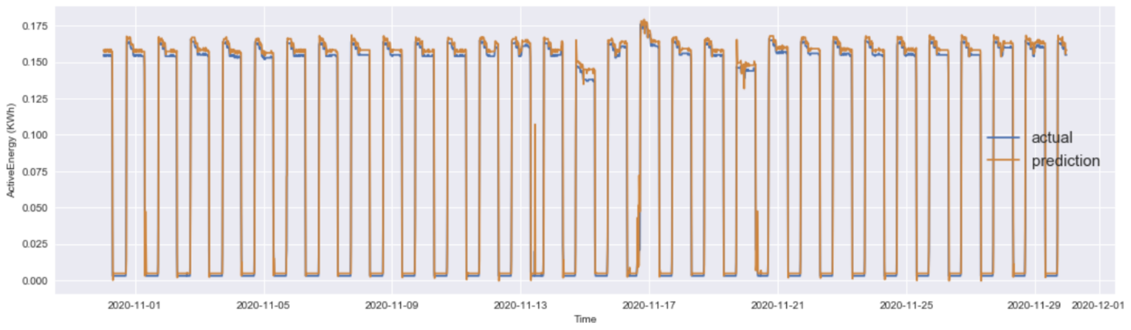


Figure 7.20: Actual and predicted energy consumption forecasting of RF with HC feature.

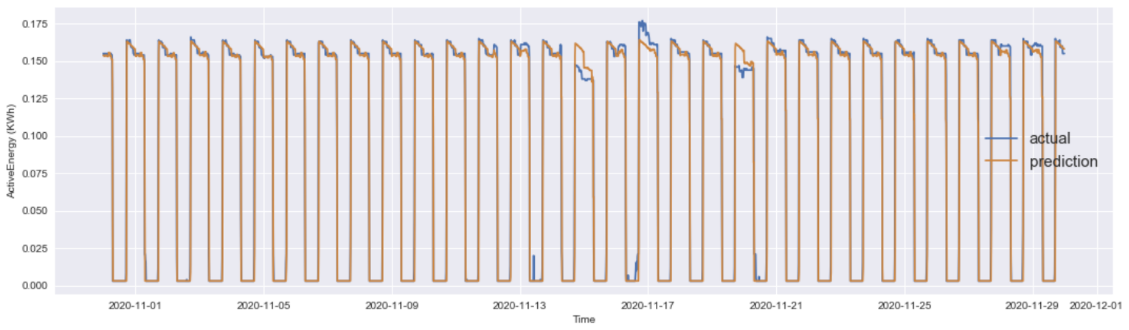


Figure 7.21: Actual and predicted energy consumption forecasting of RF with HC and time (hour of the day) feature.

7.6.3.7 MLP Forecasting Results

MLP is a type of ML algorithm that is a feedforward neural network architecture with an input layer, one or more hidden layers, and output layer [193]. Working of MLP consists of two phases: 1) the training phase of the model in which weights of the connections between neurons are optimized by employing various algorithms, for example, backpropagation combined with stochastic gradient descent (SGD) [248] to minimize a loss function, and 2) the testing phase in which the trained MLP model is applied on the unseen data to fulfill the desire objective (details in sec. 7.5.7). To evaluate the forecasting performance of the MLP model, we used two different combinations of feature sets: HC with AR(1), and HC with time information (hour of the day). In order to determine the optimal parameter settings for the MLP model, we employed grid search. Table 7.10 presents the parameter settings used in this empirical evaluation, and Table 7.11 shows the forecasting results obtained by using two different versions of MLP.

Table 7.10: Parameters configuration of MLP model.

Parameters	**Used value**
Hidden Layers	2
Activation Function	Relu
Optimizer	Adam
Batch size	16
Epochs	150

Table 7.11: MLP forecasting errors.

Model	**Input Features**	**RMSE**	**MAE**	**MAPE**	**R**
MLP	HC	0.0121	0.0051	0.3729	0.9889
MLP	HC & time (hour)	0.0127	0.0043	0.3696	0.9856

The results in Table 7.11 show that MLP produced improved forecasting results especially in the case when we evaluated the model by using historic consumption. On the other hand, it can be seen that the addition of time information does not have a significant impact on the performance of the MLP model as shown in 7.11. Figure 7.22 and Figure 7.23 shows the comparison of actual and predicted energy consumption values of the model for both versions.

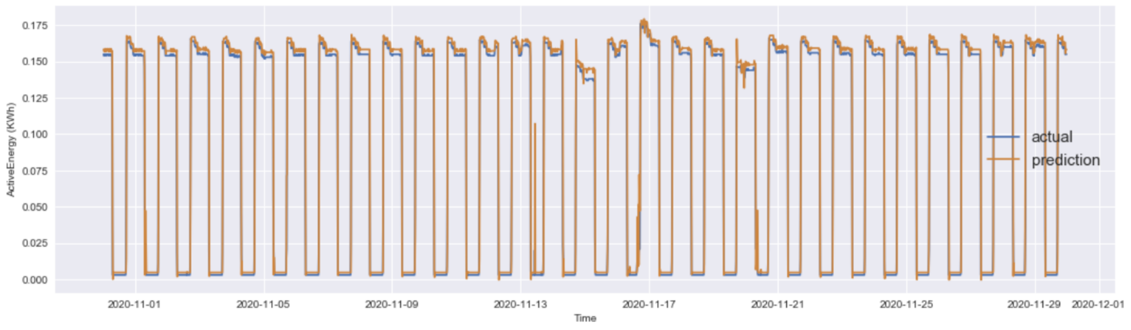


Figure 7.22: Actual and predicted energy consumption forecasting of MLP with HC feature.

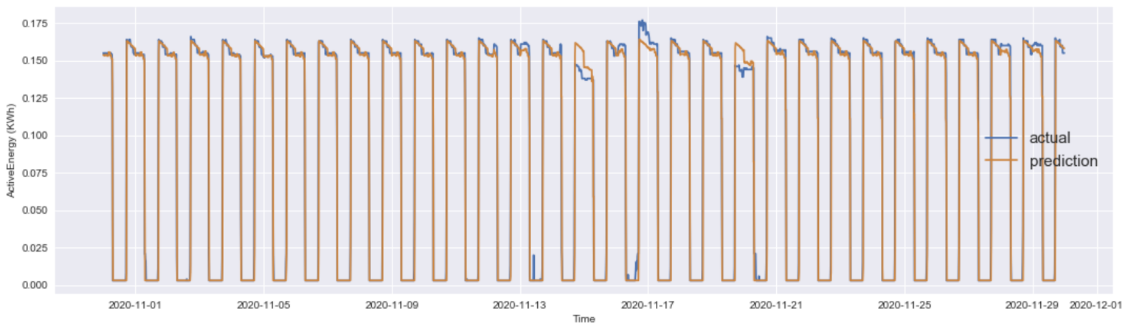


Figure 7.23: Actual and predicted energy consumption forecasting of MLP with HC and time (hour of the day) feature.

7.6.4 Analysis and Discussion

In this empirical investigation, we evaluated the performance of selected forecasting models on PELL street lighting data. We performed forecasting by considering fifteen minutes granularity and different well-known evaluation metrics are used to measure the forecasting errors. In this investigation, we configure two different versions of each forecasting model. In the first version, we used HC to train the model, and in the second version, HC with time information (hour of the day) is used for model training. In the experimental evaluation, it is observed that the forecasting accuracy of the models is improved by training the models on HC and time information features. We noted that the accuracy of the MLP model almost similar in both the configured versions. It is important to note that the forecasting results of KNN and RF are significantly improved by providing the time information in addition to HC. In conclusion, we can see from the results in Table 7.12, RF outperformed its counterpart models.

7.6.5 Results on Energy Usage Data

In this section, we present experimental results of forecasting models with their exploratory settings on Energy Usage Data[8]. It contains two years (from Oct 2014 to Oct 2016) worth of energy consumption measures of 15 street light smart meters (details in sec. 7.2.2). We used two smart meters (MID:CC030050742, MID:CC029790952) for empirical evaluation. For this data set, we applied the same forecasting models that we applied on the PELL data. Design and evaluation of the

[8]https://data.world/lasvegasnevada/street-light-energy-usage/

Table 7.12: Performance comparison of the used forecasting models on PELL data

Model	**Input Features**	**RMSE**	**MAE**	**MAPE**	**R**
NP	HC	0.0191	0.0055	0.4955	0.9530
LR	HC	0.0176	0.0058	0.5387	0.9721
	HC & time (hour)	0.0174	0.0055	0.4926	0.9726
SVR	HC	0.0174	0.0041	0.4365	0.9721
	HC & time (hour)	0.0151	0.0037	0.3039	0.9795
KNN	HC	0.0222	0.0046	0.2932	0.9566
	HC & time (hour)	0.0110	0.0021	0.0761	0.9892
DT	HC	0.0185	0.0037	0.3486	0.9694
	HC & time (hour)	0.0153	0.0031	0.2319	0.9791
RF	HC	0.0163	0.0041	0.3378	0.9760
	HC & time (hour)	0.0105	0.0022	0.0837	0.9902
MLP	HC	0.0121	0.0051	0.3729	0.9889
	HC & time (hour)	0.0117	0.0046	0.2016	0.9884

forecasting models is also kept same. Data splitting for this data is different from what we used for PELL data. The objective of this empirical evaluation is to train the model on every single month (from Jan to Sep) of year 2015 and test the performance on the same month of year 2016. The meter with ID: CC030050742 contains more outliers than the meter with ID: CC029790952. We applied two versions of each forecasting models on Energy Usage Data as we applied on PELL data. In the first version, we trained each model by using HC, and in the second version, time information (hour of the day) is added with the HC to train the model. It is important to note that, we used this data set without applying preprocessing step, because our objective is to observe the performance of forecasting models on raw data. In the following sections, we presented the forecasting results of each model.

7.6.5.1 Rule-Based Predictor RBP Results

In this work, we developed a baseline rule-based predictor for street lighting energy consumption (details in sec. 7.5.1). It is a generic predictor and can be applied to street light energy data set. The primary objective to develop this model is to compare the performance of ML models with a simple rule-based model. We trained the proposed rule-based model on every month (Jan to Sep) of the year 2015 and calculated the forecasting errors for the same month of year 2016. In experimental evaluation, it is noted that RBP produced higher forecasting errors in the presence of outliers. It is obvious from the results in Table 7.13 when there are outliers we obtained higher forecasting errors than the results in Table 7.14 when we have fewer outliers available in the data. Figure 7.24 and Figure 7.25 shows the comparison of actual and predicted energy consumption of RBP model for both the smart meters.

Table 7.13: RBP forecasting errors with meter ID: CC030050742.

Model	Month	Input Features	RMSE
RBP	Jan	HC	0.1332
	Feb	HC	0.0822
	Mar	HC	0.1500
	April	HC	0.0888
	May	HC	0.1407
	June	HC	0.1167
	July	HC	0.1437
	Aug	HC	0.1308
	Sep	HC	0.0964
	Total (Jan to Sep)	HC	0.1231

Table 7.14: RBP forecasting errors with meter ID: CC029790952.

Model	Month	Input Features	RMSE
RBP	Jan	HC	0.0286
	Feb	HC	0.0341
	Mar	HC	0.0481
	April	HC	0.0251
	May	HC	0.0260
	June	HC	0.0297
	July	HC	0.0282
	Aug	HC	0.0288
	Sep	HC	0.0279
	Total (Jan to Sep)	HC	0.0315

7.6.5.2 LR Forecasting Results

LR forecasting model is applied by using the same experimental design and feature patterns previously discussed. The LR model produced better forecasting results than the RBP as shown in Table 7.15 and Table 7.16. In the experimental evaluation, it is observed that the performance of the LR model is affected with the presence

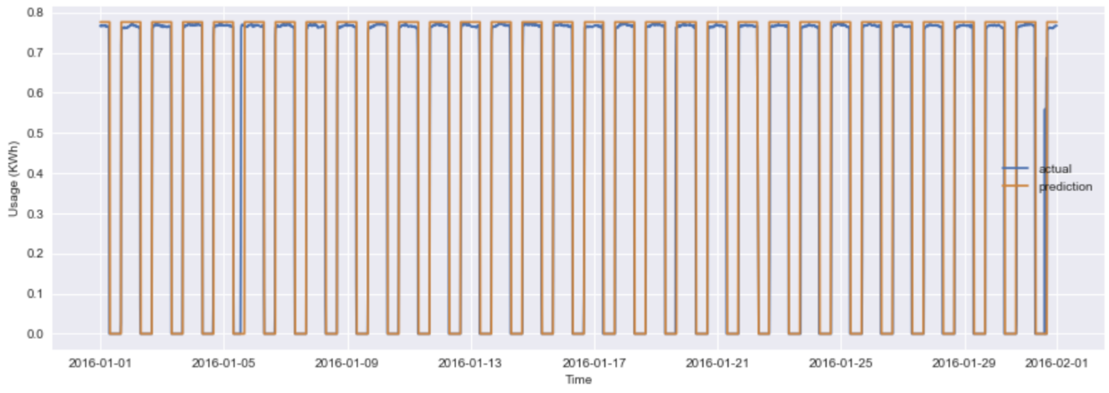


Figure 7.24: Actual and predicted energy consumption forecasting of RBP with meter ID: CC030050742.

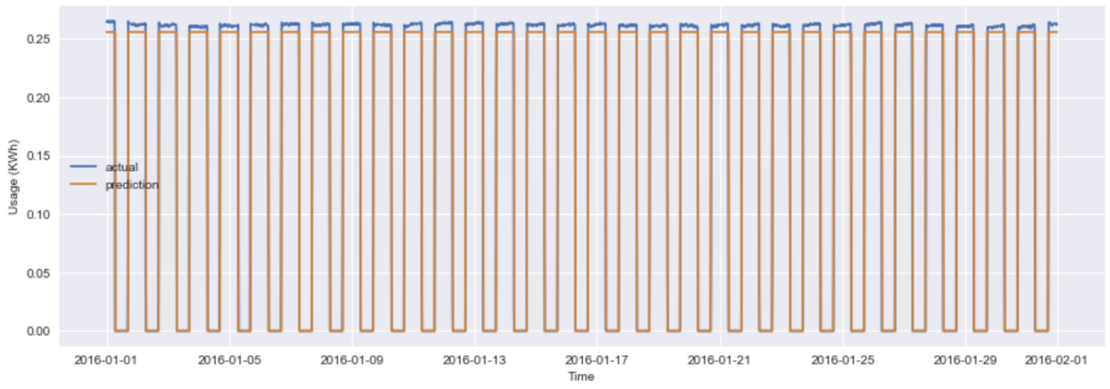


Figure 7.25: Actual and predicted energy consumption forecasting of RBP with meter ID: CC029790952.

of outliers as shown in Table 7.15. It is also observed that the addition of time information (hour of the day) with historic energy consumption value does not have a significant impact on the quality of forecasting accuracy as shown in Table 7.15 and Table 7.16. Figure 7.26 and Figure 7.27 shows the comparison of actual and predicted energy consumption of LR models for both the smart meters.

Table 7.15: LR forecasting errors with meter ID: CC030050742.

Model	Month	Input Features	RMSE
LR	Jan	HC	0.0918
		HC & time(hour)	0.0912
	Feb	HC	0.0870
		HC & time(hour)	0.0861

Continued on next page

Table 7.15 – continued from previous page

Model	Month	Input Features	RMSE
LR	Mar	HC	0.0905
		HC & time(hour)	0.0897
	Apr	HC	0.0924
		HC & time(hour)	0.0914
	May	HC	0.0911
		HC & time(hour)	0.0900
	June	HC	0.0954
		HC & time(hour)	0.0943
	July	HC	0.0892
		HC & time(hour)	0.0880
	Aug	HC	0.0908
		HC & time(hour)	0.0898
	Sep	HC	0.0908
		HC & time(hour)	0.0876
	Total (Jan to Sep)	HC	0.0908
		HC & time(hour)	0.0898

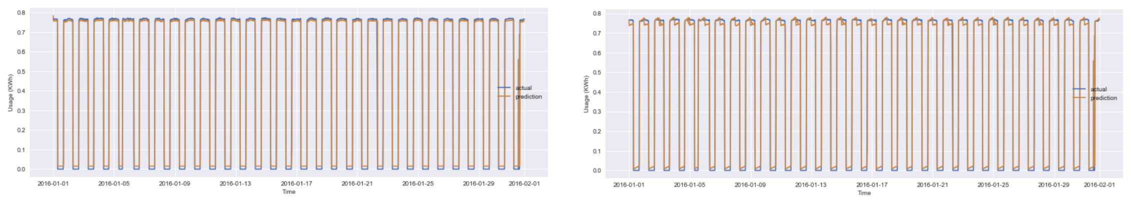
Table 7.16: LR forecasting errors with meter ID: CC029790952.

Model	Month	Input Features	RMSE
LR	Jan	HC	0.0308
		HC & time(hour)	0.0306
	Feb	HC	0.0307
		HC & time(hour)	0.0304
	Mar	HC	0.0303
		HC & time(hour)	0.0300
	Apr	HC	0.0304
		HC & time(hour)	0.0301
	May	HC	0.0302
		HC & time(hour)	0.0298
	June	HC	0.0297
		HC & time(hour)	0.0292

Continued on next page

Table 7.16 – continued from previous page

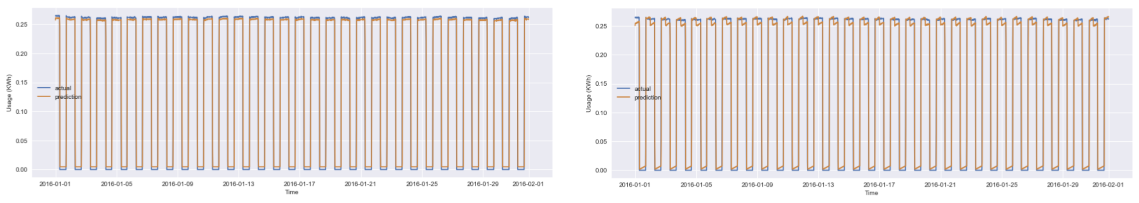
Model	Month	Input Features	RMSE
	July	HC	0.0312
		HC & time(hour)	0.0308
	Aug	HC	0.0278
		HC & time(hour)	0.0284
	Sep	HC	0.0305
		HC & time(hour)	0.0301
	Total (Jan to Sep)	HC	0.0303
		HC & time(hour)	0.0299



(a) Actual vs predicted energy consumption with HC feature.

