OULUN YLIOPISTO
UNIVERSITY of OULU

Predicting parking space availability based on heterogeneous data using Machine Learning techniques

University of Oulu
Faculty of Information Technology and
Electrical Engineering / OASIS
Master's Thesis
Hassan Mehmood
27.03.2019

Abstract

These days, smart cities are focused on improving their services and bringing quality to everyday life, leveraging modern ICT technologies. For this reason, the data from connected IoT devices, environmental sensors, economic platforms, social networking sites, governance systems, and others can be gathered for achieving such goals. The rapid increase in the number of vehicles in major cities of the world has made mobility in urban areas difficult, due to traffic congestion and parking availability issues. Finding a suitable parking space is often influenced by various factors such as weather conditions, traffic flows, and geographical information (markets, hospitals, parks, and others). In this study, a predictive analysis has been performed to estimate the availability of parking spaces using heterogeneous data from Cork County, Ireland. However, accumulating, processing, and analysing the produced data from heterogeneous sources is itself a challenge, due to their diverse nature and different acquisition frequencies. Therefore, a data lake has been proposed in this study to collect, process, analyse, and visualize data from disparate sources. In addition, the proposed platform is used for predicting the available parking spaces using the collected data from heterogeneous sources. The study includes proposed design and implementation details of data lake as well as the developed parking space availability prediction model using machine learning techniques.

Keywords
Big data, Smart cities, machine learning, parking prediction

Supervisors
Piiastiina Tikka
Ekaterina Gilman

Foreword

In writing this thesis, I must acknowledge the support and bits of advice I received from my peers and my supervisors. Without constructive criticism and unabated support of D.Sc. (Tech.) Ekaterina Gilman and Dr Piiastiina Tikka, I would not have been able to complete this work in a timely manner. I would also like to thank Mr Lauri Lovén for his continuous guidance. I owe my current and future success to all the people who relentlessly supported my academic and professional development.
I would also like to mention my family members for continuously believing in my abilities, without their support I couldn't have achieved this milestone.

Hassan Mehmood

Oulu, March 27, 2019

Abbreviations

AI	Artificial Intelligence
API	Application Programming Interface
ARIMA	Autoregressive Integrated Moving Average
BRANN	Bayesian Regularization Artificial Neural Network
CUTLER	Coastal Urban developmenT through the Lenses of Resiliency
CPS	Cyber-Physical Systems
CDH	Cloudera Distribution including Hadoop
CoAP	Constrained Application Protocol
CSV	Comma-separated values
DSR	Design Science Research
EU	European Union
FNN	Feedforward Neural Network
GUI	Graphical User Interface
GeoJSON	Geographic JavaScript Object Notation
HDFS	Hadoop Distributed File System
HTML	Hypertext Markup Language
IS	Information Systems
ICT	Information and Communication Technology
IoT	Internet of Things
JSON	JavaScript Object Notation
LSTM	Long Short-term Memory Network
ML	Machine Learning
MAE	Mean Absolute Error
PCA	Principle Component Analysis

RMSE	Root Mean Square Error
RNN	Recurrent Neural Network
SVR	Support Vector Regression
SQL	Structured Query Language
TSV	Tab separated values
URL	Uniform Resource Locator
VCS	Version Control System
XML	Extensible Markup Language

Contents

Figures

Tables

1. Introduction

The exponential growth in the number of vehicles in large cities has given rise to high traffic flows and traffic congestion issues. The traffic congestion issues have several side effects including increased air pollution, noise pollution, as well as parking issues. Parking one's vehicle has always been problematic especially in cities with high population density. It has been estimated that about 30% of traffic is caused by vehicles searching for empty parking spaces (Ionita, Pomp, Cochez, Meisen, & Decker, 2018). Usually, drivers in densely populated areas spend around 3.5 to 14 minutes searching for empty parking spaces (Badii, Nesi, & Paoli, 2018). The spent time in searching for empty parking spaces depends on various factors: the purpose of travel, the proximity of parking garage, the difference in prices among parking spaces, and familiarity with the geographical landscape of the area (Badii et al., 2018). During the search for available parking slots, drivers usually circle around the locations they would like to park, which could lead to environmental and economic impacts (Ionita et al., 2018). The effort of finding available parking spaces can be reduced by notifying drivers about empty slots beforehand and can help in improving existing traffic management policies.

The search for parking has brought great demand for parking information and services. Many cities in the world have adopted the concept of "smart cities" for bringing reforms to unaddressed issues, which directly affect an individual's life (Lim, Kim, & Maglio, 2018). However, several smart cities have overcome this issue by providing smartphone-based applications, which provide real-time statistics on the availability of parking spaces in user-specified areas. Recent research has shown that with the support of big data, artificial intelligence (AI), and heterogeneous data sources like weather conditions, traffic data, and points of attraction can help in providing more accurate and reliable predictions (Badii et al., 2018). Big data and AI have collectively been used to provide various decision-making options in the everyday lives of people. AI-based intelligent systems are being developed for solving day-to-day problems, but designing an accurate intelligent system is itself a complicated task especially in real-time scenarios.

The data management platforms with attributes of predictive analysis require efficient techniques for handling large-scale data. In cases where data streams are heterogeneous in nature, more advanced techniques and procedures are required to collect, process, and transform the data into useful information for decision-making (Zheng, Rajasegarar, & Leckie, 2015). Current estimations show that approximately 2.5 quintillion bytes of heterogeneous data is produced globally and the percentile of unstructured data is about 90% (Sivarajah, Kamal, Irani, & Weerakkody, 2017). The availability of such an amount of data has given birth to various challenges for data management, data ingestion, data storage, and data analysis.

These days, the concept of "Data Lake" has gained prominence for accumulating data from disparate sources. Data lakes are considered useful for storing a large amount of data in raw format as a single physical data repository, providing different functionalities of data ingestion, data analysis, and data management (Madera & Laurent, 2016). The concept of data lakes emerged from the Big Data wave and Apache incubated Hadoop projects (Madera & Laurent, 2016). In this regard, data lakes are usually closely related to the Apache Hadoop ecosystem (O'Leary, 2014). Data lakes are mostly batch processing oriented due to their design for making data available first (Madera & Laurent, 2016). Therefore, in some cases, data lakes may provide limited support for real-time processing, which has been addressed in this study.

The main issue in processing real-time data stream is its high velocity. Processing time should, therefore, be considered when performing real-time predictive analysis, as in various applications, the value of data decreases over time (Maarala, Rautiainen, Salmi, Pirttikangas, & Riekki, 2015). For example, in urban traffic flows and congestion scenarios, where quick decision making is required for re-routing of traffic, not being able to process data the moment it arrives is considered a significant delay even if it is just by a single minute (Maarala et al., 2015; Shin, Jun, & Kim, 2018).

The purpose of this Master's thesis is to predict available parking spaces enabled by a data lake (data management platform) for data collection, processing, and analysis from heterogeneous sources. The data management platform will not only provide support for predicting available parking spaces but will also enable the user to perform different kinds of analysis for different other urban problems. One publication regarding the data management platform has been accepted for publication in a workshop – DASC 2019 being held within 35th IEEE International conference on data engineering, 2019 (ICDE-2019) under the title "Implementing data lake for heterogeneous data sources" (Mehmood et al., 2019).