(b) Actual vs predicted energy consumption with HC & time feature.

Figure 7.26: Actual vs predicted energy consumption forecasting of LR with meter ID: CC030050742.



(a) Actual vs predicted consumption with HC feature.

(b) Actual vs predicted consumption with HC & time feature.

Figure 7.27: Actual vs predicted energy consumption forecasting of LR with meter ID: CC029790952.

7.6.5.3 SVR Forecasting Results

We applied the SVR model by using the same experimental design and feature patterns previously discussed. We also kept the same parameter settings to configure the model that used for PELL data as given in 7.3. The results generated by SVR forecasting model are presented in Table 7.17 and Table 7.18. We can notice that the forecasting results of SVR model are similar to LR model when we trained the model on a single HC feature to made forecasting. But on the other hand, a little bit improvement in the forecasting results is observed when the model performance is evaluated by using HC and time features. Figure 7.28 and Figure 7.29 shows the comparison of actual and predicted energy consumption of SVR models for both the smart meters.

Table 7.17: SVR forecasting errors with meter ID: CC030050742.

Model	Month	Input Features	RMSE
SVR	Jan	HC	0.0931
		HC & time(hour)	0.0688
	Feb	HC	0.0874
		HC & time(hour)	0.0651
	Mar	HC	0.0916
		HC & time(hour)	0.0749
	Apr	HC	0.0936
		HC & time(hour)	0.0628
	May	HC	0.0926
		HC & time(hour)	0.0767
	June	HC	0.0961
		HC & time(hour)	0.0949
	July	HC	0.0903
		HC & time(hour)	0.0723
	Aug	HC	0.0914
		HC & time(hour)	0.0566
	Sep	HC	0.0898
		HC & time(hour)	0.0720
	Total (Jan to Sep)	HC	0.0909
		HC & time(hour)	0.0729

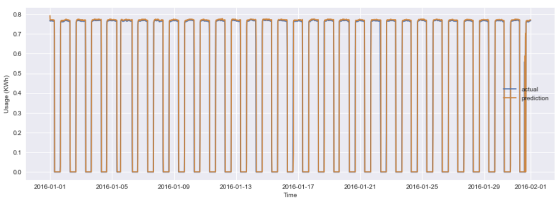
Table 7.18: SVR forecasting errors with meter ID: CC029790952.

Model	Month	Input Features	RMSE
SVR	Jan	HC	0.0310
		HC & time(hour)	0.0248
	Feb	HC	0.0312
		HC & time(hour)	0.0274
	Mar	HC	0.0304

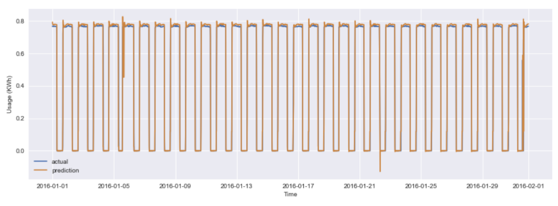
Continued on next page

Table 7.18 – continued from previous page

Model	Month	Input Features	RMSE
SVR		HC & time(hour)	0.0269
	Apr	HC	0.0306
		HC & time(hour)	0.0273
	May	HC	0.0304
		HC & time(hour)	0.0246
	June	HC	0.0298
		HC & time(hour)	0.0203
	July	HC	0.0314
		HC & time(hour)	0.0249
	Aug	HC	0.0289
		HC & time(hour)	0.0232
	Sep	HC	0.0306
		HC & time(hour)	0.0240
	Total (Jan to Sep)	HC	0.0305
		HC & time(hour)	0.0254



(a) Actual vs predicted energy consumption with HC feature.

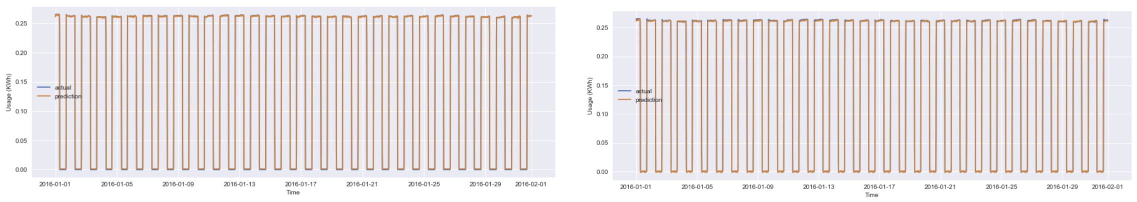


(b) Actual vs predicted energy consumption with HC & time feature.

Figure 7.28: Actual vs predicted energy consumption forecasting of SVR with meter ID: CC030050742.

7.6.5.4 DT Forecasting Results

we applied the DT forecasting model by using the same experimental design and feature patterns previously discussed. To configure the model, we employed the same parameter settings used for PELL data as given in Table 7.6. The forecasting results obtained by using DT are presented in Table 7.19 and Table 7.20.



(a) Actual vs predicted energy consumption with HC feature.

(b) Actual vs predicted energy consumption with HC & time feature.

Figure 7.29: Actual vs predicted energy consumption forecasting of SVR with meter ID: CC029790952.

Table 7.19: DT forecasting errors with meter ID: CC030050742.

Model	Month	Input Features	RMSE
DT	Jan	HC	0.0964
		HC & time(hour)	0.0998
	Feb	HC	0.1019
		HC & time(hour)	0.0984
	Mar	HC	0.1020
		HC & time(hour)	0.1020
	Apr	HC	0.1082
		HC & time(hour)	0.1033
	May	HC	0.1036
		HC & time(hour)	0.0980
	June	HC	0.1017
		HC & time(hour)	0.1017
	July	HC	0.0959
		HC & time(hour)	0.1028
	Aug	HC	0.0991
		HC & time(hour)	0.0957
	Sep	HC	0.0954
		HC & time(hour)	0.0911
	Total (Jan to Sep)	HC	0.0988
		HC & time(hour)	0.1015

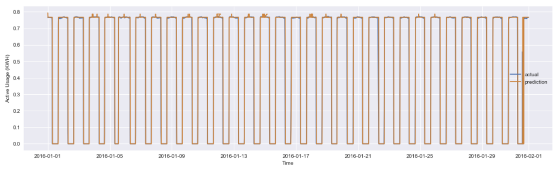
Table 7.20: DT forecasting errors with meter ID: CC029790952.

Model	Month	Input Features	RMSE
DT	Jan	HC	0.0338
		HC & time(hour)	0.0316
	Feb	HC	0.0322
		HC & time(hour)	0.0230
	Mar	HC	0.0344
		HC & time(hour)	0.0206
	Apr	HC	0.0350
		HC & time(hour)	0.0350
	May	HC	0.0338
		HC & time(hour)	0.0338
	June	HC	0.0310
		HC & time(hour)	0.0241
	July	HC	0.0333
		HC & time(hour)	0.0317
	Aug	HC	0.0348
		HC & time(hour)	0.0319
	Sep	HC	0.0354
		HC & time(hour)	0.0252
	Total (Jan to Sep)	HC	0.0324
		HC & time(hour)	0.0270

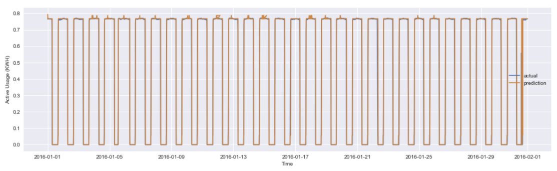
We can observe that DT models produced higher forecasting errors than the LR and SVR models especially when there are outliers in the data (e.g. meter ID: CC030050742). It can also be noted that for the same meter, DT model generated more errors when the time information is added to train the model. Whereas, for the second meter DT model still showed comparable accuracy, even though a little bit less than LR and SVR. Figure 7.30 and Figure 7.31 hows the comparison of actual and predicted energy consumption of DT models for both the smart meters.

7.6.5.5 RF Forecasting Results

RF forecasting model is also applied by using the same experimental design and feature patterns previously discussed. We also kept the same parameter settings to

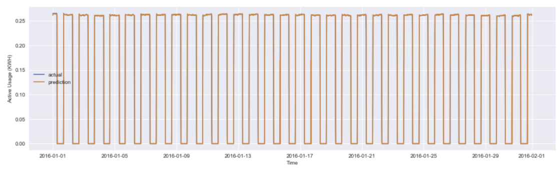


(a) Actual vs predicted energy consumption with HC feature.

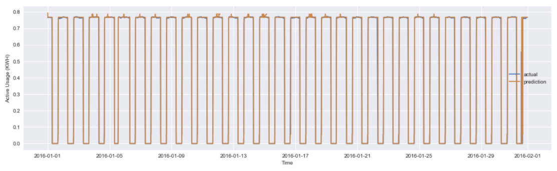


(b) Actual vs predicted energy consumption with HC & time feature.

Figure 7.30: Actual vs predicted energy consumption forecasting of DT with meter ID: CC030050742.



(a) Actual vs predicted energy consumption with HC feature.



(b) Actual vs predicted energy consumption with HC & time feature.

Figure 7.31: Actual vs predicted energy consumption forecasting of DT with meter ID: CC029790952.

configure the RF model that used for PELL data as given in 7.8. The forecasting results produced by RF are given in Table 7.21 and Table 7.22.

Table 7.21: RF forecasting errors with meter ID: CC030050742.

Model	Month	Input Features	RMSE
RF	Jan	HC	0.1145
		HC & time(hour)	0.0948
	Feb	HC	0.1506
		HC & time(hour)	0.0978
	Mar	HC	0.0976
		HC & time(hour)	0.0609
	Apr	HC	0.1221
		HC & time(hour)	0.0799
	May	HC	0.0976
		HC & time(hour)	0.0599
	June	HC	0.1304
		HC & time(hour)	0.1887
	July	HC	0.0856

Continued on next page

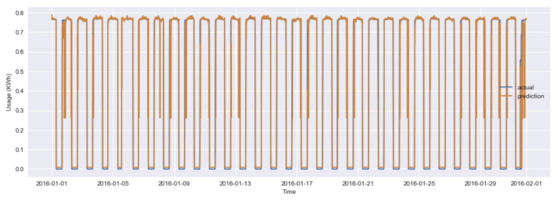
Table 7.21 – continued from previous page

Model	Month	Input Features	RMSE
		HC & time(hour)	0.0801
	Aug	HC	0.0985
		HC & time(hour)	0.0530
	Sep	HC	0.1030
		HC & time(hour)	0.0726
	Total (Jan to Sep)	HC	0.0952
		HC & time(hour)	0.0844

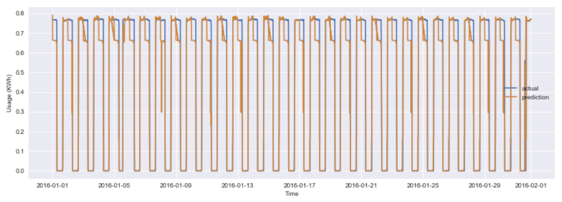
Table 7.22: RF forecasting errors with meter ID: CC029790952.

Model	Month	Input Features	RMSE
RF	Jan	HC	0.0323
		HC & time(hour)	0.0185
	Feb	HC	0.0305
		HC & time(hour)	0.0196
	Mar	HC	0.0316
		HC & time(hour)	0.0188
	Apr	HC	0.0304
		HC & time(hour)	0.0186
	May	HC	0.0299
		HC & time(hour)	0.0166
	June	HC	0.0249
		HC & time(hour)	0.0158
	July	HC	0.0305
		HC & time(hour)	0.0158
	Aug	HC	0.0297
		HC & time(hour)	0.0178
	Sep	HC	0.0340
		HC & time(hour)	0.0225
	Total (Jan to Sep)	HC	0.0301
		HC & time(hour)	0.0187

We can notice that RF forecasting model still showed comparable forecasting accuracy, even though little bit less than LR and SVR, and much better than DT for the meter ID: CC030050742. We can also observe that for the same meter an improvement in forecasting accuracy is observed while using the HC and time (hour of the day) as features set. Regarding the second meter ID: CC029790952, we can notice that RF forecasting model outperformed all the selected forecasting models as shown in Table 7.22. Especially a significant improvement can be seen from results in Table 7.22 when we evaluated the model by using both time and HC as feature set. The results show that RF produced superior forecasting results than all the selected models when there are fewer outliers in the data. Figure 7.32 and Figure 7.32 shows the comparison of actual and predicted energy consumption of RF models for both the smart meters.

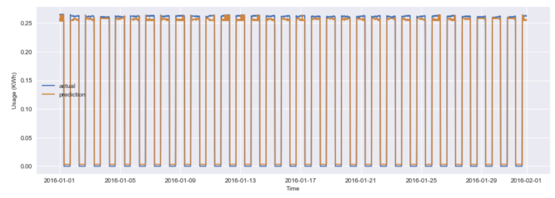


(a) Actual vs predicted energy consumption with HC feature.

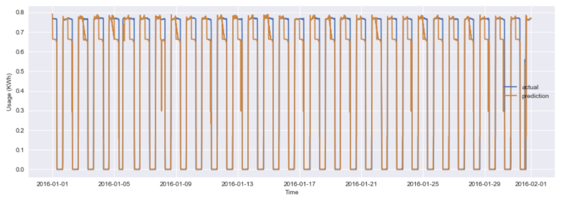


(b) Actual vs predicted energy consumption with HC & time feature.

Figure 7.32: Actual vs predicted energy consumption forecasting of RF with meter ID: CC030050742.



(a) Actual vs predicted energy consumption with HC feature.



(b) Actual vs predicted energy consumption with HC & time feature.

Figure 7.33: Actual vs predicted energy consumption forecasting of RF with meter ID: CC029790952.

7.6.5.6 KNN Forecasting Results

We applied the KNN forecasting model on energy usage data by using the same experimental design and feature patterns previously discussed. We also kept the same parameter settings to configure the model that used for PELL data i.e. K value to one. The forecasting results produced by KNN model are given in Table 7.23 and Table 7.24. We can notice that KNN generated higher forecasting errors in the case when there are outliers in the data as shown in Table 7.23. We can also observe that the KNN model produced worst results when we evaluated the model by using both time and HC as feature set for the meter (ID: CC030050742). Regarding the seconed meter (ID: CC029790952), we can see KNN forecasting model showed comparable forecasting errors, even though a little bit less than RBP, LR, SVR, RF, and DT. Whereas , for the same meter, a significant improvement in the forecasting quality is observed when we evaluated the model by adding time information with the historic consumption as shown in Table 7.24. Figure 7.34 and Figure 7.35 hows the comparison of actual and predicted energy consumption of KNN models for both the meters.

Table 7.23: KNN forecasting errors with meter ID: CC030050742.

Model	Month	Input Features	RMSE
KNN	Jan	HC	0.1538
		HC & time(hour)	0.0669
	Feb	HC	0.2071
		HC & time(hour)	0.1090
	Mar	HC	0.1069
		HC & time(hour)	0.0668
	Apr	HC	0.1767
		HC & time(hour)	0.0910
	May	HC	0.1169
		HC & time(hour)	0.0715
	June	HC	0.1747
		HC & time(hour)	0.1311
	July	HC	0.0936
		HC & time(hour)	0.1173
	Aug	HC	0.1141
		HC & time(hour)	0.0840
	Sep	HC	0.1345
		HC & time(hour)	0.0987
	Total (Jan to Sep)	HC	0.1023
		HC & time(hour)	0.1226

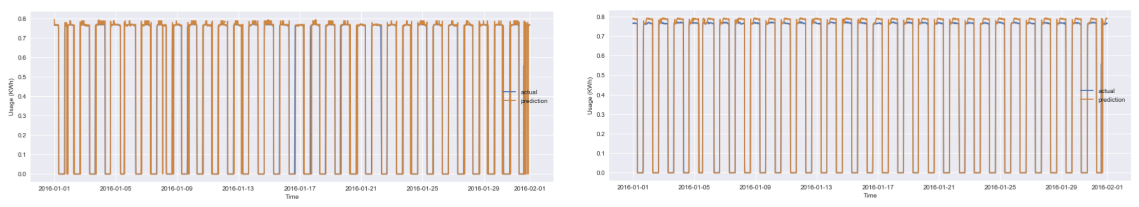
Table 7.24: KNN forecasting errors with meter ID: CC029790952.

Model	Month	Input Features	RMSE
KNN	Jan	HC	0.0323
		HC & time(hour)	0.0295
	Feb	HC	0.0294

Continued on next page

Table 7.24 – continued from previous page

Model	Month	Input Features	RMSE
KNN		HC & time(hour)	0.0230
	Mar	HC	0.0406
		HC & time(hour)	0.0218
	Apr	HC	0.0303
		HC & time(hour)	0.0270
	May	HC	0.0299
		HC & time(hour)	0.0308
	June	HC	0.0245
		HC & time(hour)	0.0207
	July	HC	0.0305
		HC & time(hour)	0.0312
	Aug	HC	0.0295
		HC & time(hour)	0.0320
	Sep	HC	0.0326
		HC & time(hour)	0.0245
	Total (Jan to Sep)	HC	0.0333
		HC & time(hour)	0.0205



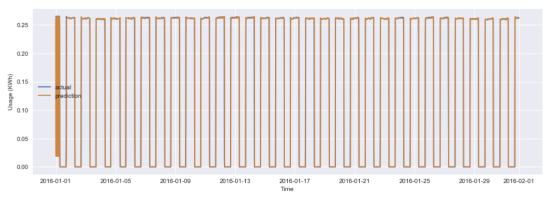
(a) Actual vs predicted energy consumption with HC feature.

(b) Actual vs predicted energy consumption with HC & time feature.

Figure 7.34: Actual vs predicted energy consumption forecasting of KNN with meter ID: CC030050742.

7.6.5.7 MLP Forecasting Results

MLP forecasting model is also applied by using the same experimental design and feature patterns previously discussed. We also kept the same parameter settings to configure the MLP model that used for the PELL data as given in 7.10. The forecasting results produced by MLP are given in Table 7.25 and Table 7.26.


(a) Actual vs predicted energy consumption with HC feature.

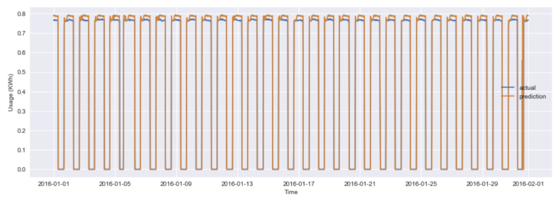

(b) Actual vs predicted energy consumption with HC & time feature.

Figure 7.35: Actual vs predicted energy consumption forecasting of KNN with meter ID: CC029790952.

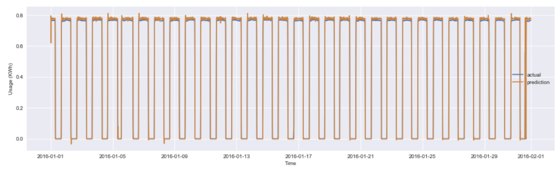
Table 7.25: MLP forecasting errors with meter ID: CC030050742.

Model	Month	Input Features	RMSE
MLP	Jan	HC	0.0639
		HC & time(hour)	0.0585
	Feb	HC	0.0655
		HC & time(hour)	0.0586
	Mar	HC	0.0718
		HC & time(hour)	0.0643
	Apr	HC	0.0726
		HC & time(hour)	0.0648
	May	HC	0.0717
		HC & time(hour)	0.0686
	June	HC	0.0747
		HC & time(hour)	0.0689
	July	HC	0.0602
		HC & time(hour)	0.0606
	Aug	HC	0.0705
		HC & time(hour)	0.0551
	Sep	HC	0.0615
		HC & time(hour)	0.0582
	Total (Jan to Sep)	HC	0.0657
		HC & time(hour)	0.0645

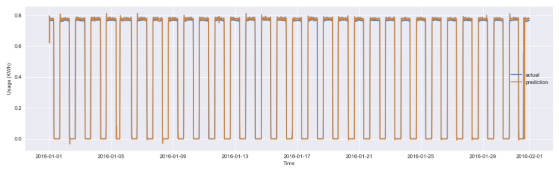
Table 7.26: MLP forecasting errors with meter ID: CC029790952.

Model	Month	Input Features	RMSE
MLP	Jan	HC	0.0240
		HC & time(hour)	0.0233
	Feb	HC	0.0237
		HC & time(hour)	0.0240
	Mar	HC	0.0210
		HC & time(hour)	0.0207
	Apr	HC	0.0238
		HC & time(hour)	0.0243
	May	HC	0.0223
		HC & time(hour)	0.0227
	June	HC	0.0221
		HC & time(hour)	0.0212
	July	HC	0.0233
		HC & time(hour)	0.0219
	Aug	HC	0.0200
		HC & time(hour)	0.0213
	Sep	HC	0.0247
		HC & time(hour)	0.0234
	Total (Jan to Sep)	HC	0.0215
		HC & time(hour)	0.0210

We can notice that MLP produced superior forecasting results even in the presence of outliers in the data as shown in Table 7.25. Regarding the second meter, we can observe that the forecasting quality of MLP is better than LR, SVR, and DT. It is important to note that, we can not see a significant improvement in the forecasting quality of the MLP when the model is trained on historic consumption and time features. It means that the addition of the time feature does not have an impact on the forecasting accuracy of the MLP model for street energy consumption forecasting. Figure 7.36 and Figure 7.37 hows the comparison of actual and predicted energy consumption of KNN models for both the meters.

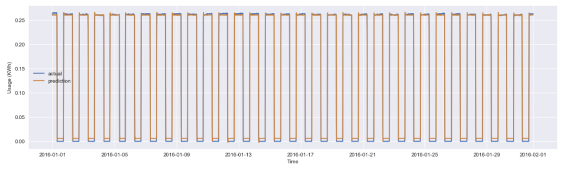


(a) Actual vs predicted energy consumption with HC feature.

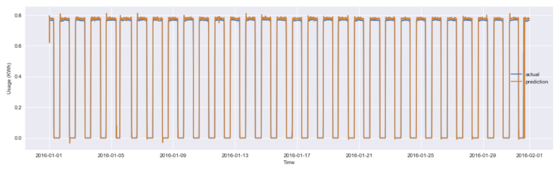


(b) Actual vs predicted energy consumption with HC & time feature.

Figure 7.36: Actual vs predicted energy consumption forecasting of MLP with meter ID: CC030050742.



(a) Actual vs predicted energy consumption with HC feature.



(b) Actual vs predicted energy consumption with HC & time feature.

Figure 7.37: Actual vs predicted energy consumption forecasting of MLP with meter ID: CC029790952.

7.6.6 Analysis and Discussion

In this empirical investigation, we evaluated the performance of selected forecasting models on two different smart meters data. We observed that the performance of the selected forecasting models except MLP suffered in the presence of outliers in the data. For the first meter, we noted that the proposed RBP and KNN forecasting models generated worst forecasting results with a single feature i.e. HC as shown in Table 7.27.

For the same meter, when we trained the models by using multi-features, i.e. HC with time information (hour of the day), the KNN and DT models generated the worst forecasting results whereas an improvement is noted in the forecasting quality of the LR, SVR, and RF as shown in Table 7.30. We observed that MLP forecasting model produced superior forecasting results in the presence of the outlier as shown in Table 7.27 and Table 7.28. Regrading the second meter, we observed that by using the single feature i.e. HC, MLP outperformed the other forecasting models. On the other hand, when time information (hour of the day) is added with HC, we noted that RF produced superior forecasting results. In this analysis, we concluded that MLP would be a better option to forecast the street lighting energy in the presence of outliers in the data. But, in the case of fine tune data, RF with

Table 7.27: Performance comparison of the used forecasting models for meter ID:CC030050742 with HC as feature set.

Month	**RBP**	**LR**	**SVR**	**KNN**	**RF**	**DT**	**MLP**
	RMSE	RMSE	RMSE	RMSE	RMSE	RMSE	RMSE
Jan	0.1332	0.0918	0.0931	0.1538	0.1145	0.0964	0.0639
Feb	0.0822	0.087	0.0874	0.2071	0.1506	0.1019	0.0655
March	0.15	0.0905	0.0916	0.1069	0.0976	0.102	0.0718
April	0.0888	0.0924	0.0936	0.1767	0.1221	0.1082	0.0726
May	0.1407	0.0911	0.0926	0.1169	0.0976	0.1036	0.0717
June	0.1167	0.0954	0.0961	0.1747	0.1304	0.1017	0.0747
July	0.1437	0.0892	0.0903	0.0936	0.0856	0.0959	0.0602
Aug	0.1308	0.0908	0.0914	0.1141	0.0985	0.0991	0.0705
Sep	0.0964	0.0908	0.0898	0.1345	0.103	0.0954	0.0615
Total (Jan to Sep)	0.1231	0.0908	0.0909	0.1023	0.0952	0.0988	0.0657

multi-features (HC and time information) would be an appropriate choice for street lighting energy consumption forecasting.

Table 7.28: Performance comparison of the used forecasting models for meter ID:CC030050742 with HC & time information (day of the hour) as feature set.

Month	**LR**	**SVR**	**RF**	**KNN**	**DT**	**MLP**
	RMSE	RMSE	RMSE	RMSE	RMSE	RMSE
Jan	0.0912	0.0688	0.0948	0.0669	0.0998	0.0585
Feb	0.0861	0.0651	0.0978	0.109	0.0984	0.0586
March	0.0897	0.0749	0.0609	0.0668	0.102	0.0643
April	0.0914	0.0628	0.0799	0.091	0.1033	0.0648
May	0.09	0.0767	0.0599	0.0715	0.098	0.0686
June	0.0943	0.0949	0.1887	0.1311	0.1017	0.0689
July	0.088	0.0723	0.0801	0.1173	0.1028	0.0606
Aug	0.0898	0.0566	0.053	0.084	0.0957	0.0551
Sep	0.0876	0.072	0.0726	0.0987	0.0911	0.0582
Total (Jan to Sep)	0.0898	0.0729	0.0844	0.1226	0.1015	0.0645

7.7 Threats to Validity

Construct validity threats may arise due to the lack of benchmark data of public street lighting. To mitigate these threats, we used a PELL data set based on an established benchmark from ENEA, and a publicly available street lighting energy usage data set.

Table 7.29: Performance comparison of the used forecasting models for meter ID:CC029790952 with HC as feature set.

Month	**RBP**	**LR**	**SVR**	**KNN**	**RF**	**DT**	**MLP**
	RMSE	RMSE	RMSE	RMSE	RMSE	RMSE	RMSE
Jan	0.0286	0.0308	0.031	0.0323	0.0323	0.0338	0.0240
Feb	0.0341	0.0307	0.0312	0.0294	0.0305	0.0322	0.0237
March	0.0481	0.0303	0.0304	0.0406	0.0316	0.0344	0.0210
April	0.0251	0.0304	0.0306	0.0303	0.0304	0.0350	0.0238
May	0.026	0.0302	0.0304	0.0299	0.0299	0.0338	0.0223
June	0.0297	0.0297	0.0298	0.0245	0.0249	0.0310	0.0221
July	0.0282	0.0312	0.0314	0.0305	0.0305	0.0333	0.0233
Aug	0.0288	0.0278	0.0289	0.0295	0.0297	0.0348	0.0200
Sep	0.0279	0.0305	0.0306	0.0326	0.034	0.0354	0.0247
Total (Jan to Sep)	0.0315	0.0303	0.0305	0.0333	0.0301	0.0324	0.0215

Internal validity threats may be caused by bias in establishing cause-effect relationships in our experiments. To limit these threats, we performed extensive experiments for each forecasting model by setting different parameter settings. We also enable replication by making the experimental results publicly available.

External validity threats may exist if the characteristics of the data set in our benchmark are not indicative of the characteristics of data produced by other street lighting systems. This type of threat applied to our context at some extent since we rely on the PELL data model that is aimed at collecting and treat in an uniform way all different characteristics related to heterogeneous street lighting plants. Moreover, we evaluated the performance of forecasting models on PELL data as well as on a publicly available street lighting energy data.

To avoid threats to *conclusion validity*, we repeated our experiments multiple times. In particular, forecasting evaluation metrics (R, RMSE, MAE, MAPE) have been measured calculating them on two different street lighting data sets. Furthermore, each experiment has been repeated five times and average results is considered for each forecasting model.

Regarding the *scalability analysis*, we do not perform any scalability study in terms of high volume of data. We postpone scalability analysis as future work such a study with the availability of a very larger data set from different municipalities.

Table 7.30: Performance comparison of the used forecasting models for meter ID:CC029790952 with HC and time information (hour of the day) as feature set.

Month	**LR**	**SVR**	**KNN**	**RF**	**DT**	**MLP**
	RMSE	RMSE	RMSE	RMSE	RMSE	RMSE
Jan	0.0306	0.0248	0.0295	0.0185	0.0316	0.0233
Feb	0.0304	0.0274	0.023	0.0196	0.023	0.0240
March	0.03	0.0269	0.0218	0.0188	0.0206	0.0237
April	0.0301	0.0273	0.027	0.0186	0.035	0.0243
May	0.0298	0.0246	0.0308	0.0166	0.0338	0.0227
June	0.0292	0.0203	0.0207	0.0158	0.0241	0.0212
July	0.0308	0.0249	0.0312	0.0169	0.0317	0.0219
Aug	0.0284	0.0232	0.032	0.0178	0.0319	0.0213
Sep	0.0301	0.024	0.0245	0.0225	0.0252	0.0234
Total (Jan to Sep)	0.0299	0.0254	0.0205	0.0187	0.0270	0.0210

7.8 Summary

In this chapter, we presented an energy consumption forecasting approach for public street lighting. This approach is developed as the the part of PELL SCP. We developed a novel domain specific rule-based predictor to forecast street lighting energy consumption. Furthermore, we also used different ML forecasting models such LR, SVR, KNN, RF, DT, and MLP to forecast street lighting energy consumption. The objective of the proposed forecasting approach is to make a comparison of ML models with the developed rule-based predictor and to identify a most suitable forecasting model for street lighting energy consumption forecasting. In order to investigate the performance of the selected models, we employed two street light data (discussed in Section 7.2.1 and Section 7.2.2). In the first experiment, we make a comparative analysis of the ML models by using PELL street lighting data. We used two different versions of each ML models. The only difference between the used versions is which features are used as an input to train the model. In the first version we used HC to train the models and in the second version we added time information (hour of the day) with the HC. In order to optimally configure the ML forecasting models, we used grid search approach to determine the best parameter settings for the selected models. In this analysis, we infer that the addition of time information (hour of the day) with the HC value improved the forecasting quality of the ML models, and RF produced superior forecasting results than the other used models.

In the second evaluation, we make a comparative analysis of the selected ML forecasting models with proposed RBP by using public energy street lighting data (discussed in Section 7.2.2). We applied all the selected forecasting model by using the same experimental design and feature patterns we used for PELL data. We also kept the same parameter settings to configure the models that we employed for PELL data forecasting. In this empirical evaluation, we observed that all the ML models produced better forecasting results than the RBP. We observed that forecasting accuracy of the selected models is significantly improved when we trained the models by using multi-features such as HC value and time information (hour of the day).

Chapter 8

Conclusions and Future Research Directions

In this final chapter, we summarize the overall contribution of this dissertation by discussing the objectives in relation to the findings achieved in the earlier chapters. Afterwards, we discuss future work.

8.1 Research Findings

The primary objective of the Smart City is to improve the city infrastructures and services for citizens by making use of BD and IoT solutions. Development of intelligent smart services is equally important for citizens as well as for the business organizations. Public lighting management is one of the most crucial problems for smart cities. No doubt, BD technologies are very essentials for the storage and aggregation of large scale heterogeneous multimodal energy data, but computationally intensive machine intelligence and data analytics solutions are required to extract the hidden useful information that facilitate the city planners. In this context, the main research question that we have tackled in this dissertation is:

RQ: How to enrich a smart city platform for big street lighting data with data analytics features?

Although BD technological solutions make the collection and availability of very large amounts of data possible, they also demand for new paradigms for massive data and models that go beyond processing, storing and accessing records rapidly.

New challenges of scalability, quality, processing and visualization have emerged with handling and structuring data with new magnitudes of dimensionality, complexity, heterogeneity, and timeliness. These new facing challenges, along with the lack of best practices and of reported development experiences may increase sense of confusion about the use of the BD technology and the effectiveness of the process for gaining valuable insights from data, thus hindering their application for driving better business and shaping smarter city services. This dissertation addressed these critical challenges by dealing with four objectives.

O1: Exploration from a software-architecture perspective of a system engineering approach to architect a smart-city platform with BDA. To achieve the first research objective, in Chapter 4 we performed a systematic literature review on software platforms for smart cities at the architectural level, focusing on self-adaptation and BDA, and the relation between the two. The findings of this analysis give insights on existing software platforms providing smart city services by revealing features, applicability, and limitations of current software architectural solutions. In addition, a comprehensive review of open source technology of BD management, analytics and visualization. Based on this analysis, we reengineered the PELL SCP by a typical data pipeline architecture style for BD systems that need to process data in near real-time and adopted Apache Spark as main open-source technology for large-scale data processing and analytics. This architecture pattern allowed us to reduce the design complexity and effort in developing such a software platform. It also allowed achieving the desired engineering characteristics of *scalability* as the amount of ingested data increases, easy data *accessibility* (preferably through an analytic query language), and *efficiency* in term of readiness of data and analytic results with low latency.

O2: Providing descriptive analytics through formulation and implementation of KPIs to analyze, measure and monitor electric energy consumption of public street lighting plants. To achieve the second objective of the dissertation , in Chapter 5 we examined the main building blocks realizing a scalable and efficient BD software pipeline for processing and managing urban data in the public street lighting domain, which is a substantial part of the smart city concept. We also provided some insights on how we defined and implemented some KPIs for managing energy consumption as part of the BDA framework of the ENEA PELL

SCP. It is important to underline that the main goal of these KPIs is to provide a feedback about the actual behavior of the lighting plant in respect of the expected behavior according to the declared information given in the static data. Thus, they are not designed to produce effective energy saving but only to improve the overall knowledge of the lighting plant behavior. This is precisely how smart data is produced in this specific context. The outcomes of the proposed KPIs are helpful for the managers to measure and monitor electric energy consumption of public street lighting plants.

O3: Devise a predictive analytics for the detection of anomalies from time series of street lighting energy consumption data. To achieve the third objective of the dissertation, in Chapter 6 we proposed an anomaly detection approach as part of the PELL SCP. The proposed approach is based on unsupervised clustering algorithms, namely K-means, DBSCAN, and OPTICS. In this work, we identified some anomalous scenarios, which are specific to the street lighting domain. Based on these anomalous scenarios, synthetic anomalies (point anomalies and collective anomalies) are introduced into the original electricity consumption data set. These synthetic anomalies are injected by using the application of novel algorithms. The performance of the clustering algorithms is measured on the basis of detection of synthetic anomalies by using standard evaluation metrics such as precision, recall, and F1-measure. This is a generic approach and can be applied on any street lighting data set to detect anomalies. The achieved results demonstrate that the proposed approach can help the managers in better decision-making to reduce the wasted energy and also promote sustainable and energy efficient behavior. In addition, it can also help in maintaining integrity and consistency in the data, which could be used by the ML models for predictive analysis and therefore could be helpful to attain better predictive performance.