Parking a vehicle is usually dependent on various factors, such as the motive for travelling, weather situation, and the flow of traffic, therefore proper assessment criteria are required for predicting available parking spaces. Predictive analysis can be performed using traffic density data, real-time parking data from multiple garages, weather condition and geographical information (e.g. a list of amenities from maps), to predict information pertaining to parking spaces. The adopted approach for implementing prediction model for available parking spaces also addresses the existing limitations in previous studies.

In this research, *supervised* machine learning (ML) techniques will be used on a feature set derived from accumulated data i.e. hourly traffic counters located near parking garages, real-time parking occupancy, hourly weather conditions in the area, geographical information like names of landmarks (e.g. markets, parks, hospitals etc.) to predict and analyse parking spaces availability. To address the objective of this thesis, the Design Science Research (DSR) method by Hevner, March, Park, and Ram (2004) will be used to answer the following research questions:

RQ1: In what ways can data analytics be used in the prediction of available parking spaces using machine learning techniques?

RQ2: What kind of technological solutions can be used for designing a data management platform enabling parking spaces prediction?

This thesis is being done as part of EU Horizon 2020 project CUTLER: Coastal Urban developmenT through the LEnses of Resiliency, which includes four pilot cities: Cork, Antalya, Thessaloniki, and Antwerp. The purpose of this project is to use big data analytics methods to make sense of data for developing urban policies, which could help cities in resolving various problems.

In this study, a general use case from Cork County has experimented. The city of Cork is the second largest city in the Republic of Ireland and has been going through infrastructural modifications, which has resulted in the elimination of many parking places (Roche, 2018). Hence, to address the parking problems of the city of Cork, a predictive analysis for parking availability will be performed using heterogeneous data sources.

The structure of this thesis has been organized in the following way: The research problem and chosen research methodology are explained in Chapter 2. Chapter 3 provides a theoretical background on the topic from identified scientific studies. The next chapter (Chapter 4) provides details on the elicited requirements for the artefact from the conducted literature review and identified problem of predicting parking available parking spaces. In Chapter 5, the architecture of the artefact is explained. The implementation of the artefact including data lake and predictive analysis is described in Chapter 6. In Chapter 7, details on the artefact evaluation are provided. The results of the study have been discussed along with its main limitations in Chapter 8. The future work and limitations of this thesis are explained in Chapter 9. Lastly, Chapter 10 concludes the whole study.

2. Research Problem & Methodology

This chapter consists of two main sections, in the first section research problem & questions are explained. The next section will describe the used research methodology, to design and implement a data management platform for heterogeneous data sources and to address the identified research problem. In this study, the DSR method by Hevner et al. (2004) will be used, which is the commonly adopted method in information systems disciplines these days.

2.1 Research Problem

The objective of this thesis is to design a data management platform for heterogeneous data sources and for predicting available parking spaces from the accumulated data. This can be further explained to answer the identified research questions for the said objective. As described in the introduction chapter, predicting available parking spaces could be an extensive task because of multiple parking garages available in the city, and varying traffic and weather conditions. The parking places are either located on streets or specified parking garages by city municipality or private vendors and these parking places can be sometimes monitored and un-monitored (Badii et al., 2018; Ionita et al., 2018).

The parking prediction is not only dependent on the occupancy of parking spaces but also on the motive of driver, weather conditions and traffic density (J. Li, Li, & Zhang, 2018). Especially, during the peak hours and special occasions finding parking becomes difficult due to congestion on the main routes of the city (J. Li et al., 2018). Besides, there are other factors which influence the parking availability like parking charges, opening hours and accessibility to the parking place. The following research questions are formed to find the best answers to the research problem.

RQ 1: In what ways can big data analytics be used in predicting available parking spaces using machine learning techniques?

The first research question is aimed at identifying the available machine learning methods and techniques for performing predictive analysis. The answer to this research question will provide the needed knowledge to understand different machine learning mechanisms that can be used for performing predictive analysis using heterogeneous data sources. Also, what kind of data integration and transformations methods can be used for building an accurate learning model for predicting parking spaces from disparate data sources. As well as how data sources (traffic counts, weather conditions, parking count, and geographical information) in the scope are correlated with each other. Furthermore, the analysis will be made on the basis of what are the limitations of the chosen methods, and how can they be improved.

RQ2: What kind of technological solutions can be used for designing a data management platform enabling parking spaces prediction?

The second research question comes from RQ1, as predicting parking spaces relies on various data sources, and requires proper data management platform. Therefore, this research question is focused on providing an in-depth understanding of different big data technology solutions, which can be used for data collection, data processing, data analysis, and creating visually-aided results. Furthermore, how can the chosen technologies be used for collecting and processing data with varying frequencies i.e. per minute, hourly, daily, and so on. The answer to this question will provide the needed knowledge base for discussion and required necessary steps for building a data management platform infrastructure and what techniques and tools can be used for accumulating heterogeneous data sources to the storage repository. Secondly, it will also help to understand and discuss the various issues of data management, such as data monitoring, data integration, and data processing.

2.2 Research Method

The DSR method by Hevner et al. (2004) these days is a commonly used method in the field of information systems because of its support for developing new and innovative solutions. According to Hevner et al. (2004), the focus of the DSR framework is to allow new technology-based designs in the field of information system driven by innovative ideas. The main idea behind DSR method is to first understand the problem, gather relevant knowledge, build the artefact using different techniques and methods, and evaluate it by strictly following the guidelines presented by Hevner et al. (2004). The artefact in the DSR method can be either a construct, model, or an instantiation depending upon the problem space (Hevner et al., 2004).

2.2.1 Design Science Research Framework

There are various design science frameworks available, but they differ in their own way i.e. during the research process what should be in focus and how the whole process should be executed depending upon the domain. The main research question for this study is to predict parking spaces, and develop a data management platform to collect, process and provide support for predictive analysis, as described in Chapter 1. A new artefact will be built and evaluated for the identified problem space using DSR method by Hevner et al. (2004), the complete process of DSR method is presented in Figure 1. In addition, the seven design guidelines by Hevner et al. (2004) will also be followed for constructing the artefact. The detailed reflection and assessment of DSR method in context to the identified research problem are explained in Chapter 8.