O4: Empirical investigation of predictive models to select a suitable forecasting model for street lighting energy consumption. Finally to achieve the fourth objective of the dissertation, in Chapter 7 we proposed an approach to street lighting energy consumption forecasting as part of the PELL SCP. The proposed forecasting approach is based on ML forecasting models. In addition, we developed a domain specific rule-based predictor for street lighting energy consumption. This predictor is used as a baseline forecasting model to make a comparison with the

ML forecasting models. Furthermore, the proposed rule-based predictor is a generic predictor and can be applied on street lighting energy consumption data. In order to evaluate the effectiveness of the selected forecasting models, two different street lighting data sets (PELL data and a publicly available energy usage data[1]) were used. The predictive accuracy of the forecasting models is measured by using well known error metrics such as RMSE, MAE, MAPE, and R. The outcomes of this empirical investigation can help the managers to select the most suitable forecasting model to street lighting energy consumption, which can help to optimize energy allocation.

8.2 Future Work

Regarding exploration from a software-architecture perspective to architect a smart-city platform with BDA (see Chapter 4), we want to select a core subset from the examined architectures, considering those with a broad scope that may lay foundation to a reference architecture in the smart city context. In particular, we want to examine how the reported architectural patterns affect or benefit the accomplishment of requirements of quality attributes in smart city services. We want also to analyze the state of practice by looking at smart city initiatives by the major ICT companies (such as IBM, Microsoft, Oracle and Huawei) to discover if both the academy and industry are aligned together. We aim at developing more advanced lighting KPIs (see Chapter 5). In particular, we want to formulate and develop more advanced lighting KPIs as part of the PELL SCP which could helps to analyze, measure and monitor electric energy consumption of public street lighting plants.

As concerns the anomaly detection approach (see Chapter 6), we want to investigate the performance of the clustering algorithms on a large scale data set. It will help to strengthen the reliability and validity of the results produced by the anomaly detection algorithms. In addition, we also want to evaluate the real-time performance of our anomaly detection module as part of the PELL SCP. Regarding our street lighting energy consumption forecasting approach (see Chapter 6), we want to extend our experimentation on different benchmarks as it will help to make a robust comparison between the performance of rule-based predictor and ML forecasting models. In addition, it will also help to determine a most suitable model

[1]https://data.world/lasvegasnevada/street-light-energy-usage/

for street lighting energy consumption forecasting.

In this work, we designed and implemented a Big Data-centric software architecture for the analysis of energy consumption data in the context of public street lighting. We enriched the ENEA PELL SCP with data analytics features, which help the administrative authorities to analyze street lighting energy consumption and gain more insights for making on-time decisions about energy optimization. We implemented analytical services offline, but in the future we plan to work on an online version of the analytics framework by exposing its functionalities in terms of (micro-)services within the ENEA PELL SCP. As a future work, we also want to explore the proposed analytical framework within the context of the ENEA PELL project for the building energy domain.

Appendices

A Primary Studies

Tables A.1 and A.2 report all the primary studies identified for the first and the second search strings, respectively; while Table A.3 reports those selected from the initial manual search.

Table A.1: First search – List of primary studies

ID	Year	Authors	Title	Source
P1	2016	Michael Vogler et al.	A Scalable Framework for Provisioning Large-scale IoT Deployments	ACM Transactions on Internet Technology
P2	2017	Antonella Longo et al.	Crowd-Sourced Data Collection for Urban Monitoring via Mobile Sensors	ACM Transactions on Internet Technology Â· October 2017
P3	2014	Rodger Lea et al.	Smart Cities: an IoT-centric Approach	IWWISS '14: International Workshop on Web Intelligence and Smart Sensing
P4	2017	Diego O. Rodrigues et al.	SMAFramework: Urban Data Integration Framework for Mobility Analysis in Smart Cities	MSWiM '17: Proceedings of the 20th ACM International Conference on Modelling, Analysis and Simulation of Wireless and Mobile Systems
P5	2018	Sara Benyahia et al.	MARSC: A Macro-Architecture for Smart Cities	HPCCTâ€™18, June 22â€“24, 2018, Beijing, China

Continued on next page

Table A.1 – continued from previous page

No.	Year	Authors	Title	Source
P6	2018	Rahul Pandey et al.	Generic Architecture of a Social Media-driven Intervention Support System for Smart Cities	ICDCN â€™18: Workshops co-located with the International Conference on Distributed Computing and Networks 2018
P7	2017	Ralf-Martin Soe et al.	FINEST Twins: platform for cross-border smart city solutions	dg.o '17: Proceedings of the 18th Annual International Conference on Digital Government Research
P8	2017	Philipp LÃ¤mmel et al.	Enhancing Cloud based Data Platforms for Smart Cities with Authentication and Authorization Features	ucc'17 companion
P9	2016	Talal Shaikh et al.	Aura Minora: A user centric IOT architecture for Smart City	BDAW '16: Proceedings of the International Conference on Big Data and Advanced Wireless Technologies
P10	2018	Moussa Witti et l.	A Secure and Privacy-preserving Internet of Things Framework for Smart City	ICIT 2018: Proceedings of the 6th International Conference on Information Technology: IoT and Smart City
P11	2011	Y.-K. Juan et al.	A decision-support system for smarter city planning and management	IBM Journal of Research and Development
P12	2019	Fikret Sivrikaya et al.	Internet of Smart City Objects: A Distributed Framework for Service Discovery and Composition	IEEE access
P13	2016	Amir Sinaeepourfard et al.	A Data LifeCycle Model for Smart Cities	ICTC 2016
P14	2018	Gianfranco Gagliardi et al.	A Smart City Adaptive Lighting System	International Conference on Fog and Mobile Edge Computing (FMEC)
P15	2017	Riccardo Petrolo et al.	Adaptive Filtering as a Service for Smart City Applications	IEEE Int. conf. on Networking, Sensing and Control

Continued on next page

Table A.1 – continued from previous page

No.	Year	Authors	Title	Source
P16	2017	Mohammad Abu-Matar et al.	Data Driven Reference Architecture for Smart City Ecosystems	IEEE International Conference on Smart City Innovations (SCI) 2017
P17	2016	Antoine Auger et al.	iQAS: An Integration Platform for QoI Assessment as a Service for Smart Cities	Internet of Things (WF-IoT), IEEE World Forum
P18	2014	M. Dbouk et al.	CityPro; an integrated city protection collaborative platform	EUSP'2014
P19	2017	Sergio Trilles et al.	Deployment of an open sensorized platform in a smart city context	Elsevier Future Generation Computer Systems
P20	2014	C. Dobre et al.	Intelligent services for Big Data science	Elsevier Future Generation Computer Systems
P21	2017	Jose Aguilar et al.	An Extension of the MiSCi Middleware for Smart Cities Based on Fog Computing	IGI Global – Journal of Information Technology Research
P22	2019	JongGwan An et al.	Towards Global IoT-enabled Smart Cities Interworking using Adaptive Semantic Adapter	IEEE INTERNET OF THINGS JOURNAL
P23	2018	Ichiro Satoh	Adaptive Deployment of Software in Smart-City	Goodtechs '18: Proceedings of the 4th EAI International Conference on Smart Objects and Technologies for Social Good
P24	2018	Mohamed Dbouk et al.	CityPro: From Big-Data to Intelligent-Data; a Smart Approach.	1st International Conference on Big Data and Cyber-Security Intelligence, BDCSIntell 2018
P25	2019	Amr Azzam et al.	The CitySPIN Platform: A CPSS Environment for City-Wide Infrastructures	1st Workshop on Cyber-Physical Social Systems (CPSS2019)
P26	2015	Maria Giatsoglou et al.	CITYPULSE: A Platform Prototype for Smart City Social Data Mining	Springer Journal of the Knowledge Economy
P27	2020	Paraskevi Tsoutsa et al.	Nexus Services in Smart City Ecosystems	Springer Journal of the Knowledge Economy

Continued on next page

Table A.1 – continued from previous page

No.	Year	Authors	Title	Source
P28	2012	Sotiris Zygiaris	Smart City Reference Model: Assisting Planners to Conceptualize the Building of Smart City Innovation Ecosystems	Springer Journal of the Knowledge Economy
P29	2019	Stanimir Stoyanov et al.	A Reference Architecture Supporting Smart City Applications	International Conference on Business Information Systems
P30	2017	Fangping Li et al.	Aggregating Heterogeneous Services in the Smart City: The Practice in China	SmartCom 2016
P31	2018	Paolo Nesi et al.	An Integrated Smart City Platform	IKC 2017: Semantic Keyword-Based Search on Structured Data Sources
P32	2019	Pedro Martins et al.	CityAction a Smart-City Platform Architecture.	FICC 2019
P33	2017	Prathviraj Nagaraj et al.	Context-Aware Network for Smart City Services: A Layered Approach	SmartCom 2017
P34	2016	Ahmed Hefnawy et al.	Integration of Smart City and Lifecycle Concepts for Enhanced Large-Scale Event Management	IFIP International Federation for Information Processing 2016
P35	2019	Marcos Roriz Junior et al.	Mensageria: A Smart City Framework for Real-Time Analysis of Traffic Data Streams	Workshop on Big Social Data and Urban Computing
P36	2016	Jazon Coelho et al.	ROTA: A Smart City Platform to Improve Public Safety	New Advances in Information Systems and Technologies

Table A.2: Second search – List of primary studies

ID	Year	Authors	Title	Source
P1	2018	Paula Ta-Shma et al	An Ingestion and Analytics Architecture for IoT Applied to Smart City Use Cases	IEEE Internet of Things Journal
P2	2019	Sandro Fiore et al	An Integrated Big and Fast Data Analytics Platform for Smart Urban Transportation Management	IEEE Access

Continued on next page

Table A.2 – continued from previous page

No.	Year	Authors	Title	Source
P3	2019	Qimei Cui et al	Big Data Analytics and Network Calculus Enabling Intelligent Management of Autonomous Vehicles in a Smart City	IEEE Internet of Things Journal
P4	2019	Shih-Hau Fang et al	CityTracker: Citywide Individual and Crowd Trajectory Analysis Using Hidden Markov Model	IEEE Sensors Journal
P5	2019	Bilal Jan et al	Designing a Smart Transportation System: An Internet of Things and Big Data Approach	IEEE Wireless Communications
P6	2018	Jaganathan Venkatesh et al	Modular and Personalized Smart Health Application Design in a Smart City Environment	IEEE Internet of Things Journal
P7	2018	Claudio Badii et al	Predicting Available Parking Slots on Critical and Regular Services by Exploiting a Range of Open Data	IEEE Access
P8	2018	Theodora S. Brisimi et al	Predicting Chronic Disease Hospitalizations from Electronic Health Records: An Interpretable Classification Approach	Proceedings of the IEEE
P9	2018	Amr S. El-Wakeel et al	Towards a Practical Crowdsensing System for Road Surface Conditions Monitoring	IEEE Internet of Things Journal
P10	2019	Syed Attique Shah et al	Towards Disaster Resilient Smart Cities: Can Internet of Things and Big Data Analytics Be the Game Changers?	IEEE Access
P11	2018	Thaha Muhammed et al	UbeHealth: A Personalized Ubiquitous Cloud and Edge-Enabled Networked Healthcare System for Smart Cities	IEEE Access
P12	2018	Javier Martínez García et al	MIMO-FMCW Radar-Based Parking Monitoring Application With a Modified Convolutional Neural Network With Spatial Priors	IEEE Access
P13	2016	Rohith Polishetty et al	A Next-Generation Secure Cloud-Based Deep Learning License Plate Recognition for Smart Cities	2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA)

Continued on next page

Table A.2 – continued from previous page

No.	Year	Authors	Title	Source
P14	2016	Ikram Belhajem	A robust low cost approach for real time car positioning in a smart city using Extended Kalman Filter and evolutionary machine learning	2016 4th IEEE International Colloquium on Information Science and Technology (CiSt)
P15	2018	Tugce Bilen et al	A Smart City Application: Business Location Estimator Using Machine Learning Techniques	2018 IEEE 20th International Conference on High Performance Computing and Communications
P16	2017	Johnattan D. F. Viana et al	A visualization and analysis approach of cyclist data obtained through sensors	2017 IEEE First Summer School on Smart Cities (S3C)
P17	2014	Giuseppe Cardone et al	Activity recognition for Smart City scenarios: Google Play Services vs. MoST facilities	2014 IEEE Symposium on Computers and Communications (ISCC)
P18	2018	Liu Dan	Big Data Analytics Architecture for Internet-of-Vehicles Based on the Spark	2018 International Conference on Intelligent Transportation, Big Data & Smart City (ICITBS)
P19	2018	Johannes Zenkert et al	Big data analytics in smart mobility: Modeling and analysis of the Aarhus smart city dataset	2018 IEEE Industrial Cyber-Physical Systems (ICPS)
P20	2017	Satyanarayana V. Nandury et al	Big data for smart grid operation in smart cities	2017 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET)
P21	2017	Davide Mulfari et al	Building TensorFlow Applications in Smart City Scenarios	2017 IEEE International Conference on Smart Computing (SMARTCOMP)
P22	2016	Gusseppe Bravo Rocca et al	Citizen security using machine learning algorithms through open data	2016 8th IEEE Latin-American Conference on Communications (LATINCOM)
P23	2018	Jayraj Chopda et al	Cotton Crop Disease Detection using Decision Tree Classifier	2018 International Conference on Smart City and Emerging Technology (ICSCET)

Continued on next page

Table A.2 – continued from previous page

No.	Year	Authors	Title	Source
P24	2017	Osama A.Ghoneim et al	Forecasting of Ozone Concentration in Smart City using Deep Learning	2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)
P25	2017	Abida Sharif et al	Internet of things - smart traffic management system for smart cities using big data analytics	2017 14th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP)
P26	2017	Nitin Sadashiv Desai et al	IoT based air pollution monitoring and predictor system on Beagle bone black	2017 International Conference on Nextgen Electronic Technologies: Silicon to Software (ICNETS2)
P27	2018	Pavan Chhatpar,	Machine Learning Solutions to Vehicular Traffic Congestion	2018 International Conference on Smart City and Emerging Technology (ICSCET)
P28	2019	Danni Yuan et al	Privacy-Preserving Pedestrian Detection for Smart City with Edge Computing	2019 11th International Conference on Wireless Communications and Signal Processing (WCSP)
P29	2016	Patan Rizwan,	Real-time smart traffic management system for smart cities by using Internet of Things and big data	2017 14th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP)
P30	2017	Basheer Qolomany et al	Role of Deep LSTM Neural Networks and Wi-Fi Networks in Support of Occupancy Prediction in Smart Buildings	2017 IEEE 19th International Conference on High Performance Computing and Communications
P31	2019	Paul Seymer et al	Secure Outdoor Smart Parking Using Dual Mode Bluetooth Mesh Networks	2019 IEEE 89th Vehicular Technology Conference (VTC2019-Spring)

Continued on next page

Table A.2 – continued from previous page

No.	Year	Authors	Title	Source
P32	2018	Nader Mohamed et al	Service-Oriented Big Data Analytics for Improving Buildings Energy Management in Smart Cities	2018 14th International Wireless Communications & Mobile Computing Conference (IWCMC)
P33	2018	Zeenat Tariq et al	Smart 311 Request System with Automatic Noise Detection for Safe Neighborhood	2018 IEEE International Smart Cities Conference (ISC2)
P34	2018	Wael Alsafery et al	Smart Car Parking System Solution for the Internet of Things in Smart Cities	2018 1st International Conference on Computer Applications & Information Security (ICCAIS)
P35	2018	Adelson Ara´ujo Jr. et al	Towards a Crime Hotspot Detection Framework for Patrol Planning	2018 IEEE 20th International Conference on High Performance Computing and Communications
P36	2018	Nejdet Dogru	Traffic accident detection using random forest classifier	2018 15th Learning and Technology Conference (L&T)
P37	2018	Forough Goudarzi	Travel Time Prediction: Comparison of Machine Learning Algorithms in a Case Study	2018 IEEE 20th International Conference on High Performance Computing and Communications
P38	2019	Qilei Ren et al	Using Blockchain to Enhance and Optimize IoT-based Intelligent Traffic System	2019 International Conference on Platform Technology and Service (PlatCon)
P39	2016	Dinesh Singh et al	Visual Big Data Analytics for Traffic Monitoring in Smart City	2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA)
P40	2019	Adam Morrissett et al	A Physical Testbed for Intelligent Transportation Systems	IEEE Intelligent Systems
P41	2018	Gizem Abalı et al	Detecting Citizen Problems and Their Locations Using Twitter Data	2018 6th International Istanbul Smart Grids and Cities Congress and Fair (ICSG)
P42	2019	Sandeep Kumar E et al	A novel architecture to identify locations for Real Estate Investment	International Journal of Information Management

Continued on next page

Table A.2 – continued from previous page

No.	Year	Authors	Title	Source
P43	2019	Gurdit Singha et al	A smartphone based technique to monitor driving behavior using DTW and crowdsensing	Pervasive and Mobile Computing
P44	2020	Sandeep Saharan et al	An efficient smart parking pricing system for smart city environment: A machine-learning based approach	Future Generation Computer Systems
P45	2019	Hiram Ponce et al	An indoor predicting climate conditions approach using Internet-of-Things and artificial hydrocarbon networks	Measurement
P46	2020	Xiangtian Nie et al	Big Data analytics and IoT in Operation safety management in Under Water Management	Computer Communications
P47	2020	Rahat iqbal et al	Big data analytics: Computational intelligence techniques and application areas	Technological Forecasting and Social Change
P48	2019	Emilio Serrano et al	Deep neural network architectures for social services diagnosis in smart cities	Future Generation Computer Systems
P49	2018	M. Mazhar Rathore et al	Exploiting IoT and big data analytics: Defining Smart Digital City using real-time urban data	Sustainable Cities and Society
P50	2019	José Vázquez-Cantelia et al	Fusing TensorFlow with building energy simulation for intelligent energy management in smart cities	Sustainable Cities and Society
P51	2020	Marijana Zekić-Sušaca et al	Machine learning based system for managing energy efficiency of public sector as an approach towards smart cities	International Journal of Information Management
P52	2018	Marco Anisetti et al	Privacy-aware Big Data Analytics as a service for public health policies in smart cities	Sustainable Cities and Society
P53	2018	Moneeb Gohar et al	SMART TSS: Defining transportation system behavior using big data analytics in smart cities	Sustainable Cities and Society
P54	2017	Victoria Moreno	A data-driven methodology for heating optimization in smart buildings	2nd International Conference on Internet of Things, Big Data and Security (IoTBDS 2017)

Continued on next page

Table A.2 – continued from previous page

No.	Year	Authors	Title	Source
P55	2018	Wenrui Li et al	A mutilple-level assessment system for smart city street cleanliness	Thirtieth International Conference on Software Engineering and Knowledge Engineering (SEKE 2018)
P56	2019	Nur Tasnim Shamsuddin et al	Big Data Analytics Framework for Smart Universities Implementations	Proceedings of the 3rd International Symposium of Information and Internet Technology (SYMINTECH 2018)
P57	2018	Chia-Chun Chung et al	Information extraction methodology by web scraping for smart cities: Using machine learning to train air quality monitor for smart cities	CAADRIA 2018 - 23rd International Conference on Computer-Aided Architectural Design Research in Asia
P58	2020	Padma Prasada et al	Novel Approach in IoT-Based Smart Road with Traffic Decongestion Strategy for Smart Cities	Advances in Communication, Signal Processing, VLSI, and Embedded Systems
P59	2018	Abd-ElhamidM. Taha	An IoT Architecture for Assessing Road Safety in Smart Cities	Wireless Communications and Mobile Computing
P60	2018	Mamata Rath	Appraisal of soft computing methods in collaboration with smart city applications and wireless network	International Journal of e-Collaboration
P61	2019	B. Lalithadevi et al	Iot based WSN ground water monitoring system with cloud-based monitoring as a service (Maas) and prediction using machine learning	International Journal of Innovative Technology and Exploring Engineering
P62	2019	Mohammed G. H. AL Zamil et al	Multimedia-oriented action recognition in Smart City-based IoT using multilayer perceptron	Multimedia Tools and Applications
P63	2020	Bhupesh Kumar Mishra et al	A novel application of deep learning with image cropping: a smart city use case for flood monitoring	Journal of Reliable Intelligent Environments
P64	2020	B. Janakiramaiah et al	Automatic alert generation in a surveillance systems for smart city environment using deep learning algorithm	Evolutionary Intelligence

Continued on next page

Table A.2 – continued from previous page

No.	Year	Authors	Title	Source
P65	2018	Ikram Belhajem et al	Improving Vehicle Localization in a Smart City with Low Cost Sensor Networks and Support Vector Machines	Mobile Networks and Applications
P66	2019	Debadri Dutta et al	IoT based pollution monitoring and health correlation: a case study on smart city	International Journal of System Assurance Engineering and Management
P67	2018	M. Mazhar Rathore at al	Real-time video processing for traffic control in smart city using Hadoop ecosystem with GPUs	Soft Computing
P68	2019	Irshad Ullah	Smart Lightning Detection System for Smart-City Infrastructure Using Artificial Neural Network	Wireless Personal Communications
P69	2019	Alfredo Cuzzocrea et al	A Big-Data-Analytics System for Supporting Decision Making Processes in Complex Smart-City Applications	ICCSA: International Conference on Computational Science and Its Applications
P70	2017	Ikram Belhajem et al	A Hybrid Machine Learning Based Low Cost Approach for Real Time Vehicle Position Estimation in a Smart City	Advances in Ubiquitous Networking 2
P71	2013	Qingnan Zou	A Novel Taxi Dispatch System for Smart City	DAPI: International Conference on Distributed, Ambient, and Pervasive Interactions
P72	2017	Kenro Aihara et al	A Smart City Application for Sharing Up-to-date Road Surface Conditions Detected from Crowdsourced Data	DAPI: International Conference on Distributed, Ambient, and Pervasive Interactions
P73	2017	Ikram Belhajem et al	An Improved Robust Low Cost Approach for Real Time Vehicle Positioning in a Smart City	INISCOM: International Conference on Industrial Networks and Intelligent Systems
P74	2020	Bartosz Pawłowicz et al	Infrastructure of RFID-Based Smart City Traffic Control System	Automation 2019
P75	2019	Muthoni Masinde et al	Internet of Things-Based Framework for Public Transportation Fleet Management in Non-smart City	AFRICATEK: International Conference on Emerging Technologies for Developing Countries

Continued on next page

Table A.2 – continued from previous page

No.	Year	Authors	Title	Source
P76	2017	Walaa Alajali et al	On-Street Car Parking Prediction in Smart City: A Multi-source Data Analysis in Sensor-Cloud Environment	SpaCCS: International Conference on Security, Privacy and Anonymity in Computation, Communication and Storage
P77	2018	Suresh Limkar et al	Small Effort to Build Pune as a Smart City: Smart Real-Time Road Condition Detection and Efficient Management System	Smart Computing and Informatics
P78	2019	Bartosz Pawłowicz et al	Smart City Traffic Monitoring System Based on 5G Cellular Network, RFID and Machine Learning	KKIO: KKIO Software Engineering Conference
P79	2020	G. S. Nagaraja et al	Spatial Data Infrastructures for Urban Governance Using High-Performance Computing for Smart City Applications	Proceedings of the Third International Conference on Smart Computing and Informatics, Volume 2
P80	2018	Doreswamy et al	Traffic Jams Detection and Congestion Avoidance in Smart City Using Parallel K-Means Clustering Algorithm	Proceedings of International Conference on Cognition and Recognition
P81	2019	Federico Concone et al	A Fog-Based Application for Human Activity Recognition Using Personal Smart Devices	ACM Transactions on Internet Technology
P82	2017	Chao Huang et al	An Unsupervised Approach to Inferring the Localness of People Using Incomplete Geotemporal Online Check-In Data	ACM Transactions on Intelligent Systems and Technology
P83	2019	Zipei Fan et al	Decentralized Attention-based Personalized Human Mobility Prediction	Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies
P84	2017	M. Mazhar Rathore et al	Hadoop-Based Intelligent Care System (HICS): Analytical Approach for Big Data in IoT	ACM Transactions on Internet Technology
P85	2017	Iman Azimi et al	HiCH: Hierarchical Fog-Assisted Computing Architecture for Healthcare IoT	ACM Transactions on Embedded Computing Systems
P86	2017	Catta Prandi et al	On the Need of Trustworthy Sensing and Crowdsourcing for Urban Accessibility in Smart City	ACM Transactions on Internet Technology

Continued on next page

Table A.2 – continued from previous page

No.	Year	Authors	Title	Source
P87	2018	Zipei Fan et al	Online Deep Ensemble Learning for Predicting Citywide Human Mobility	Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies
P88	2017	Ruilin Liu et al	Your Search Path Tells Others Where to Park: Towards Fine-Grained Parking Availability Crowdsourcing Using Parking Decision Models	Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies
P89	2016	Debopriya Ghosh et al	Big Data-based Smart City Platform: Real-Time Crime Analysis	Proceedings of the 17th International Digital Government Research Conference on Digital Government Research
P90	2019	Isha Pradhan et al	Exploratory Data Analysis and Crime Prediction for Smart Cities	Proceedings of the 23rd International Database Applications & Engineering Symposium
P91	2015	Yaoci Han et al	An Ontology-oriented Decision Support System for Emergency Management Based on Information Fusion	Proceedings of the 1st ACM SIGSPATIAL International Workshop on the Use of GIS in Emergency Management
P92	2017	Boudhir Anouar Abdelhakim et al	Big data architecture for decision making in protocols and medications assignment	Proceedings of the Mediterranean Symposium on Smart City Application
P93	2018	Ashraf Tahat et al	A Smart City Environmental Monitoring Network and Analysis Relying on Big Data Techniques	Proceedings of the 2018 International Conference on Software Engineering and Information Management
P94	2019	Chrysovalantis Anastasiou et al	ADMSv2: A Modern Architecture for Transportation Data Management and Analysis	Proceedings of the 2nd ACM SIGSPATIAL International Workshop on Advances on Resilient and Intelligent Cities

Continued on next page

Table A.2 – continued from previous page

No.	Year	Authors	Title	Source
P95	2019	Pruthvish Rajput et al	Advanced Urban Public Transportation System for Indian Scenarios	Proceedings of the 20th International Conference on Distributed Computing and Networking
P96	2017	Deepti Goel et al	An IoT Approach for Context-aware Smart Traffic Management Using Ontology	Proceedings of the International Conference on Web Intelligence
P97	2018	Tengyue Li et al	Counting Passengers in Public Buses by Sensing Carbon Dioxide Concentration: Data Collection and Machine Learning	Proceedings of the 2018 2nd International Conference on Big Data and Internet of Things
P98	2017	Ellen Mitsopoulou et al	Efficient Parking Allocation for SmartCities	Proceedings of the 10th International Conference on PErvasive Technologies Related to Assistive Environments
P99	2019	Mohamed Yacine Gheraibia et al	Intelligent Mobile-Based Recommender System Framework for Smart Freight Transport	Proceedings of the 5th EAI International Conference on Smart Objects and Technologies for Social Good
P100	2017	Lamia Karim et al	Real time analytics of urban congestion trajectories on Hadoop-MongoDB cloud ecosystem	Proceedings of the Second International Conference on Internet of things, Data and Cloud Computing
P101	2019	Zheyi Pan et al	Urban Traffic Prediction from Spatio-Temporal Data Using Deep Meta Learning	Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining
P102	2019	Minh-Tri Ho et al	Visual Assistant for Crowdsourced Anomaly Event Recognition in Smart City	Proceedings of the Tenth International Symposium on Information and Communication Technology

Continued on next page

Table A.2 – continued from previous page

No.	Year	Authors	Title	Source
P103	2018	Hajar Khallouki et al	An Ontology-based Context awareness for Smart Tourism Recommendation System	Proceedings of the International Conference on Learning and Optimization Algorithms: Theory and Applications

Table A.3: Manual search – List of primary studies

ID	Year	Authors	Title	Source
P1	2015	Bin Cheng et al	Building a Big Data Platform for Smart Cities: Experience and Lessons from Santander	IEEE International Congress on Big Data
P2	2016	Dan Puiu et al	CityPulse: Large Scale Data Analytics Framework for Smart Cities	IEEE Access
P3	2013	Villanueva et al	Civitas: The Smart City Middleware, from Sensors to Big Data	Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS)
P4	2014	Kenji Tei et al	ClouT : Cloud of Things for Empowering the Citizen Clout in Smart Cities	IEEE World Forum on Internet of Things (WF-IoT)
P5	2016	Maurilio Zuccalà et al	D4.2 Urban Sharing Platform Reference Model	Tech. Rep. from the EU project Sharing Cities: H2020-SCC-2015 SHAR-LLM
P6	2014	Apolinarski et al	The GAMBAS Middleware and SDK for Smart City Applications	IEEE PerCom Workshops
P7	2019	Arthur de M. Del Esposte et al	Design and evaluation of a scalable smart city software platform with large-scale simulations	Elsevier Future Generation Computer Systems
P8	2014	Jaeho Kim et al	OpenIoT: An Open Service Framework for the Internet of Things	2014 IEEE World Forum on Internet of Things (WF-IoT)
P9	2013	Asma Elmangoush et al	Design Aspects for a Reference M2M Communication Platform for Smart Cities	9th International Conference on Innovations in Information Technology (IIT)
P10	2014	Andrea Zanella et al	Internet of Things for Smart Cities	IEEE Internet of Things Journal

Continued on next page

Table A.3 – continued from previous page

No.	Year	Authors	Title	Source
P11	2019	Arianna Brutti et al	Smart City Platform Specification: A Modular Approach to Achieve Interoperability in Smart Cities	Springer book The Internet of Things for Smart Urban Ecosystems
P12	2016	Amir Sinaeepourfard et al	Sentilo–Sensor and Actuator Platform for smart Cities	Mediterranean Ad Hoc Networking Workshop (Med-Hoc-Net)
P13	2014	Luis Sanchez et al	SmartSantander: IoT experimentation over a smart city testbed	Elsevier Computer Networks
P14	2013	Giuseppe Anastasi et al	Urban and Social Sensing for Sustainable Mobility in Smart Cities	IFIP
P15	2013	Levent Gurgen et al	Self-aware cyber-physical systems and applications in smart buildings and cities	Conference on Design, Automation, and Test in Europe

Bibliography

[1] *European innovation partnership on smart-cities and communication.* http://ec.europa.eu/eip/smartcities/indexen.htm. [Online; accessed 30 Mar 2018]. 2018.

[2] J Ramon Gil-Garcia, Theresa A Pardo, and Taewoo Nam. "What makes a city smart? Identifying core components and proposing an integrative and comprehensive conceptualization". In: *Information Polity* 20.1 (2015), pp. 61–87.

[3] Hadi Habibzadeh, Cem Kaptan, Tolga Soyata, Burak Kantarci, and Azzedine Boukerche. "Smart city system design: A comprehensive study of the application and data planes". In: *ACM Computing Surveys (CSUR)* 52.2 (2019), pp. 1–38.

[4] Yosra Hajjaji, Wadii Boulila, Imed Riadh Farah, Imed Romdhani, and Amir Hussain. "Big data and IoT-based applications in smart environments: A systematic review". In: *Computer Science Review* 39 (2021), p. 100318.

[5] Eiman Al Nuaimi, Hind Al Neyadi, Nader Mohamed, and Jameela Al-Jaroodi. "Applications of big data to smart cities". In: *Journal of Internet Services and Applications* 6.1 (2015), pp. 1–15.

[6] V Bassoo, V Ramnarain-Seetohul, V Hurbungs, TP Fowdur, and Y Beeharry. "Big data analytics for smart cities". In: *Internet of Things and Big Data Analytics Toward Next-Generation Intelligence.* Springer, 2018, pp. 359–379.

[7] John Israilidis, Kayode Odusanya, and Muhammad Usman Mazhar. "Exploring knowledge management perspectives in smart city research: A review and future research agenda". In: *International Journal of Information Management* 56 (2021), p. 101989. ISSN: 0268-4012. DOI: https://doi.org/10.1016/j.ijinfomgt.2019.07.015. URL: http://www.sciencedirect.com/science/article/pii/S0268401219302245.

[8] Big Data. “Changing the Way Businesses Compete and Operate”. In: *Insights on Governance, Risk and Compliance* (2014).

[9] Hadi Habibzadeh, Cem Kaptan, Tolga Soyata, Burak Kantarci, and Azzedine Boukerche. “Smart City System Design: A Comprehensive Study of the Application and Data Planes”. In: *ACM Comput. Surv.* 52.2 (May 2019). ISSN: 0360-0300. URL: `https://doi.org/10.1145/3309545`.

[10] José-Ricardo López-Robles, José-Ramón Otegi-Olaso, I Porto Gómez, and Manuel-Jesús Cobo. “30 years of intelligence models in management and business: A bibliometric review”. In: *International journal of information management* 48 (2019), pp. 22–38.

[11] Dusko Radulovic, Srdjan Skok, and Vedran Kirincic. “Energy efficiency public lighting management in the cities”. In: *Energy* 36.4 (2011), pp. 1908–1915.

[12] Marijana Zekić-Sušac, Saša Mitrović, and Adela Has. “Machine learning based system for managing energy efficiency of public sector as an approach towards smart cities”. In: *International journal of information management* 58 (2021), p. 102074.

[13] Xiaodong Cao, Xilei Dai, and Junjie Liu. “Building energy-consumption status worldwide and the state-of-the-art technologies for zero-energy buildings during the past decade”. In: *Energy and buildings* 128 (2016), pp. 198–213.

[14] Isaac Triguero, Diego Garcia-Gil, Jesus Maillo, Julian Luengo, Salvador Garcia, and Francisco Herrera. “Transforming big data into smart data: An insight on the use of the k-nearest neighbors algorithm to obtain quality data”. In: *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 9.2 (2019), e1289.

[15] Alexander Lenk, Leif Bonorden, Astrid Hellmanns, Nico Roedder, and Stefan Jaehnichen. “Towards a taxonomy of standards in smart data”. In: *2015 IEEE International Conference on Big Data (Big Data)*. IEEE. 2015, pp. 1749–1754.

[16] Ali Davoudian and Mengchi Liu. “Big Data Systems: A Software Engineering Perspective”. In: *ACM Comput. Surv.* 53.5 (Sept. 2020). ISSN: 0360-0300. DOI: `10.1145/3408314`. URL: `https://doi.org/10.1145/3408314`.

[17] Samira Pouyanfar, Yimin Yang, Shu-Ching Chen, Mei-Ling Shyu, and S. S. Iyengar. "Multimedia Big Data Analytics: A Survey". In: *ACM Comput. Surv.* 51.1 (Jan. 2018). ISSN: 0360-0300. DOI: `10.1145/3150226`. URL: `https://doi.org/10.1145/3150226`.

[18] Marijana Zekić-Sušac, Saša Mitrović, and Adela Has. "Machine learning based system for managing energy efficiency of public sector as an approach towards smart cities". In: *International Journal of Information Management* (2020), p. 102074. ISSN: 0268-4012. DOI: `https://doi.org/10.1016/j.ijinfomgt.2020.102074`. URL: `http://www.sciencedirect.com/science/article/pii/S0268401219302968`.

[19] Ken Peffers, Tuure Tuunanen, Marcus A Rothenberger, and Samir Chatterjee. "A design science research methodology for information systems research". In: *Journal of management information systems* 24.3 (2007), pp. 45–77.

[20] Kai Petersen, Sairam Vakkalanka, and Ludwik Kuzniarz. "Guidelines for conducting systematic mapping studies in software engineering: An update". In: *Information and Software Technology* 64 (2015), pp. 1–18.

[21] Ali Emrouznejad. *Big data optimization: recent developments and challenges*. Vol. 18. Springer, 2016.

[22] Doug Laney et al. "3D data management: Controlling data volume, velocity and variety". In: *META group research note* 6.70 (2001), p. 1.