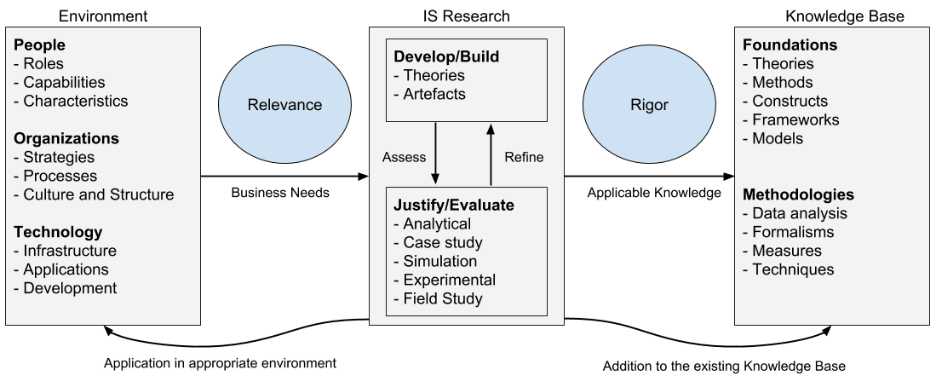


Figure 1. Information Systems Research Framework (Hevner et al., 2004)

The framework presented by Hevner et al. (2004) in Figure 1 is influenced by three main components i.e. environment, IS research and Knowledge base. The environment describes the settings of the research to be conducted, people, organization and technologies to be used. The IS research portion describes the development and evaluation stages of the research process. Whereas the knowledge base deals with the used theoretical foundations, methodologies and techniques. The environment, IS research and knowledge base for this thesis are described in detail below:

Environment

According to Hevner et al. (2004), as mentioned in Figure 1, the environment in DSR determines the problem, which contains the object of interest. The environment is mainly comprised of people, organization, and technology which are going to benefit from the conducted research. It holds all the relevant items for the study, mainly the objective, opportunities, problems, required tasks which are identified by assessing the context, culture, and business processes of the organization. The relevance, which exists between environments and IS research provides the requirements for performing the research. In the context of this thesis, it provides requirements for predicting available parking spaces from diverse sources, supported by the artefact (data management platform).

People: The data management platform will be designed for accumulating, processing, analysing heterogeneous data sources, also to perform predictive analysis on different types of data for this thesis: parking data, traffic flow counters, weather conditions, and geographical information. The geographical information contains data about the main places of the city, such as markets, hospitals, recreational parks, and educational institutes etc. The artefact being developed for this thesis can be used upon request by anyone who is interested especially, people from the University of Oulu, interested researchers, and partners from CUTLER project.

Organization: The artefact will be developed and evaluated at the University of Oulu. As this thesis is a contribution to a bigger project called CUTLER project. Therefore, partners from CUTLER project can use the created dataset and algorithms for predicting parking spaces availability. Besides, one of the partners from CUTLER (Thessaloniki) is also looking for making reforms towards parking infrastructures, the followed approach can be subsequently used by them as well based on their needs.

Technology: The artefact in this research study is a data lake developed using suitable big data technology solutions. The data management platform will be built using the data pipeline approach, a sandbox-cluster will be installed and configured using Linux OS. To collect and analyse data from disparate sources, Cloudera Distribution including Hadoop (CDH) will be distributed over the cluster containing parcels like Apache Spark, Zookeeper, Hue, Apache Flume, and others. The configuration files and the implemented scripts for data collection, transformation, and analysis will be maintained through Jupyter Notebook and GitHub as a version control system (VCS).

IS Research

The design science research in IS mainly includes two phases: the development phase and validation phase (Hevner et al., 2004). The development phase in the present research will involve the implementation of a predictive model for available parking spaces through machine learning techniques. As well as, the implementation of a data management platform based on the identified needs for predictive analysis to collect data through custom wrote scripts and big data tools, data processing, and data analysis from heterogeneous data sources.

The constructed platform is designed to support the process of data collection and ingestion from data sources which are static and dynamic in nature – data sources with different acquisition frequencies i.e. every minute, hourly or daily. After the data collection stage, gathered data will be injected into a central repository – a data lake for analysis. The analysed information will then be evaluated based on the identified assessment criteria e.g. accuracy of the prediction, does the used machine learning model fulfils the needs, and does parking data correlated with other data sources based on the assumption.

Knowledge Base

The knowledge base is a key component in DSR, knowledge base allows the researcher to build intended artefact for research problem by using already built knowledge in variously related research. The knowledge base in this study is the existing literature (related work) on the topics of big data analytics pipeline, data collection from heterogeneous data sources, smart city platforms, and machine learning techniques. The knowledge base of this study is elaborated in Chapter 3.

2.3 Development Cycles

The core of this IS research is to develop an artefact for data collection, data ingestion, data analysis, and data visualization and to predict parking availability based on heterogeneous data sources. The development process of the artefact is iterative in nature, where implementation is done in multiple cycles. The artefact is considered as completed after all requirements are met, which are described in Chapter 4. Each cycle in the development process has certain objectives and the evaluation of artefact will be done accordingly. As the system is developed as a prediction framework, all chosen methods and techniques will be tested through specified assessment criteria. In this section, C represents the cycle, each cycle is defined using prefix as C and suffix a numerical value followed by cycle description.

C1: Identifying Data sources

The objective of the first cycle is to identify data sources for predictive analysis for available parking spaces. The data sources for this study are gathered from Cork County. The selected data sources should be able to support the prediction process in the developed platform. The created data catalogue should reflect available data types, access points and access methods. This cycle will be evaluated by analysing whether the data is accessible and there are no restrictions in terms of licensing and privacy.

C2: Designing the infrastructure

The requirement of this cycle is to design and implement the infrastructure for the artefact. The chosen architecture must allow all the activities to be executed for accumulating, processing and analysing the data in the context of the identified research problem. This cycle will be evaluated by reviewing the chosen architecture, packages and operating system i.e. whether selected components provide support for performing predictive analysis. This section is also evaluated from prior research in Chapter 3.

C3: Data collection

The third cycle is about implementing data collection pipeline, which will stream data from identified sources in C1 based on their acquisition frequencies. The collection process is scheduled for data ingestion to a central repository. In this context, it will be a Hadoop Distributed File System (HDFS). This cycle will be evaluated by keeping track of inserted data into HDFS, relevant log files will be reviewed for failures and missing data.

C4: Pre-Processing of data

The aim of this cycle is to pre-process collected data in HDFS and create an integrated data set containing features like time, parking occupancy details, weather conditions, traffic density counters, and geographical information. The pre-processing will include cleaning of data (removing special characters, time conversion, and harmonizing the accumulated data). This cycle will be validated through by closely analysing whether data is understandable and meaningful for analysis.

C5: Analysis & Prediction

The objective of this cycle is to analyse the gathered data as part of predictive analysis using machine learning models. The cycle will be evaluated by measuring the performance and accuracy of chosen machine learning models, and further, it will be compared with models used in existing literature.

C6: Testing the data management platform

The testing process will be executed throughout all the cycles, but in this cycle, specified testing will be performed based on assessment criteria. The assessment criteria will mainly constitute the performance of infrastructure, the performance of data analytics pipeline and measuring the accuracy of the final machine learning model.

2.4 Validation & Evaluation Method

This study is developed using DSR method by Hevner et al. (2004). The evaluation of the artefact being developed in this study will be evaluated using evaluation activities by Sonnenberg and vom Brocke (2012). In IS research the outcome of DSR method is knowledge generated in the form of an artefact and proposition. Usually, DSR is mainly comprised of two activities i.e. build and evaluate. However, validating a design process and built artefact has always found to be crucial (Sonnenberg & vom Brocke, 2012). According to Sonnenberg and vom Brocke (2012), the generated knowledge from DSR method can have a truth value, if its guidelines are followed properly along with a continuous assessment of the artefact being built.