[23] Marcos D Assunção, Rodrigo N Calheiros, Silvia Bianchi, Marco AS Netto, and Rajkumar Buyya. "Big Data computing and clouds: Trends and future directions". In: *Journal of parallel and distributed computing* 79 (2015), pp. 3–15.

[24] Erik Meijer. "The world according to LINQ". In: *Communications of the ACM* 54.10 (2011), pp. 45–51.

[25] Min Chen, Shiwen Mao, and Yunhao Liu. "Big data: A survey". In: *Mobile networks and applications* 19.2 (2014), pp. 171–209.

[26] James Manyika, Michael Chui, Brad Brown, Jacques Bughin, Richard Dobbs, Charles Roxburgh, Angela Hung Byers, et al. *Big data: The next frontier for innovation, competition, and productivity*. McKinsey Global Institute, 2011.

[27] European Commission. "The EU data protection reform and Big Data [Fact sheet]". In: (2015).

[28] Hsinchun Chen, Roger HL Chiang, and Veda C Storey. "Business intelligence and analytics: From big data to big impact". In: *MIS quarterly* (2012), pp. 1165–1188.

[29] Ohbyung Kwon, Namyeon Lee, and Bongsik Shin. "Data quality management, data usage experience and acquisition intention of big data analytics". In: *International journal of information management* 34.3 (2014), pp. 387–394.

[30] VS Thiyagarajan and K Venkatachalapathy. "Isolating values from big data with the help of four V'S". In: *International Journal of Research in Engineering and Technology* 4.1 (2014), pp. 132–135.

[31] Hiba Jasim Hadi, Ammar Hameed Shnain, Sarah Hadishaheed, and Azizahbt Haji Ahmad. "Big data and five V's characteristics". In: *International Journal of Advances in Electronics and Computer Science* 2.1 (2015), pp. 16–23.

[32] J Anuradha et al. "A brief introduction on Big Data 5Vs characteristics and Hadoop technology". In: *Procedia computer science* 48 (2015), pp. 319–324.

[33] Muhammad Fahim Uddin, Navarun Gupta, et al. "Seven V's of Big Data understanding Big Data to extract value". In: *Proceedings of the 2014 zone 1 conference of the American Society for Engineering Education.* IEEE. 2014, pp. 1–5.

[34] Suhail Sami Owais and Nada Sael Hussein. "Extract five categories CPIVW from the 9V's characteristics of the big data". In: *International Journal of Advanced Computer Science and Applications* 7.3 (2016), pp. 254–258.

[35] Andrew McAfee, Erik Brynjolfsson, Thomas H Davenport, DJ Patil, and Dominic Barton. "Big data: the management revolution". In: *Harvard business review* 90.10 (2012), pp. 60–68.

[36] V Rajaraman. "Big data analytics". In: *Resonance* 21.8 (2016), pp. 695–716.

[37] Nawsher Khan, Mohammed Alsaqer, Habib Shah, Gran Badsha, Aftab Ahmad Abbasi, and Soulmaz Salehian. "The 10 Vs, issues and challenges of big data". In: *Proceedings of the 2018 international conference on big data and education.* 2018, pp. 52–56.

[38] Borko Furht and Flavio Villanustre. “Big data technologies and applications”. In: (2016).

[39] In Lee. “Big data: Dimensions, evolution, impacts, and challenges”. In: *Business horizons* 60.3 (2017), pp. 293–303.

[40] Yuri Demchenko, Paola Grosso, Cees De Laat, and Peter Membrey. “Addressing big data issues in scientific data infrastructure”. In: *2013 International conference on collaboration technologies and systems (CTS)*. IEEE. 2013, pp. 48–55.

[41] Victoria Rubin and Tatiana Lukoianova. “Veracity roadmap: Is big data objective, truthful and credible?” In: *Advances in Classification Research Online* 24.1 (2013), p. 4.

[42] Tom White. *Hadoop: The definitive guide, chapter Meet Hadoop*. 2015.

[43] Nupur N Mall, Sheetal Rana, et al. “Overview of big data and Hadoop”. In: *Imperial Journal of Interdisciplinary Research* 2.5 (2016), pp. 1399–1406.

[44] Debi Prasanna Acharjya and Kauser Ahmed. “A survey on big data analytics: challenges, open research issues and tools”. In: *International Journal of Advanced Computer Science and Applications* 7.2 (2016), pp. 511–518.

[45] Holden Karau. “Fastdata Processing with Spark”. In: (2013).

[46] Abhay Kumar Bhadani and Dhanya Jothimani. “Big data: challenges, opportunities, and realities”. In: *Effective big data management and opportunities for implementation*. IGI Global, 2016, pp. 1–24.

[47] Holden Karau, Andy Konwinski, Patrick Wendell, and Matei Zaharia. *Learning spark: lightning-fast big data analysis*. " O’Reilly Media, Inc.", 2015.

[48] Sherif Sakr. “General-purpose big data processing systems”. In: *Big Data 2.0 Processing Systems*. Springer, 2016, pp. 15–39.

[49] Bahaaldine Azarmi. “Scalable big data architecture”. In: *A Practitioner’s Guide to Choosing Relevant Big Data Architecture. Apress, Berkeley* (2016).

[50] Sara Landset, Taghi M Khoshgoftaar, Aaron N Richter, and Tawfiq Hasanin. “A survey of open source tools for machine learning with big data in the Hadoop ecosystem”. In: *Journal of Big Data* 2.1 (2015), pp. 1–36.

[51] Amy N Langville and Carl D Meyer. *Google’s PageRank and beyond*. Princeton university press, 2011.

[52] Ashwin Belle, Raghuram Thiagarajan, SM Soroushmehr, Fatemeh Navidi, Daniel A Beard, and Kayvan Najarian. “Big data analytics in healthcare”. In: *BioMed research international* 2015 (2015).

[53] Ciprian Dobre and Fatos Xhafa. “Intelligent services for big data science”. In: *Future generation computer systems* 37 (2014), pp. 267–281.

[54] Franz Loewenherz, Victor Bahl, and Yinhai Wang. “Video analytics towards vision zero”. In: *Institute of Transportation Engineers. ITE Journal* 87.3 (2017), p. 25.

[55] Ahmed Oussous, Fatima-Zahra Benjelloun, Ayoub Ait Lahcen, and Samir Belfkih. “Big Data technologies: A survey”. In: *Journal of King Saud University-Computer and Information Sciences* 30.4 (2018), pp. 431–448.

[56] The-Hien Dang-Ha, Roland Olsson, and Hao Wang. “The role of big data on smart grid transition”. In: *2015 IEEE International Conference on Smart City/SocialCom/SustainCom (SmartCity)*. IEEE. 2015, pp. 33–39.

[57] Sarah Brayne. “Big data surveillance: The case of policing”. In: *American sociological review* 82.5 (2017), pp. 977–1008.

[58] Joseph E Beck and Jack Mostow. “How who should practice: Using learning decomposition to evaluate the efficacy of different types of practice for different types of students”. In: *International conference on intelligent tutoring systems*. Springer. 2008, pp. 353–362.

[59] Simon Elias Bibri. “The IoT for smart sustainable cities of the future: An analytical framework for sensor-based big data applications for environmental sustainability”. In: *Sustainable cities and society* 38 (2018), pp. 230–253.

[60] Ibrahim Abaker Targio Hashem, Victor Chang, Nor Badrul Anuar, Kayode Adewole, Ibrar Yaqoob, Abdullah Gani, Ejaz Ahmed, and Haruna Chiroma. “The role of big data in smart city”. In: *International Journal of Information Management* 36.5 (2016), pp. 748–758.

[61] Amr Azzam et al. “The CitySPIN Platform: A CPSS Environment for City-Wide Infrastructures”. In: (2019).

[62] Dan Puiu et al. “Citypulse: Large scale data analytics framework for smart cities”. In: *IEEE Access* 4 (2016), pp. 1086–1108.

[63] Pedro Martins, Daniel Albuquerque, Cristina Wanzeller, Filipe Caldeira, Paulo Tomé, and Filipe Sá. “CityAction a Smart-City Platform Architecture”. In: *Future of Information and Communication Conference*. Springer. 2019, pp. 217–236.

[64] Mohamed Dbouk, Mariam Hakim, and Ihab Sbeity. “CityPro: From Big-Data to Intelligent-Data; a Smart Approach.” In: *BDCSIntell.* 2018, pp. 100–106.

[65] Paula Ta-Shma, Adnan Akbar, Guy Gerson-Golan, Guy Hadash, Francois Carrez, and Klaus Moessner. “An ingestion and analytics architecture for iot applied to smart city use cases”. In: *IEEE Internet of Things Journal* 5.2 (2017), pp. 765–774.

[66] Sanny Schmid, Ilias Gerostathopoulos, Christian Prehofer, and Tomas Bures. “Self-adaptation based on big data analytics: a model problem and tool”. In: *2017 IEEE/ACM 12th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*. IEEE. 2017, pp. 102–108.

[67] Marijana Zekić-Sušac, Saša Mitrović, and Adela Has. “Machine learning based system for managing energy efficiency of public sector as an approach towards smart cities”. In: *International Journal of Information Management* (2020), p. 102074.

[68] S Jangili and K Bikshalu. “Smart grid administration using big data and wireless sensor networks”. In: *Int. J. Adv. Res. Sci. Eng* 6 (2017), pp. 629–636.

[69] Vangelis Marinakis and Haris Doukas. “An advanced IoT-based system for intelligent energy management in buildings”. In: *Sensors* 18.2 (2018), p. 610.

[70] Željko Tomšić, Ivan Gašić, and Goran Čačić. “ENERGY MANAGEMENT IN THE PUBLIC BUILDING SECTOR–ISGE/ISEMIC MODEL”. In: *Energija* 64.1-4 (2015), pp. 0–0.

[71] Antonio Galicia, R Talavera-Llames, A Troncoso, Irena Koprinska, and Francisco Martinez-Alvarez. “Multi-step forecasting for big data time series based on ensemble learning”. In: *Knowledge-Based Systems* 163 (2019), pp. 830–841.

[72] José F Torres, Antonio Galicia, A Troncoso, and Francisco Martinez-Alvarez. “A scalable approach based on deep learning for big data time series forecasting”. In: *Integrated Computer-Aided Engineering* 25.4 (2018), pp. 335–348.

[73] Abdul-Rahman Al-Ali, Imran A Zualkernan, Mohammed Rashid, Ragini Gupta, and Mazin Alikarar. "A smart home energy management system using IoT and big data analytics approach". In: *IEEE Transactions on Consumer Electronics* 63.4 (2017), pp. 426–434.

[74] Gianfranco Gagliardi, Alessandro Casavola, Marco Lupia, Gianni Cario, Francesco Tedesco, Fabrizio Lo Scudo, Francesco Cicchello Gaccio, and Antonio Augimeri. "A smart city adaptive lighting system". In: *2018 Third International Conference on Fog and Mobile Edge Computing (FMEC)*. IEEE. 2018, pp. 258–263.

[75] Mariagrazia Leccisi, Fabio Leccese, Fabio Moretti, Laura Blaso, Arianna Brutti, and Nicoletta Gozo. "An iot application for industry 4.0: a new and efficient public lighting management model". In: *2020 IEEE International Workshop on Metrology for Industry 4.0 & IoT*. IEEE. 2020, pp. 669–673.

[76] Y-K Juan, Ling Wang, Jie Wang, James O Leckie, and K-M Li. "A decision-support system for smarter city planning and management". In: *IBM Journal of Research and Development* 55.1.2 (2011), pp. 3–1.

[77] Yuriy Brun et al. "Engineering self-adaptive systems through feedback loops". In: *Software engineering for self-adaptive systems*. Springer, 2009, pp. 48–70.

[78] Rogério De Lemos et al. "Software engineering for self-adaptive systems: A second research roadmap". In: *Software Engineering for Self-Adaptive Systems II*. Springer, 2013, pp. 1–32.

[79] Eduardo Felipe Zambom Santana, Ana Paula Chaves, Marco Aurelio Gerosa, Fabio Kon, and Dejan S. Milojicic. "Software Platforms for Smart Cities: Concepts, Requirements, Challenges, and a Unified Reference Architecture". In: *ACM Comput. Surv.* 50.6 (Nov. 2017). ISSN: 0360-0300. URL: `https://doi.org/10.1145/3124391`.

[80] Welington M. da Silva, Alexandre Alvaro, Gustavo H. R. P. Tomas, Ricardo A. Afonso, Kelvin L. Dias, and Vinicius C. Garcia. "Smart Cities Software Architectures: A Survey". In: *Proceedings of the 28th Annual ACM Symposium on Applied Computing*. SAC '13. Coimbra, Portugal: Association for Computing Machinery, 2013, pp. 1722–1727. ISBN: 9781450316569. URL: `https://doi.org/10.1145/2480362.2480688`.

[81] ChuanTao Yin, Zhang Xiong, Hui Chen, JingYuan Wang, Daven Cooper, and Bertrand David. “A literature survey on smart cities”. In: *Science China Information Sciences* 58.10 (2015), pp. 1–18.

[82] Ruben Sánchez-Corcuera, Adrián Nuñez-Marcos, Jesus Sesma-Solance, Aritz Bilbao-Jayo, Rubén Mulero, Unai Zulaika, Gorka Azkune, and Aitor Almeida. “Smart cities survey: Technologies, application domains and challenges for the cities of the future”. In: *International Journal of Distributed Sensor Networks* 15.6 (2019), p. 1550147719853984.

[83] Henry Muccini, Mohammad Sharaf, and Danny Weyns. “Self-adaptation for cyber-physical systems: a systematic literature review”. In: *Proceedings of the 11th International Symposium on Software Engineering for Adaptive and Self-Managing Systems, SEAMS@ICSE 2016, Austin, Texas, USA, May 14-22, 2016*. ACM, 2016, pp. 75–81. ISBN: 978-1-4503-4187-5. URL: `https://doi.org/10.1145/2897053.2897069`.

[84] E. M. Grua, I. Malavolta, and P. Lago. “Self-Adaptation in Mobile Apps: a Systematic Literature Study”. In: *2019 IEEE/ACM 14th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*. 2019, pp. 51–62.

[85] Kai Petersen, Sairam Vakkalanka, and Ludwik Kuzniarz. “Guidelines for conducting systematic mapping studies in software engineering: An update”. In: *Information and Software Technology* 64 (2015), pp. 1–18. ISSN: 0950-5849. URL: `http://www.sciencedirect.com/science/article/pii/S0950584915000646`.

[86] Ali Mubashir and Patrizia Scandurra. *The replication package of the study* Self-Adaptation and Big Data Analytics in Software Platforms for Smart Cities: a Systematic Literature Review. `https://drive.google.com/drive/folders/1LDMhKzLZO4goyWJy5XXU8nMS61Ep3reV?usp=sharing`. 2020.

[87] Jeffrey O Kephart and M David. “Chess”. In: *The vision of autonomic computing* 36 (2003), pp. 41–50.

[88] Christian Krupitzer, Felix Maximilian Roth, Sebastian VanSyckel, Gregor Schiele, and Christian Becker. “A Survey on Engineering Approaches for Self-adaptive Systems”. In: *Pervasive Mob. Comput.* 17.PB (Feb. 2015), pp. 184–206. ISSN: 1574-1192.

[89] Arianna Brutti, Piero De Sabbata, Angelo Frascella, Nicola Gessa, Raffaele Ianniello, Cristiano Novelli, Stefano Pizzuti, and Giovanni Ponti. “Smart City Platform Specification: A Modular Approach to Achieve Interoperability in Smart Cities”. In: *The Internet of Things for Smart Urban Ecosystems*. Ed. by Franco Cicirelli, Antonio Guerrieri, Carlo Mastroianni, Giandomenico Spezzano, and Andrea Vinci. Springer, 2019, pp. 25–50. ISBN: 978-3-319-96549-9. URL: https://doi.org/10.1007/978-3-319-96550-5%5C_2.

[90] Pedro Martins, Daniel Albuquerque, Cristina Wanzeller, Filipe Caldeira, Paulo Tomé, and Filipe Sá. “CityAction a Smart-City Platform Architecture”. In: *Advances in Information and Communication*. Ed. by Kohei Arai and Rahul Bhatia. Cham: Springer International Publishing, 2020, pp. 217–236.

[91] *European project Sharing Cities. Deliverable D4.2 Urban Sharing Platform Reference Model*. http://www.sharingcities.eu/. 2016.

[92] Maurilio Zuccalà and Emiliano Sergio Verga. “Enabling Energy Smart Cities through Urban Sharing Ecosystems”. In: *Energy Procedia* 111 (2017). 8th International Conference on Sustainability in Energy and Buildings, SEB-16, 11-13 September 2016, Turin, Italy, pp. 826–835. ISSN: 1876-6102. URL: http://www.sciencedirect.com/science/article/pii/S1876610217302783.

[93] Arthur de M. Del Esposte, Eduardo F.Z. Santana, Lucas Kanashiro, Fabio M. Costa, Kelly R. Braghetto, Nelson Lago, and Fabio Kon. “Design and evaluation of a scalable smart city software platform with large-scale simulations”. In: *Future Generation Computer Systems* 93 (2019), pp. 427–441. ISSN: 0167-739X. URL: http://www.sciencedirect.com/science/article/pii/S0167739X18307301.

[94] A. Auger, E. Exposito, and E. Lochin. “iQAS: An integration platform for QoI assessment as a service for smart cities”. In: *2016 IEEE 3rd World Forum on Internet of Things (WF-IoT)*. 2016, pp. 88–93.

[95] Sotiris Zygiaris. “Smart City Reference Model: Assisting Planners to Conceptualize the Building of Smart City Innovation Ecosystems”. In: *Journal of the Knowledge Economy* 4.2 (June 2013), pp. 217–231. URL: https://ideas.repec.org/a/spr/jknowl/v4y2013i2p217-231.html.

[96] Yilin Huang, Giacomo Poderi, Sanja Scepanovic, Hanna Hasselqvist, Martijn Warnier, and Frances M. T. Brazier. “Embedding Internet-of-Things in Large-Scale Socio-technical Systems: A Community-Oriented Design in Future Smart Grids”. In: *The Internet of Things for Smart Urban Ecosystems.* Ed. by Franco Cicirelli, Antonio Guerrieri, Carlo Mastroianni, Giandomenico Spezzano, and Andrea Vinci. Springer, 2019, pp. 125–150. ISBN: 978-3-319-96549-9. URL: `https://doi.org/10.1007/978-3-319-96550-5%5C_6`.

[97] Mazeiar Salehie and Ladan Tahvildari. “Self-Adaptive Software: Landscape and Research Challenges”. In: *ACM Trans. Auton. Adapt. Syst.* 4.2 (May 2009). ISSN: 1556-4665. URL: `https://doi.org/10.1145/1516533.1516538`.

[98] F. Sivrikaya, N. Ben-Sassi, X. Dang, O. C. Görür, and C. Kuster. “Internet of Smart City Objects: A Distributed Framework for Service Discovery and Composition”. In: *IEEE Access* 7 (2019), pp. 14434–14454.

[99] L. Gurgen, O. Gunalp, Y. Benazzouz, and M. Gallissot. “Self-aware cyber-physical systems and applications in smart buildings and cities”. In: *2013 Design, Automation Test in Europe Conference Exhibition (DATE).* 2013, pp. 1149–1154.

[100] Gabriel A. Moreno, Javier Cámara, David Garlan, and Bradley Schmerl. “Proactive Self-Adaptation under Uncertainty: A Probabilistic Model Checking Approach”. In: *Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering.* ESEC/FSE 2015. Bergamo, Italy: Association for Computing Machinery, 2015, pp. 1–12. ISBN: 9781450336758. DOI: `10.1145/2786805.2786853`. URL: `https://doi.org/10.1145/2786805.2786853`.

[101] Federico Quin, Danny Weyns, Thomas Bamelis, Sarpreet Singh Buttar, and Sam Michiels. “Efficient analysis of large adaptation spaces in self-adaptive systems using machine learning”. In: *Proceedings of the 14th International Symposium on Software Engineering for Adaptive and Self-Managing Systems, SEAMS@ICSE 2019, Montreal, QC, Canada, May 25-31, 2019.* Ed. by Marin Litoiu, Siobhán Clarke, and Kenji Tei. ACM, 2019, pp. 1–12. ISBN: 978-1-7281-3368-3. DOI: `10.1109/SEAMS.2019.00011`. URL: `https://doi.org/10.1109/SEAMS.2019.00011`.

[102] Paula Ta-Shma, Adnan Akbar, Guy Gerson-Golan, Guy Hadash, François Carrez, and Klaus Moessner. “An Ingestion and Analytics Architecture for IoT Applied to Smart City Use Cases”. In: *IEEE Internet of Things Journal*

5.2 (2018), pp. 765–774. URL: https://doi.org/10.1109/JIOT.2017.2722378.

[103] S. Schmid, I. Gerostathopoulos, C. Prehofer, and T. Bures. "Self-Adaptation Based on Big Data Analytics: A Model Problem and Tool". In: *2017 IEEE/ACM 12th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*. 2017, pp. 102–108.

[104] D. Puiu et al. "CityPulse: Large Scale Data Analytics Framework for Smart Cities". In: *IEEE Access* 4 (2016), pp. 1086–1108.

[105] Pengcheng Zhang, Wennan Cao, and Henry Muccini. "Quality Assurance Technologies of Big Data Applications: A Systematic Literature Review". In: *CoRR* abs/2002.01759 (2020). arXiv: 2002.01759. URL: https://arxiv.org/abs/2002.01759.

[106] Charith Perera, Yongrui Qin, Julio C. Estrella, Stephan Reiff-Marganiec, and Athanasios V. Vasilakos. "Fog Computing for Sustainable Smart Cities: A Survey". In: *ACM Comput. Surv.* 50.3 (June 2017). ISSN: 0360-0300. DOI: 10.1145/3057266. URL: https://doi.org/10.1145/3057266.

[107] Mirko D'Angelo. "Decentralized Self-Adaptive Computing at the Edge". In: *Proceedings of the 13th International Conference on Software Engineering for Adaptive and Self-Managing Systems*. SEAMS '18. Gothenburg, Sweden: Association for Computing Machinery, 2018, pp. 144–148. ISBN: 9781450357159. URL: https://doi.org/10.1145/3194133.3194160.

[108] CL Philip Chen and Chun-Yang Zhang. "Data-intensive applications, challenges, techniques and technologies: A survey on Big Data". In: *Information sciences* 275 (2014), pp. 314–347.

[109] Elias Andersson and Patrik Thollander. "Key performance indicators for energy management in the Swedish pulp and paper industry". In: *Energy Strategy Reviews* 24 (2019), pp. 229–235.

[110] Gökan May, Ilaria Barletta, Bojan Stahl, and Marco Taisch. "Energy management in production: A novel method to develop key performance indicators for improving energy efficiency". In: *Applied Energy* 149 (2015), pp. 46–61.

[111] Fayas Malik Kanchiralla, Noor Jalo, Simon Johnsson, Patrik Thollander, and Maria Andersson. “Energy End-Use Categorization and Performance Indicators for Energy Management in the Engineering Industry”. In: *Energies* 13.2 (2020), p. 369.

[112] Lavinia Chiara Tagliabue, Fulvio Re Cecconi, Nicola Moretti, Stefano Rinaldi, Paolo Bellagente, and Angelo Luigi Camillo Ciribini. “Security Assessment of Urban Areas through a GIS-Based Analysis of Lighting Data Generated by IoT Sensors”. In: *Applied Sciences* 10.6 (2020), p. 2174.

[113] António Grilo, Augusto Casaca, Mário Nunes, Alberto Bernardo, Paulo Rodrigues, and João Pinto Almeida. “A management system for low voltage grids”. In: *2017 IEEE Manchester PowerTech*. IEEE. 2017, pp. 1–6.

[114] Raffaele Carli, Paolo Deidda, Mariagrazia Dotoli, and Roberta Pellegrino. “An urban control center for the energy governance of a smart city”. In: *Proceedings of the 2014 IEEE Emerging Technology and Factory Automation (ETFA)*. IEEE. 2014, pp. 1–7.

[115] Charu C Aggarwal. *Outlier analysis. Springer Publishing Company, Incorporated, 2nd edition.* 2016.

[116] Cátia M Salgado, Carlos Azevedo, Hugo Proença, and Susana M Vieira. “Noise versus outliers”. In: *Secondary analysis of electronic health records* (2016), pp. 163–183.

[117] Nikou Günnemann, Stephan Günnemann, and Christos Faloutsos. “Robust multivariate autoregression for anomaly detection in dynamic product ratings”. In: *Proceedings of the 23rd international conference on World wide web*. 2014, pp. 361–372.

[118] V Chandola and A Banerjee. “Kumar, v.," Anomaly Detection: A Survey”. In: *ACM Computing Surveys* 41.3 (2009).

[119] Frank E Grubbs. “Procedures for detecting outlying observations in samples”. In: *Technometrics* 11.1 (1969), pp. 1–21.

[120] Douglas M Hawkins. *Identification of outliers*. Vol. 11. Springer, 1980.

[121] Mubashir Ali, Patrizia Scandurra, Fabio Moretti, Laura Blaso, Mariagrazia Leccisi, and Fabio Leccese. “From Big Data to Smart Data-centric Software Architectures for City Analytics: the case of the PELL Smart City Platform”. In: *2021 IEEE International Conference on Smart Data Services (SMDS)*. IEEE. 2021, pp. 95–104.

[122] Mohammad Braei and Sebastian Wagner. “Anomaly detection in univariate time-series: A survey on the state-of-the-art”. In: *arXiv preprint arXiv:2004.00433* (2020).

[123] Iqbal H Sarker. “A machine learning based robust prediction model for real-life mobile phone data”. In: *Internet of Things* 5 (2019), pp. 180–193.

[124] Victoria Hodge and Jim Austin. “A survey of outlier detection methodologies”. In: *Artificial intelligence review* 22.2 (2004), pp. 85–126.

[125] Varun Chandola, Arindam Banerjee, and Vipin Kumar. “Survey of anomaly detection”. In: *ACM Computing Survey (CSUR)* 41.3 (2009), pp. 1–72.

[126] Raghavendra Chalapathy and Sanjay Chawla. “Deep learning for anomaly detection: A survey”. In: *arXiv preprint arXiv:1901.03407* (2019).

[127] Daniel B Araya, Katarina Grolinger, Hany F ElYamany, Miriam AM Capretz, and Girma Bitsuamlak. “An ensemble learning framework for anomaly detection in building energy consumption”. In: *Energy and Buildings* 144 (2017), pp. 191–206.

[128] Mete Çelik, Filiz Dadaşer-Çelik, and Ahmet Şakir Dokuz. “Anomaly detection in temperature data using dbscan algorithm”. In: *2011 international symposium on innovations in intelligent systems and applications*. IEEE. 2011, pp. 91–95.

[129] Hari Krishna Kanagala and VV Jaya Rama Krishnaiah. “A comparative study of K-Means, DBSCAN and OPTICS”. In: *2016 International Conference on Computer Communication and Informatics (ICCCI)*. IEEE. 2016, pp. 1–6.

[130] William Hurst, Casimiro A Curbelo Montañez, and Nathan Shone. “Time-Pattern Profiling from Smart Meter Data to Detect Outliers in Energy Consumption”. In: *IoT* 1.1 (2020), pp. 92–108.

[131] R Ranjith, J Joshan Athanesious, and V Vaidehi. “Anomaly detection using DBSCAN clustering technique for traffic video surveillance”. In: *2015 Seventh International Conference on Advanced Computing (ICoAC)*. IEEE. 2015, pp. 1–6.

[132] Gerhard Münz, Sa Li, and Georg Carle. “Traffic anomaly detection using k-means clustering”. In: *GI/ITG Workshop MMBnet*. 2007, pp. 13–14.

[133] Kashif Sultan, Hazrat Ali, and Zhongshan Zhang. “Call detail records driven anomaly detection and traffic prediction in mobile cellular networks”. In: *IEEE Access* 6 (2018), pp. 41728–41737.

[134] Kyung-A Yoon, Oh-Sung Kwon, and Doo-Hwan Bae. “An approach to outlier detection of software measurement data using the k-means clustering method”. In: *First International Symposium on Empirical Software Engineering and Measurement (ESEM 2007)*. IEEE. 2007, pp. 443–445.

[135] Vannel Zeufack, Donghyun Kim, Daehee Seo, and Ahyoung Lee. “An unsupervised anomaly detection framework for detecting anomalies in real time through network system’s log files analysis”. In: *High-Confidence Computing* 1.2 (2021), p. 100030.

[136] Rony Chowdhury Ripan, Iqbal H Sarker, Syed Md Minhaz Hossain, Md Musfique Anwar, Raza Nowrozy, Mohammed Moshiul Hoque, and Md Hasan Furhad. “A data-driven heart disease prediction model through K-means clustering-based anomaly detection”. In: *SN Computer Science* 2.2 (2021), pp. 1–12.

[137] Rashmi Kumari, MK Singh, R Jha, NK Singh, et al. “Anomaly detection in network traffic using K-mean clustering”. In: *2016 3rd International Conference on Recent Advances in Information Technology (RAIT)*. IEEE. 2016, pp. 387–393.

[138] Yassine Himeur, Khalida Ghanem, Abdullah Alsalemi, Faycal Bensaali, and Abbes Amira. “Artificial intelligence based anomaly detection of energy consumption in buildings: A review, current trends and new perspectives”. In: *Applied Energy* 287 (2021), p. 116601.

[139] Desh Deepak Sharma and SN Singh. “Aberration detection in electricity consumption using clustering technique”. In: *International Journal of Energy Sector Management* (2015).

[140] Pengyuan Wang and Manimaran Govindarasu. "Anomaly detection for power system generation control based on hierarchical dbscan". In: *2018 North American Power Symposium (NAPS)*. IEEE. 2018, pp. 1–5.

[141] Wei Mao, Xiu Cao, Tong Yan, Yongkang Zhang, et al. "Anomaly detection for power consumption data based on isolated forest". In: *2018 international conference on power system technology (POWERCON)*. IEEE. 2018, pp. 4169–4174.

[142] Kedi Zheng, Yi Wang, Qixin Chen, and Yuanpeng Li. "Electricity theft detecting based on density-clustering method". In: *2017 IEEE Innovative Smart Grid Technologies-Asia (ISGT-Asia)*. IEEE. 2017, pp. 1–6.

[143] Jaime Yeckle and Bo Tang. "Detection of electricity theft in customer consumption using outlier detection algorithms". In: *2018 1st international conference on data intelligence and security (ICDIS)*. IEEE. 2018, pp. 135–140.

[144] Bruno Rossi, Stanislav Chren, Barbora Buhnova, and Tomas Pitner. "Anomaly detection in smart grid data: An experience report". In: *2016 ieee international conference on systems, man, and cybernetics (smc)*. IEEE. 2016, pp. 002313–002318.

[145] Jecinta Mulongo, Marcellin Atemkeng, Theophilus Ansah-Narh, Rockefeller Rockefeller, Gabin Maxime Nguegnang, and Marco Andrea Garuti. "Anomaly detection in power generation plants using machine learning and neural networks". In: *Applied Artificial Intelligence* 34.1 (2020), pp. 64–79.

[146] Jea Nagi, Keem Siah Yap, Sieh Kiong Tiong, Syed Khaleel Ahmed, and AM Mohammad. "Detection of abnormalities and electricity theft using genetic support vector machines". In: *TENCON 2008-2008 IEEE region 10 conference*. IEEE. 2008, pp. 1–6.

[147] Leping Zhang, Lu Wan, Yong Xiao, Shuangquan Li, and Chengpeng Zhu. "Anomaly Detection method of Smart Meters data based on GMM-LDA clustering feature Learning and PSO Support Vector Machine". In: *2019 IEEE sustainable power and energy conference (ISPEC)*. IEEE. 2019, pp. 2407–2412.

[148] Christa Cody, Vitaly Ford, and Ambareen Siraj. "Decision tree learning for fraud detection in consumer energy consumption". In: *2015 IEEE 14th inter-*

national conference on machine learning and applications (ICMLA). IEEE. 2015, pp. 1175–1179.

[149] Klaus Kammerer, Burkhard Hoppenstedt, Rüdiger Pryss, Steffen Stökler, Johannes Allgaier, and Manfred Reichert. "Anomaly detections for manufacturing systems based on sensor data—insights into two challenging real-world production settings". In: *Sensors* 19.24 (2019), p. 5370.

[150] Matthias Reif, Markus Goldstein, Armin Stahl, and Thomas M Breuel. "Anomaly detection by combining decision trees and parametric densities". In: *2008 19th international conference on pattern recognition*. IEEE. 2008, pp. 1–4.

[151] Yu Weng, Ning Zhang, and Chunlei Xia. "Multi-agent-based unsupervised detection of energy consumption anomalies on smart campus". In: *IEEE Access* 7 (2018), pp. 2169–2178.

[152] Shuan Li, Yinghua Han, Xu Yao, Song Yingchen, Jinkuan Wang, and Qiang Zhao. "Electricity theft detection in power grids with deep learning and random forests". In: *Journal of Electrical and Computer Engineering* 2019 (2019).

[153] Khaled Alrawashdeh and Carla Purdy. "Toward an online anomaly intrusion detection system based on deep learning". In: *2016 15th IEEE international conference on machine learning and applications (ICMLA)*. IEEE. 2016, pp. 195–200.

[154] Brandon K Vaughn. *Data analysis using regression and multilevel/hierarchical models*. 2008.

[155] Chengliang Xu and Huanxin Chen. "A hybrid data mining approach for anomaly detection and evaluation in residential buildings energy data". In: *Energy and Buildings* 215 (2020), p. 109864.

[156] Mubashir Ali, Anees Baqir, Giuseppe Psaila, and Sayyam Malik. "Towards the discovery of influencers to follow in micro-blogs (twitter) by detecting topics in posted messages (tweets)". In: *Applied Sciences* 10.16 (2020), p. 5715.

[157] Nesime Tatbul, Tae Jun Lee, Stan Zdonik, Mejbah Alam, and Justin Gottschlich. "Precision and recall for time series". In: *arXiv preprint arXiv:1803.03639* (2018).

[158] J MacQueen. “Classification and analysis of multivariate observations”. In: *5th Berkeley Symp. Math. Statist. Probability.* 1967, pp. 281–297.

[159] James MacQueen et al. “Some methods for classification and analysis of multivariate observations”. In: *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability.* Vol. 1. 14. Oakland, CA, USA. 1967, pp. 281–297.

[160] Aristidis Likas, Nikos Vlassis, and Jakob J Verbeek. “The global k-means clustering algorithm”. In: *Pattern recognition* 36.2 (2003), pp. 451–461.

[161] Sam T Roweis and Lawrence K Saul. “Nonlinear dimensionality reduction by locally linear embedding”. In: *science* 290.5500 (2000), pp. 2323–2326.

[162] Hadley Wickham and Lisa Stryjewski. *40 years of boxplots.* Tech. rep. had.co.nz, 2012.

[163] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. “A density-based algorithm for discovering clusters in large spatial databases with noise.” In: *kdd.* Vol. 96. 34. 1996, pp. 226–231.

[164] Mihael Ankerst, Markus M Breunig, Hans-Peter Kriegel, and Jörg Sander. “OPTICS: Ordering points to identify the clustering structure”. In: *ACM Sigmod record* 28.2 (1999), pp. 49–60.

[165] Erich Schubert and Michael Gertz. “Improving the Cluster Structure Extracted from OPTICS Plots.” In: *LWDA.* 2018, pp. 318–329.

[166] Peter J Rousseeuw. “Silhouettes: a graphical aid to the interpretation and validation of cluster analysis”. In: *Journal of computational and applied mathematics* 20 (1987), pp. 53–65.