The developed artefact can be validated by formulating a suitable criterion, if the artefact is going to be used in practice, the artefact can be evaluated against the criteria: a) its importance, b) suitability and c) its accessibility (Sonnenberg & vom Brocke, 2012). However, there are other criteria, which can be used as well, such as completeness, efficiency, elegance, robustness, consistency and others, depending upon the type of artefact. Moreover, the mentioned criteria can be used while performing evaluations. In this regard, evaluation patterns (design-evaluate-construct-evaluate) by Sonnenberg and vom Brocke (2012) can be used for validating the artefact in an incremental manner.

The evaluation pattern has been distributed into two main stages i.e. *ex-ante evaluation* and *ex-post evaluation*, where within each stage, there are sub stages consisting of DSR activities, such as identification of problem, design, develop and use. The ex-ante evaluation is about validating the design of the artefact and ex-post evaluations are about validating the designed artefact instances and their use. In this way, each activity can be evaluated using the suggested criteria by Sonnenberg and vom Brocke (2012). As can be seen from Figure 2, the proposed process by Sonnenberg and vom Brocke (2012), suggests that each activity is to be evaluated upon its completion and the same cycle can be repeated. The artefact in this thesis was developed iteratively, as described in Section 2.3. Thus, the evaluation activities proposed by Sonnenberg and vom Brocke (2012) are well-suited for this thesis. The flow of evaluation patterns is shown in Figure 2.

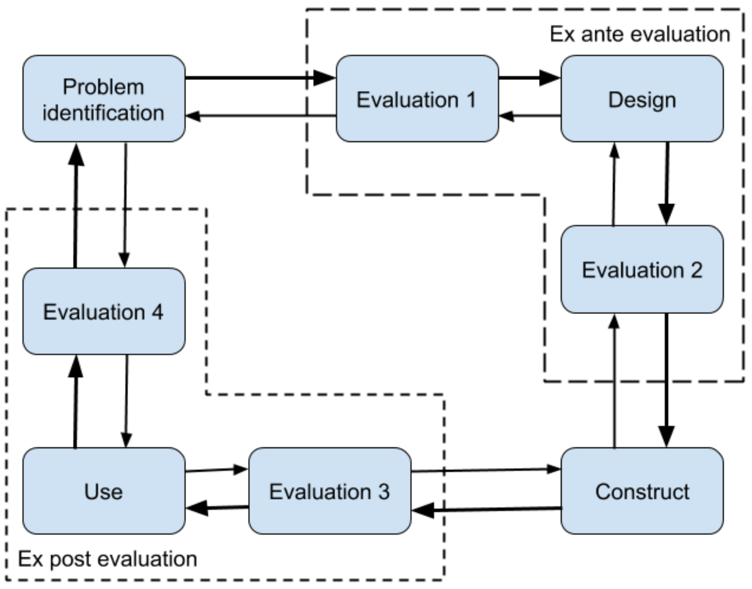


Figure 2. Evaluation activities in DSR, redrawn from (Sonnenberg & vom Brocke, 2012)

3. Prior Research

The present chapter draws together and describes existing knowledge as regards to existing the research topic of this thesis. The heterogeneity of data in context with big data applications is described with the commonly faced challenges in collecting, storing and creating conformity for such data. The chapter also explains the ways to implement the architecture as complete data pipeline for heterogeneous data sources. The available studies on similar subject i.e. heterogeneous data sources, big data, smart city platforms, data lakes, big data technologies, and machine learning techniques for building the aimed artefact are reviewed to create a sound knowledge base for this study. Furthermore, different studies on parking prediction frameworks are reviewed to comprehend the research problem, identify the different approaches and machine learning techniques which can be used to address the said problem, as described in Chapter 2.

3.1 Big Data and Smart City Platforms

The advances in technology and digitalization trends have aided in producing great volumes of data. Within this flow, the demand for data analysis is becoming more and more common, people who are interested in analysis data coming from heterogeneous sources are hearing the phrase *Big Data* more frequently these days (Marron, 2017).

The term *Big Data* is described in different ways but the commonly accepted definition is defined by five Vs: Volume, Value, Variety, Veracity and Velocity (Iqbal, Doctor, More, Mahmud, & Yousuf, 2017). According to Iqbal et al. (2017), the volume is described by the amount of data created and stored every second globally. The value is described as the provision of insights from the created data, for example, finding and analysing anomalies from such data (Iqbal et al., 2017). The variety depicts the types of data which are being created every second globally and veracity is referred to trustworthiness and the amount of noise does the data contains (Iqbal et al., 2017). The fifth V (velocity) of big data is defined by the frequency of data creation i.e. the speed of data creation in seconds, minutes or hours (Iqbal et al., 2017).

The demand for analysing the produced heterogeneous data from sources like social media, sensors, logs, stock exchanges etc. is now becoming more widespread for decision-making by researchers, industrialists and especially organizations who are involved in business intelligence systems (Ishizuka, Chen, & Paik, 2016). The types of data produced from heterogeneous sources can be textual, structured and unstructured formats, media-based data containing videos, images, sensor readings for traffic flow and weather conditions etc., communication data and data created from the internet of things (IoT) (Maarala et al., 2015; Sivarajah et al., 2017). The increasing volumes of generated data require specified workflow-based systems for processing heterogeneous data for minimizing failure risks, developing new applications and decision making (Ishizuka et al., 2016).

3.1.1 Smart City Platforms

The concept of smart cities first appeared in the 1990s (Lim et al., 2018). Since then the concept of smart has been attracting researchers, policy makers and urban planners. The researchers have been involved in investigating the information and communication technology (ICT) based approaches for constructing smart cities e.g. smart city application development such as IoT interfaces, sensors, smart cards, security management platforms, and big data analytics (Lim et al., 2018). Despite the progress, which has been made in smart cities, evaluating its applications in a real-world environment is still difficult (Santana, Chaves, Gerosa, Kon, & Milojicic, 2017).

In the past decade, an excessive amount of test-beds have been deployed, which are still experimental due to various challenges such as scalability, heterogeneity, and less involvement of end user in experimenting IoT applications (Santana et al., 2017). However, along with its challenges the number functional requirements have also amplified. These days, one can find the concept of smart cities aiming to improve the urban needs such as in transportation, hospitals, health care, energy and waste management, governance, safe cities, and others (Brohi, Bamiah, & Brohi, 2018). With its progression, various IoT and sensor-based applications have been deployed for bringing reforms in living standards and governance systems. The data produced from the employed smart systems and application is diverse in nature. Alongside its benefits, there are some challenges such as maintaining the quality of data, dealing with security and privacy matters, data integration, mapping user's needs, and developing feasible smart systems.

With the emerging trend of smart cities, one can find examples of technical maturity among the technologies and tools which are being utilized for designing smart cities. According to a study conducted by Santana et al. (2017), there are four main technologies which are leading smart city projects i.e. IoT, Big data, cloud computing, and cyber-physical systems (CPS). Most of the smart city platforms are not designed with a focus on one technology rather a combination of these technologies is used. A three-tiered IoT-based architecture was designed for SmartSantader to sense air quality, noise, and energy emission (Sanchez et al., 2014). The applications of the project were not limited to environment sensing but also to parking management and providing support for irrigation in parks and gardens such as water requirements and weather conditions (Sanchez et al., 2014).