[167] Michael Hahsler, Matthew Piekenbrock, and Derek Doran. “dbscan: Fast density-based clustering with R”. In: *Journal of Statistical Software* 91.1 (2019), pp. 1–30.

[168] John H Cochrane. “Time series for macroeconomics and finance”. In: *Manuscript, University of Chicago* (2005), pp. 1–136.

[169] Keith W Hipel and A Ian McLeod. *Time series modelling of water resources and environmental systems.* Elsevier, 1994.

[170] Thanapant Raicharoen, Chidchanok Lursinsap, and Paron Sanguanbhokai. “Application of critical support vector machine to time series prediction”. In: *Proceedings of the 2003 International Symposium on Circuits and Systems, 2003. ISCAS’03.* Vol. 5. IEEE. 2003, pp. V–V.

[171] Ratnadip Adhikari and Ramesh K Agrawal. “An introductory study on time series modeling and forecasting”. In: *arXiv preprint arXiv:1302.6613* (2013).

[172] Neethu Mohan, KP Soman, and S Sachin Kumar. “A data-driven strategy for short-term electric load forecasting using dynamic mode decomposition model”. In: *Applied energy* 232 (2018), pp. 229–244.

[173] Chengjin Ye, Yi Ding, Peng Wang, and Zhenzhi Lin. “A data-driven bottom-up approach for spatial and temporal electric load forecasting”. In: *IEEE Transactions on Power Systems* 34.3 (2019), pp. 1966–1979.

[174] Abdolrahman Khoshrou and Eric J Pauwels. “Short-term scenario-based probabilistic load forecasting: A data-driven approach”. In: *Applied energy* 238 (2019), pp. 1258–1268.

[175] Yiyan Li, Dong Han, and Zheng Yan. “Long-term system load forecasting based on data-driven linear clustering method”. In: *Journal of Modern Power Systems and Clean Energy* 6.2 (2018), pp. 306–316.

[176] Joaquim Massana, Carles Pous, Llorenç Burgas, Joaquim Melendez, and Joan Colomer. “Identifying services for short-term load forecasting using data driven models in a Smart City platform”. In: *Sustainable cities and society* 28 (2017), pp. 108–117.

[177] Xin Liu, Zijun Zhang, and Zhe Song. “A comparative study of the data-driven day-ahead hourly provincial load forecasting methods: From classical data mining to deep learning”. In: *Renewable and Sustainable Energy Reviews* 119 (2020), p. 109632.

[178] Sneha Rai and Mala De. “Analysis of classical and machine learning based short-term and mid-term load forecasting for smart grid”. In: *International Journal of Sustainable Energy* 40.9 (2021), pp. 821–839.

[179] Sahbi Boubaker. “Identification of nonlinear Hammerstein system using mixed integer-real coded particle swarm optimization: application to the electric daily peak-load forecasting”. In: *Nonlinear Dynamics* 90.2 (2017), pp. 797–814.

[180] Yusen He, Jiahao Deng, and Huajin Li. “Short-term power load forecasting with deep belief network and copula models”. In: *2017 9th International conference on intelligent human-machine systems and cybernetics (IHMSC)*. Vol. 1. IEEE. 2017, pp. 191–194.

[181] Jia Ding, Maolin Wang, Zuowei Ping, Dongfei Fu, and Vassilios S Vassiliadis. “An integrated method based on relevance vector machine for short-term load forecasting”. In: *European Journal of Operational Research* 287.2 (2020), pp. 497–510.

[182] Ni Guo, Wei Chen, Manli Wang, Zijian Tian, and Haoyue Jin. “Appling an Improved Method Based on ARIMA Model to Predict the Short-Term Electricity Consumption Transmitted by the Internet of Things (IoT)”. In: *Wireless Communications and Mobile Computing* 2021 (2021).

[183] Subrina Noureen, Sharif Atique, Vishwajit Roy, and Stephen Bayne. “Analysis and application of seasonal ARIMA model in Energy Demand Forecasting: A case study of small scale agricultural load”. In: *2019 IEEE 62nd International Midwest Symposium on Circuits and Systems (MWSCAS)*. IEEE. 2019, pp. 521–524.

[184] Shuojiang Xu, Hing Kai Chan, and Tiantian Zhang. “Forecasting the demand of the aviation industry using hybrid time series SARIMA-SVR approach”. In: *Transportation Research Part E: Logistics and Transportation Review* 122 (2019), pp. 169–180.

[185] Jeong Hyun Lee, Jae Sung Kim, Young Ho Ahn, and Wan Sup Cho. “Daily forecasting of energy demand using SARIMA and LSTM method to support decision making on V2G”. In: *The Korean Data & Information Science Society* 30.4 (2019), pp. 779–795.

[186] Waseem Raza, Hina Nasir, and Nadeem Javaid. “Unification of RF energy harvesting schemes under mixed Rayleigh-Rician fading channels”. In: *AEU-International Journal of Electronics and Communications* 123 (2020), p. 153244.

[187] Paula Serras, Gabriel Ibarra-Berastegi, Jon Sáenz, and Alain Ulazia. “Combining random forests and physics-based models to forecast the electricity generated by ocean waves: A case study of the Mutriku wave farm”. In: *Ocean Engineering* 189 (2019), p. 106314.

[188] Pedro C Albuquerque, Daniel O Cajueiro, and Marina DC Rossi. “Machine learning models for forecasting power electricity consumption using a high dimensional dataset”. In: *Expert Systems with Applications* 187 (2022), p. 115917.

[189] Jui-Sheng Chou and Duc-Son Tran. “Forecasting energy consumption time series using machine learning techniques based on usage patterns of residential householders”. In: *Energy* 165 (2018), pp. 709–726.

[190] Diogo MF Izidio, Paulo SG de Mattos Neto, Luciano Barbosa, João FL de Oliveira, Manoel Henrique da Nóbrega Marinho, and Guilherme Ferretti Rissi. “Evolutionary hybrid system for energy consumption forecasting for smart meters”. In: *Energies* 14.7 (2021), p. 1794.

[191] Alejandro J del Real, Fernando Dorado, and Jaime Durán. “Energy demand forecasting using deep learning: applications for the French grid”. In: *Energies* 13.9 (2020), p. 2242.

[192] Nikolaos G Paterakis, Elena Mocanu, Madeleine Gibescu, Bart Stappers, and Walter van Alst. “Deep learning versus traditional machine learning methods for aggregated energy demand prediction”. In: *2017 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT-Europe)*. IEEE. 2017, pp. 1–6.

[193] Jihoon Moon, Sungwoo Park, Seungmin Rho, and Eenjun Hwang. “A comparative analysis of artificial neural network architectures for building energy consumption forecasting”. In: *International Journal of Distributed Sensor Networks* 15.9 (2019), p. 1550147719877616.

[194] Nivethitha Somu, Gauthama Raman MR, and Krithi Ramamritham. “A deep learning framework for building energy consumption forecast”. In: *Renewable and Sustainable Energy Reviews* 137 (2021), p. 110591.

[195] Michal Pavlicko, Mária Vojteková, and Ol’ga Blažeková. “Forecasting of Electrical Energy Consumption in Slovakia”. In: *Mathematics* 10.4 (2022), p. 577.

[196] Anam-Nawaz Khan, Naeem Iqbal, Atif Rizwan, Rashid Ahmad, and Do-Hyeun Kim. “An ensemble energy consumption forecasting model based on spatial-temporal clustering analysis in residential buildings”. In: *Energies* 14.11 (2021), p. 3020.

[197] Jihoon Moon, Sungwoo Park, Seungmin Rho, and Eenjun Hwang. “Robust building energy consumption forecasting using an online learning approach with R ranger”. In: *Journal of Building Engineering* 47 (2022), p. 103851.

[198] Joanna Henzel, Łukasz Wróbel, Marcin Fice, and Marek Sikora. “Energy Consumption Forecasting for the Digital-Twin Model of the Building”. In: *Energies* 15.12 (2022), p. 4318.

[199] Ning Jin, Fan Yang, Yuchang Mo, Yongkang Zeng, Xiaokang Zhou, Ke Yan, and Xiang Ma. “Highly accurate energy consumption forecasting model based on parallel LSTM neural networks”. In: *Advanced Engineering Informatics* 51 (2022), p. 101442.

[200] Rita Banik, Priyanath Das, Srimanta Ray, and Ankur Biswas. “Prediction of electrical energy consumption based on machine learning technique”. In: *Electrical Engineering* 103.2 (2021), pp. 909–920.

[201] Filipe Rodrigues, Carlos Cardeira, and João Mamiel Ferreira Calado. “The daily and hourly energy consumption and load forecasting using artificial neural network method: a case study using a set of 93 households in Portugal”. In: *Energy Procedia* 62 (2014), pp. 220–229.

[202] Muhammad Waseem Ahmad, Monjur Mourshed, and Yacine Rezgui. “Trees vs Neurons: Comparison between random forest and ANN for high-resolution prediction of building energy consumption”. In: *Energy and buildings* 147 (2017), pp. 77–89.

[203] Peter Lusis, Kaveh Rajab Khalilpour, Lachlan Andrew, and Ariel Liebman. “Short-term residential load forecasting: Impact of calendar effects and forecast granularity”. In: *Applied energy* 205 (2017), pp. 654–669.

[204] Hengfang Deng, David Fannon, and Matthew J Eckelman. “Predictive modeling for US commercial building energy use: A comparison of existing statistical and machine learning algorithms using CBECS microdata”. In: *Energy and Buildings* 163 (2018), pp. 34–43.

[205] Federico Divina, Miguel Garcia Torres, Francisco A Goméz Vela, and Jose Luis Vazquez Noguera. “A comparative study of time series forecasting methods for short term electric energy consumption prediction in smart buildings”. In: *Energies* 12.10 (2019), p. 1934.

[206] Zeyu Wang, Yueren Wang, Ruochen Zeng, Ravi S Srinivasan, and Sherry Ahrentzen. "Random Forest based hourly building energy prediction". In: *Energy and Buildings* 171 (2018), pp. 11–25.

[207] Tao Liu, Zehan Tan, Chengliang Xu, Huanxin Chen, and Zhengfei Li. "Study on deep reinforcement learning techniques for building energy consumption forecasting". In: *Energy and Buildings* 208 (2020), p. 109675.

[208] Duc-Hoc Tran, Duc-Long Luong, and Jui-Sheng Chou. "Nature-inspired metaheuristic ensemble model for forecasting energy consumption in residential buildings". In: *Energy* 191 (2020), p. 116552.

[209] Douglas C Montgomery, Elizabeth A Peck, and G Geoffrey Vining. *Introduction to linear regression analysis.* John Wiley & Sons, 2021.

[210] Graeme L Hickey, Evangelos Kontopantelis, Johanna JM Takkenberg, and Friedhelm Beyersdorf. "Statistical primer: checking model assumptions with regression diagnostics". In: *Interactive cardiovascular and thoracic surgery* 28.1 (2019), pp. 1–8.

[211] Dan Campbell and Sherlock Campbell. "Introduction to regression and data analysis". In: *StatLab Workshop Series.* 2008, pp. 1–14.

[212] Li-Juan Cao and Francis Eng Hock Tay. "Support vector machine with adaptive parameters in financial time series forecasting". In: *IEEE Transactions on neural networks* 14.6 (2003), pp. 1506–1518.

[213] Satish Kumar. *Neural networks: a classroom approach.* Tata McGraw-Hill Education, 2004.

[214] Vladimir Vapnik. "Statistical learning theory new york". In: *NY: Wiley* 1.2 (1998), p. 3.

[215] Johan AK Suykens and Joos Vandewalle. "Least squares support vector machine classifiers". In: *Neural processing letters* 9.3 (1999), pp. 293–300.

[216] Tahir Farooq, Aziz Guergachi, and Sridhar Krishnan. "Chaotic time series prediction using knowledge based Green's Kernel and least-squares support vector machines". In: *2007 IEEE International Conference on Systems, Man and Cybernetics.* IEEE. 2007, pp. 373–378.

[217] Yugang Fan, Ping Li, and Zhihuan Song. "Dynamic least squares support vector machine". In: *2006 6th World Congress on Intelligent Control and Automation.* Vol. 1. IEEE. 2006, pp. 4886–4889.

[218] Johan AK Suykens and Joos Vandewalle. “Recurrent least squares support vector machines”. In: *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications* 47.7 (2000), pp. 1109–1114.

[219] Tony Van Gestel, Johan AK Suykens, D-E Baestaens, Annemie Lambrechts, Gert Lanckriet, Bruno Vandaele, Bart De Moor, and Joos Vandewalle. “Financial time series prediction using least squares support vector machines within the evidence framework”. In: *IEEE Transactions on neural networks* 12.4 (2001), pp. 809–821.

[220] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. “Multilayer feedforward networks are universal approximators”. In: *Neural networks* 2.5 (1989), pp. 359–366.

[221] Roberto Perrelli. “Introduction to ARCH & GARCH models”. In: *University of illinois optional TA handout* (2001), pp. 1–7.

[222] Jerome H Friedman et al. “A recursive partitioning decision rule for nonparametric classification”. In: *IEEE Trans. Computers* 26.4 (1977), pp. 404–408.

[223] J. Ross Quinlan. “Induction of decision trees”. In: *Machine learning* 1.1 (1986), pp. 81–106.

[224] Charu C Aggarwal et al. *Data mining: the textbook.* Vol. 1. Springer, 2015.

[225] Leo Breiman. “Random forests”. In: *Machine learning* 45.1 (2001), pp. 5–32.

[226] Judea Pearl. “On the connection between the complexity and credibility of inferred models”. In: *International Journal of General System* 4.4 (1978), pp. 255–264.

[227] Leo Breiman and Ross Ihaka. *Nonlinear discriminant analysis via scaling and ACE.* Department of Statistics, University of California Davis One Shields Avenue . . ., 1984.

[228] J Ross Quinlan. *C4. 5: programs for machine learning.* Elsevier, 2014.

[229] Leo Breiman. “Bagging predictors”. In: *Machine learning* 24.2 (1996), pp. 123–140.

[230] Balaji Lakshminarayanan. “Decision trees and forests: a probabilistic perspective”. PhD thesis. UCL (University College London), 2016.

[231] David H Wolpert. “Stacked generalization”. In: *Neural networks* 5.2 (1992), pp. 241–259.

[232] Pierre Geurts, Damien Ernst, and Louis Wehenkel. “Extremely randomized trees”. In: *Machine learning* 63.1 (2006), pp. 3–42.

[233] Manuel Fernández-Delgado, Eva Cernadas, Senén Barro, and Dinani Amorim. “Do we need hundreds of classifiers to solve real world classification problems?” In: *The journal of machine learning research* 15.1 (2014), pp. 3133–3181.

[234] Evelyn Fix and Joseph Lawson Hodges. “Discriminatory analysis. Nonparametric discrimination: Consistency properties”. In: *International Statistical Review/Revue Internationale de Statistique* 57.3 (1989), pp. 238–247.

[235] Thomas Cover and Peter Hart. “Nearest neighbor pattern classification”. In: *IEEE transactions on information theory* 13.1 (1967), pp. 21–27.

[236] Phan Thanh Noi and Martin Kappas. “Comparison of random forest, k-nearest neighbor, and support vector machine classifiers for land cover classification using Sentinel-2 imagery”. In: *Sensors* 18.1 (2017), p. 18.

[237] K-R Muller, Charles W Anderson, and Gary E Birch. “Linear and nonlinear methods for brain-computer interfaces”. In: *IEEE transactions on neural systems and rehabilitation engineering* 11.2 (2003), pp. 165–169.

[238] Stephen D Bay. “Combining Nearest Neighbor Classifiers Through Multiple Feature Subsets.” In: *ICML*. Vol. 98. Citeseer. 1998, pp. 37–45.

[239] Wei Yuan, Juan Liu, and Huai-Bei Zhou. “An improved KNN method and its application to tumor diagnosis”. In: *Proceedings of 2004 International Conference on Machine Learning and Cybernetics (IEEE Cat. No. 04EX826)*. Vol. 5. IEEE. 2004, pp. 2836–2841.

[240] Keinosuke Fukunaga and Raymond R. Hayes. “Effects of sample size in classifier design”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 11.8 (1989), pp. 873–885.

[241] Juha Reunanen. “A Pitfall in Determining the Optimal Feature Subset Size.” In: *PRIS*. 2004, pp. 176–185.

[242] Chao Sima and Edward R Dougherty. “The peaking phenomenon in the presence of feature-selection”. In: *Pattern Recognition Letters* 29.11 (2008), pp. 1667–1674.

[243] Jinn-Min Yang, Pao-Ta Yu, and Bor-Chen Kuo. “A nonparametric feature extraction and its application to nearest neighbor classification for hyperspectral image data”. In: *IEEE Transactions on Geoscience and Remote Sensing* 48.3 (2009), pp. 1279–1293.

[244] Jian Yang, Lei Zhang, Jing-yu Yang, and David Zhang. “From classifiers to discriminators: A nearest neighbor rule induced discriminant analysis”. In: *Pattern recognition* 44.7 (2011), pp. 1387–1402.

[245] Mauricio Villegas and Roberto Paredes. “Dimensionality reduction by minimizing nearest-neighbor classification error”. In: *Pattern Recognition Letters* 32.4 (2011), pp. 633–639.

[246] Andreas Zell et al. “SNNS: Stuttgart Neural Network Simulator. User Manual, Version 4.2”. In: *Institute for Parallel and Distributed High Performance Systems, Technical Report* 6/95 (1995).

[247] Raúl Rojas. *Neural networks: a systematic introduction*. Springer Science & Business Media, 2013.

[248] Léon Bottou and Olivier Bousquet. “The tradeoffs of large scale learning”. In: *Advances in neural information processing systems* 20 (2007).

[249] Franco Cicirelli, Antonio Guerrieri, Carlo Mastroianni, Giandomenico Spezzano, and Andrea Vinci, eds. *The Internet of Things for Smart Urban Ecosystems*. Springer, 2019. ISBN: 978-3-319-96549-9. URL: `https://doi.org/10.1007/978-3-319-96550-5`.