Another smart city project called Padova was developed using IoT devices to address environmental and transportation issues (traffic flows, parking management, and others) (Cenedese, Zanella, Vangelista, & Zorzi, 2014). The architecture of the developed platform consisted of multiple IoT sensors, where constrained application protocol (CoAP) was used for communication purposes (Cenedese et al., 2014). Despite, the advancements in this field, making data reusable is still difficult due to its diverse nature. The data from these smart city projects is produced from disparate sources, which makes data integration, and standardization a major challenge towards big data analytics (Lim et al., 2018). However, some of these challenges have been addressed with the support of cloud computing and big data technology solutions. (Cheng, Longo, Cirillo, Bauer, & Kovacs, 2015).

3.2 Data Lakes

The statistics presented in research by Sivarajah et al. (2017) depicts that by the year 2020 the size of data generated and consumed globally will reach 40 zettabytes (40 trillion gigabytes). According to Zhao, Garg, Queiroz and Buyya (2017), about 90% of today's global data was created only in the last two years. The observations show that the created data is not only large as well as its velocity of creation has also become enormously fast (Zhao et al., 2017). Such volume and velocity of data pose the challenges of collecting, storing and processing such large-scale data.

Nowadays, the concept of data lake is becoming popular due to its support for accumulating large-scale data as a single big data repository (Fang, 2015). Additionally, data lakes are aimed to provide a new alternate environment for decision support systems. The data lakes are more agile in nature than traditional decision support systems (e.g. data warehouses) (Madera & Laurent, 2016).

Data lakes in comparison with data warehouses are storage repositories containing data from disparate sources in raw formats. The data can be either in structured, semi-structured and unstructured formats (Madera & Laurent, 2016). Data in data lakes is stored in a single physical repository, built on Hadoop technology. Unlike data warehouses, no change is performed while injecting data to the repository (Madera & Laurent, 2016). On the other hand, data warehouses are developed with extract, transfer, and load (ETL) approach, that can be further be captured for analysis from alternative databases (often require costly add-ons) (O'Leary, 2014). According to O'Leary (2014), data warehouses are generally focused on a particular data source with the objective to facilitate a particular community.

Unlike data warehouses, data lakes are not focused on facilitating specific community, rather they are an open domain, where data from heterogeneous sources is stored and can be queried, processed, and analysed using the same data repository (O'Leary, 2014). Data lakes provide various capabilities (Fang, 2015; O'Leary, 2014), some of them are as follows:

- No additional add-ons are needed to integrate with external databases, access is provided through custom written programs by data lake developers
- Data can be ingested in both batch and real-time
- Data lakes are highly scalable in cases, where the growth of data is continuous
- Data lakes are cost-effective for storing large-scale data
- Data stored in data lakes can be reused and re-purposed
- Multiple users can access the storage repository for processing and analysis

Along with many advantages of data lakes, there are some limitations as well. Despite, the availability of different frameworks to address general requirements of a data lake, implementing a data lake still requires additional technical effort (Hai, Geisler, & Quix, 2016). Moreover, integrating disparate sources demands advanced metadata management approaches. The unavailability of metadata can make the process of querying and integration challenging (Hai et al., 2016). To address such challenges, a data lake solution

(Constance) by Hai et al. (2016) was developed for metadata management of heterogeneous data, to make the integration process easier within data lakes. Furthermore, AI-based schema evolution approaches can also be utilized for harmonizing the data from disparate sources (O'Leary, 2014). Likewise, a system named CLAMS was developed to meet the challenges of data management in data lakes e.g. maintaining the integrity of data (Farid, Roatis, Ilyas, Hoffmann, & Chu, 2016).

3.2.1 Data ingestion

The data ingestion is described as a process of inserting data to HDFS from different external sources in the form of real-time data streams (Kamala & Marygladence, 2015). The external sources from where the data is fetched can be web services, log files and a relational database. There are various big data solutions available for ingesting data from sources to HDFS. One of the solutions is the Apache Foundation incubated project *Apache Sqoop.* Apache Sqoop mainly used for ingesting data to HDFS from relational databases, it consists of standard drivers and connectors enabling ingestion of data in a de-normalized format with the reduced overall load.

The other solution is *Apache Flume,* which is one of the reliable HDFS ingestion solutions for writing data in the form of logs to HDFS. Its distributed architecture allows maintaining the load on all nodes unlike traditional architectures (Maarala et al., 2015). Apache Flume is designed to run a separate Java Virtual Machine (JVM). Its architecture consists of a source, sink and channel allowing reliable and fault tolerant data ingestion to HDFS (Liu et al., 2014; Maarala et al., 2015). The HDFS has been designed for storing large files as raw data, that can be incomplete, duplicate, in different formats (Kamala & Marygladence, 2015). The source in Flume fetches the data from end source and passes it to the channel, where a channel could be of two types, memory and file. Failures are possible if the channel type is a memory, whereas the file channel allows recovering the lost data exhibiting the quality of fault-tolerance. After the data is passed to channel it is transferred to sink (destination) for writing into HDFS (Kamala & Marygladence, 2015). The architecture of Apache Flume is mentioned in Figure 2, where the stream is coming from web-server, which is then passed to the source, channel and sink respectively for writing data to HDFS.

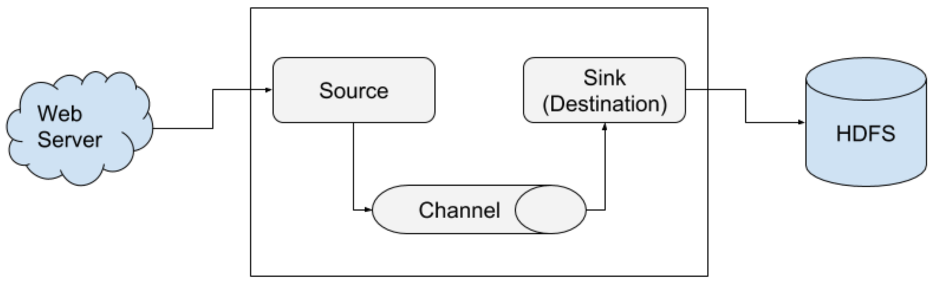


Figure 3. The architecture of Apache Flume, redrawn from (Kamala & Marygladence, 2015)

3.2.2 Data Storage

In big data context, data can be stored in HDFS. It allows storing files which are very large in size, providing sequential access for processing and analysis (Katal, Wazid, & Goudar, 2013; K. Li, Deolalikar, & Pradhan, 2015). It runs on commodity hardware, specially designed for clustered environments. The default block size of HDFS is 128 MB, enabling the reduce the number of disks seeks (Katal et al., 2013). However, the block size can be modified based on the requirement of the user. The main components of HDFS are block, name node, and data nodes. The files are partitioned in blocks, which are then stored on data nodes. The namespace, location of blocks, and file trees are managed by name node (Katal et al., 2013).

3.2.3 Data Processing and Analysis

The results from a survey conducted in Europe by Liu et al. (2014) showed, out of 271 information technology organizations, 70% of organizations emphasized the need for real-time processing. The available big data solutions were designed with a focus to achieve high throughput but processing data sources with low latency is still a challenge. The results from the survey also showed that only 9% of organizations were successful in resolving low latency issues (Liu et al., 2014). The traditional methods and solutions are suitable for processing raw in batches. However, recent advancements in the field of big data have enabled to process and analyse real-time data for decision making.

Analysing large-scale heterogeneous data requires integrating data in a meaningful way for analysis. The data integration is itself a crucial task, which requires implementing different approaches, restructuring, cleaning, and synchronizing received data streams based on their frequencies beforehand (Li & Wang, 2007; Price & Flach, 2013). Many authors have proposed the workflow-based architecture for dealing heterogeneous data, where challenges of speed, parallelism, interoperability, storage costs and operational costs are common especially when dealing with varying latencies of data (Liu, Iftikhar, & Xie, 2014; Maarala et al., 2015; Price & Flach, 2013).

The solutions like *Apache Spark* and *Apache Storm* are well-suited for analysing real-time data streams (Liu et al., 2014; Salloum et al., 2016). The following sections describe the available big data technologies for analysing heterogeneous data sources.

Apache Storm is developed for processing real-time data streams especially which are updated frequently (Maarala et al., 2015). The abstraction of Apache Storm provides user to process real-time data streams with low latency in an efficient manner. It follows a data flow like architecture, where data streamed from networks is consumed by the stream as un-sequenced tuples (Liu et al., 2014).

The other frequently used big data solution is Apache Spark. It gained much popularity in the last few years with the demand for processing real-time data streams (Salloum et al., 2016; THEIN, 2014). The Spark project was first started by Matei Zaharia at Berkeley's AMPLAB in 2009, the further improvements in Spark were done right after it came under incubation of Apache Foundation. The major improvements were done in upper libraries and core of spark (Salloum et al., 2016). It has been found to be optimal for processing real-time data streams from heterogeneous sources (structured or un-structured) (Ranjan, 2014). The processing model of Apache Spark is called as Resilient Distributed Dataset (RDD), which well-suited for processing heterogeneous data sources without storing them for quick decision-making (Liu et al., 2014; Salloum et al., 2016).

Apache Spark consists of two main components i.e. main upper libraries and Spark core. The abstraction levels allow developers to take advantage of its specific libraries for producing different kinds of analysis. It includes MLlib, spark ML and GraphX libraries which are used for producing different results based on machine learning techniques

(Ranjan, 2014; Salloum et al., 2016). The Spark SQL allows developers to analyse and process data just like normal SQL without the need for writing complex programs. In case of complex computing problems, it multiplies the memory of deployed cluster to minimize reliance of primary distributed files system, resulting in considerable performance gains comparatively to Map Reduce (Salloum et al., 2016). It is still evolving daily along with support for various general programming language support.

The master-worker based, distributed architecture of the Apache Spark allows dividing available tasks over the worker nodes, where each worker node is managed by a cluster manager – master node. It has support for diverse cluster managers such as Hadoop Yarn, Apache Mesos, Amazon EC2 and its default cluster manager, allows it to run in a distributed and standalone environment (Salloum et al., 2016). The cluster manager is responsible for handling and distributing the resources between multiple Spark applications, running over different worker nodes. Besides it provides huge support for accessing data from various kinds of sources like HDFS, Cassandra, HBase, Hive and others (Salloum et al., 2016). The architecture of Apache Spark is described in Figure 3.

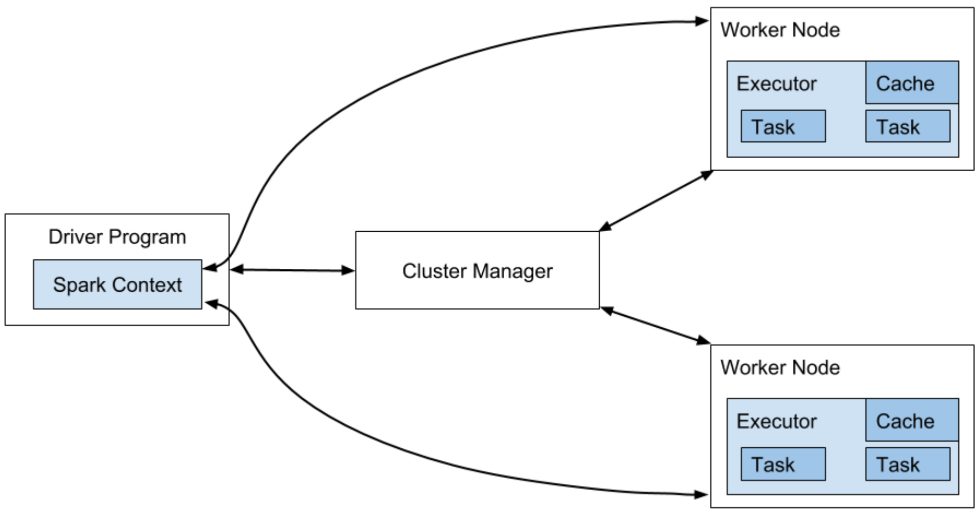


Figure 4. The architecture of Apache Spark, redrawn from (Salloum et al., 2016)

3.2.4 Data Visualization

Every day new data sources are created, and these data sources have become an important part of human life. To understand and create significant results from these data sources, researchers and practitioners are using data analytics techniques to solve day to day problems (Hajirahimova & Ismayilova, 2018). Generating results from large-scale data not only require advanced processing techniques and methods but also graphical and image-based representation approaches to understand the data and streamline the process of decision-making (Yur 'evich Gorodov & Vasil 'evich Gubarev, 2013). Using traditional data representation approach in the context of big data is found to be less effective in practice, causing issues of visual noise, extensive image perception, and loss of information (Yur 'evich Gorodov & Vasil 'evich Gubarev, 2013).

Visual noise in large-scale data representation is defined as upon visualization due to the large size of data, the created visual image may not be effective due to a large number of objects in the data. Hence, the user may only see a large cluster of points overlapped with each other (Yur 'evich Gorodov & Vasil 'evich Gubarev, 2013). Visual noise can be reduced by distributing data over multiple visuals. It will, however, organize the data objects visually but the user may lose track and may not be able to extract useful information (Yur ’evich Gorodov & Vasil ’evich Gubarev, 2013). These issues can be resolved by using data reduction techniques such as performing sampling and filtering, followed by standard visualization techniques. However, this will require high computational power depending upon the size of data and it can even overlap important parameters from data in some cases (Z. Liu, Jiang, & Heer, 2013; Yur ’evich Gorodov & Vasil ’evich Gubarev, 2013).

Parallel data querying and parallelism-based optimization approaches can help in reducing the computational loads especially when the data is large-scale (Philip Chen & Zhang, 2014; Zhang et al., 2016). Additionally, using parallelism can allow the user to provide results in a more optimized manner in cases, where sequential and real-time analysis is required. The current developments in the field of big data have produced various parallel architecture-based visualization tools, one of them is Hue (Sirisha & Kiran, 2018). Hue is an open-source web interface, which allows the user to interact with different tools (e.g. Apache Spark, Hive, Pig, HBase and others) within the Hadoop ecosystem in a more user-friendly manner. It also provides support for developing different kinds of visualization from data analysis. Additionally, for user non-technical users, there different built-in data analysis plugins, which can be used for analysing large-scale without writing complex programs (Sirisha & Kiran, 2018). Besides, standard data visualization libraries like Matplotlib (Hunter, 2007), if configured over distributed architectures, can also reduce the load time and help data scientists to produce user-friendly results. Matplotlib can be configured with IPython, which allows the user to run it in pylab mode that detects the setting of Matplotlib to draw interactive visuals and figures (Hunter, 2007).

3.3 Machine Learning

Machine learning is considered as a branch of artificial intelligence that is used for synthesizing and elucidating the existing relationships between multiple data points using different algorithms systematically (Awad & Khanna, 2015). For example, ML methods can be trained over a dataset for solving classification problems like identifying the sentiment of the given text document (whether it has positive, negative or neutral sentiment value). These days, applications of ML are becoming more widespread than ever, one can find its application in search engines, digital marketing, stock market prediction, forecasting weather conditions, big data and many others (Awad & Khanna, 2015). The purpose of ML is to predict events or scenarios which are yet to occur and not known to the computer. Awad and Khanna (2015) have to define ML as an area which makes a computer capable of learning without being explicitly programmed. However, the learning process is decisive, where the problem is generalized based on historical experience. The historical experience is provided to the computer in the form of training data set, which helps in generating accurate results based on previously available occurrences (Awad & Khanna, 2015).

Nowadays, many big data applications are leveraging different ML methods to provide effecient results. Though, data in big data context is of huge volume, diverse in nature, and the velocity of data flow has made it difficult to analyse and produce results in real-time (Awad & Khanna, 2015). Not only this, but the attribute of heterogeneity in big data has also made it difficult to create conformity between of data, and make it interoperable (Kadadi, Agrawal, Nyamful, & Atiq, 2014). In this regard, to analyse diverse kinds of data, data integration is necessary. This may involve a restructuring of data, using semantic matching to harmonize, and data model development (Kadadi et al., 2014). However, with all the present challenges, big data streams still provide numerous opportunities to stimulate the learning methods, which can help in solving various problems. According to Awad and Khanna (2015), a complete process of ML can involve these stages: a) Data collection, b) Data pre-processing, c) Data transformation, d) Training of algorithm, e) Testing the algorithm, and g) Execution.

There are various types of ML methods, such as supervised learning, unsupervised learning, semi-supervised learning, and others (Awad & Khanna, 2015). These methods can be used in different cases depending upon the requirements. However, the results produced from these methods may not relate to the way a human learns but one can get clear insights about data (Awad & Khanna, 2015). The types of ML methods are described in the following sections.

3.3.1 Supervised Learning

Supervised learning is a learning procedure, which is commonly used for solving classification and regression problems. Supervised learning works by inferring the existing relationship between the observational data – input data and the dependent variable for which prediction is to be performed (Awad & Khanna, 2015). The learning of ML models in supervised learning is performed by labelled training data (in classification problems) and known dependent variable (in regression problems) to synthesize the main model function, that attempts to understand and create a sense of the underlying relationship between input and output (Awad & Khanna, 2015). The main purpose of using machine learning algorithms is to minimize the errors for a given subject – training data set. In cases, where the quality of data is poor, the model may face overfitting, which means that the generated output is poorly generalized and erroneous (Awad & Khanna, 2015).

The classification problems by nature discriminate the provided examples for attaining reliable prediction i.e. once the model is finalized that fits the provided data, the future predictions can be compared with provided past data, if they are similar then accurate prediction can be made for other new instances (García, Luengo, & Herrera, 2015). In regression problems, when the target variable is infinite e.g. to predict a certain number between x and y interval, supervised-learning model learns output from input variable after fitting with data. In comparison with classification problems, regression problems are more difficult and require high computational resources (García et al., 2015). The general idea of a supervised learning mechanism is described in Figure 5.

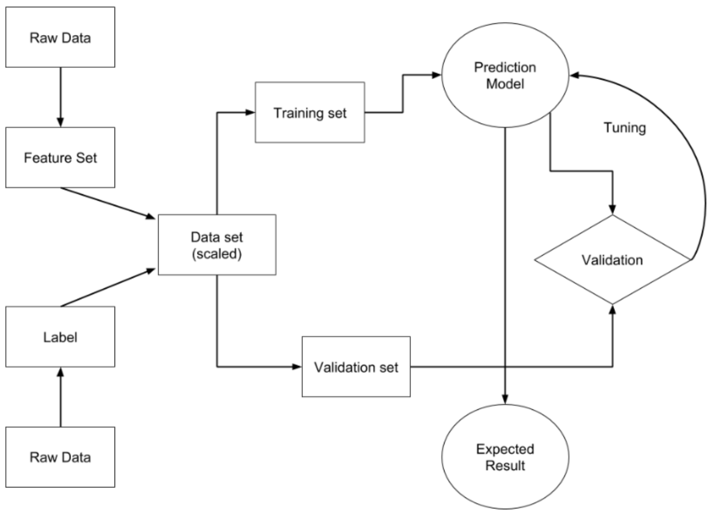


Figure 5. High Level Flow of supervised learning, redrawn from (Awad & Khanna, 2015)

Some types of the supervised learning which deals with classification and regression problems are, Naïve Bayes, Support Vector Machines, Decision Trees, Random Forest, Gradient-boosted trees, Adaboost, different neural networks, and others (Dey, 2016; Praveena & Jaiganesh, 2017).

3.3.2 Unsupervised Learning

Unsupervised learning comparative to supervised learning is much difficult where the goal of the ML model is to make a computer learn how to perform a specific task without any supervisor, only data is provided without labels (García et al., 2015). Furthermore, the aim of unsupervised learning is to find, existing relationships, irregularities, similarities, and regularities in provided input data (García et al., 2015). The unsupervised learning mechanism is utilized for forecasting of data, clustering of data, filtering, and for decision-making, where input data is hypothesized from which phase-based models can be developed as result of its training (Awad & Khanna, 2015).

According to García et al. (2015), apart from clustering, and association problems in unsupervised learning, there two other problems which can fall into its umbrella i.e. pattern mining and outlier detection. Clustering in unsupervised learning is performed, when the aim is to divide the input data to set of groups and clusters rather perform prediction (García et al., 2015). The created clusters reflect the shared properties within the input data, the working is performed by measuring the multivariate distance/gap between the set of observations (García et al., 2015). Whereas association rules are particularly aimed to find the association within the data. For example, if we are to determine the likelihood of customer buying an item: customer bought X item, he/she would also buy Y item. Moreover, association rules can also be utilized for possible sequential patterns in a set of data (García et al., 2015).

3.3.3 Semi-Supervised Learning

Semi-supervised learning is a hybrid learning paradigm for classification tasks (prediction) and descriptive analysis (clustering). Using this, different models can be designed for both labelled and unlabelled datasets (García et al., 2015). The semi-supervised learning paradigm operates across the guidelines of unsupervised and supervised learning. Due to the large quantity of unlabelled data from various origins like stock market data, biological data, message logs, and others, it has gained prominence in the field of data science (Awad & Khanna, 2015). Similarly, another mechanism called Active learning has the same objective as semi-supervised learning, attempts to select examples of most relevance from an unlabeled dataset, though the selected examples require human expertise for querying (García et al., 2015).

3.4 Parking Prediction

The main purpose of estimating available parking spaces has been known for decades (Ionita et al., 2018). There are some existing research on this topic, where researchers have collected data from smartphones to estimate parking availability through statistical and machine learning algorithms (Ionita et al., 2018). Apart from smartphone-based data, data from traffic and parking sensors have also been used for predicting the parking spaces (Xiao, Lou, & Frisby, 2018). The parking spaces can be located in different places. For example, a parking space can be on a road or in a monitored parking garage. The parking prediction research has two themes, one dealing with the prediction of parking spaces on streets and the second one with parking spaces available in garages (Badii et al., 2018). According to Badii et al. (2018), performing prediction on parking spaces from garages is a lot easier than the one on the streets, because parking spaces in garages are properly monitored through sensors or tickets. However, parking on the street requires the installation of distributed sensors. However, the availability of data for parking prediction is not easily available especially in the case of parking garages (Badii et al., 2018).

To address the issue of predicting parking spaces, researchers have used different types of probabilistic and machine learning algorithms. The street parking problem was addressed in San Francisco by measuring occupancy rate i.e. occupied parking spaces over total spaces (Chen, 2014). The work was implemented for the specific geo-located region using the autoregressive integrated moving average (ARIMA), linear regression, support vector regression, and feedforward neural network (FNN). Among all the experimented model, FNN produced the most accurate results. A study by Ionita et al. (2018), used data from parking sensors, traffic data containing information about traffic flows, weather data, fuel prices, revenues, and events which can impact the parking occupancy. The parking data was collected from a project called SF*Park*, which contained data for more than two years from on-street parking meters. They used clustering techniques along with machine learning models, such as support vector machines, multilayer perceptron, decision trees, and gradient boosting algorithm to approximate parking situation in the streets (Ionita et al., 2018).

Another study by Xiao et al., (2018), built a prediction framework using Markov M/M/C/C queueing model. The data used to predict the parking occupancy change was historical data from one parking garage located in San Francisco (Xiao et al., 2018). Similarly, to address the issue of predicting available parking spaces different models were used by (Badii et al., 2018). The data used for performing the predictive analysis was from 12 parking garages for a period of approximately three months. The prediction of available parking spaces was performed using Bayesian regularization artificial neural network (BRANN), support vector regression (SVR), recurrent neural network (RNN) and ARIMA model (Badii et al., 2018). The developed models were found to be reliable and capable of providing predictions for different time periods, such as 24 hours, 1 hour, 30 minutes. Moreover, the implemented solution was deployed to smart city applications in Tuscany and Florence, Italy to encourage vehicle drivers for efficient mobility.

3.5 The current state of Data Lakes and Parking prediction

Along with the many advantages of data lakes, they still have some challenges as well. Despite the technological advancements and availability of various frameworks for constructing a general data lake, it still requires a lot of technical effort to implement a data lake (Hai et al., 2016). The other main challenge of data lakes is to provide continuous support for both batch real-time data processing. The data lakes are associated with Apache Hadoop with the aim to make data available first for processing based on MapReduce, which is useful for batch processing (Madera & Laurent, 2016). In this regard, in some cases, the support of real-time processing in data lakes may be limited, which brings the need to implement data lake with hybrid functionality of batch and real-time processing (Madera & Laurent, 2016).

These days, smart cities are producing a large amount of data due to the integrated IoT devices, environmental sensors, traffic sensors, parking sensors, economic platforms, social networking platforms, and others. The data generated from these sources are diverse in nature, with different file formats, different structure, and varying frequencies (Lim et al., 2018). Therefore, maintaining the quality of data, integrating data from such sources, dealing with security, and meeting the end user's need is challenging (Lim et al., 2018). In addition, the data from these sources makes data integration a major challenge towards big data analytics due to their heterogeneous nature. In this regard, a data management platform with support data integration is needed to make the generated data from smart cities reusable form decision-making.

The concept of parking prediction has been there for decades, many researchers have tried to address this problem using different statistical and machine learning techniques (Ionita et al., 2018). In this thesis, different existing studies were reviewed on the topic of parking prediction for street parking and parking garages. The study by (Chen, 2014) was focused on-street parking occupancy rate i.e. occupied parking spaces over total spaces using different machine learning techniques, where FNN produced the best results. Another, similar study to address the street parking situation was conducted by Ionita et al (2018). They used the historical parking data along with fuel prices, weather condition, traffic counters, and revenues to estimate future parking situations.

On the other hand, a study by Xiao et al (2018) was focused on predicting parking spaces in parking garages. However, the used parking data was only from one parking garage and it was historical, no other data sources were taken into account. Similarly, Badi et al (2018) developed a parking prediction model using different data sources i.e. traffic flows, parking data from garages, and weather conditions. The results from this study were plausible and developed model was integrated into a smart city application providing the estimation for the next 24 hours, 1 hour, and 30 minutes. However, the geographical information was not taken in to account, which can impact the parking availability.

The purpose of this study is to address existing challenges and gaps in parking prediction and data lakes. The parking prediction model in this study will provide estimation for next quarter (15 minutes) using heterogeneous data sources to provide better results. The parking prediction will be performed using a data lake, which will provide the functionality of both batch and real-time analysis. Moreover, the reported challenges of reusability and data integration will also be addressed by integrated suitable technology solutions. The main challenge in parking prediction reported by J. Li et al (2018) is lower accuracy and required computational cost. The proposed data lake with a data pipeline approach for predicting available parking spaces is highly scalable, cost-effective i.e. do not require additional add-ons, the configured solutions with the support of custom scripts can be used for disseminating results. Furthermore, to provide better accuracy in terms of parking prediction, different data sources are used which are described in Section 6.1, Table 2. In addition, the developed prediction model for available parking spaces and data lake can be used for providing prediction every 15 minutes through an end-user application. The detailed requirements drawn from literature and identified the problem for Cork County are described in the next chapter.

4. The Artefact

The main purpose of implementing the artefact in this study is to support the prediction of parking space availability from diverse sources. The artefact being implemented in this study is a data management platform that operates on top of Linux OS and supports different functionalities of data analytics. The developed artefact will allow collecting data from heterogeneous sources through standard python-based libraries and big data technology solutions. Moreover, the collected data from different technological solution will be then stored into a data repository, from which the users can access the accumulated data for performing different kinds of data analytics tasks like data exploration and analysis. Secondly, the developed platform also enables its user to perform data processing and data analysis through different analytics techniques in both sequential (batch) and real-time manner. Finally, the developed platform will be capable of addressing the challenges of data integration, data reusability, and real-time data collection and analysis highlighted in Section 3.1.1 and 3.5.

4.1 Requirements of Data Management Platform

The purpose of the artefact is to utilize different big data technologies to enable the process of data ingestion, data processing, data analysis, data management using a single platform, without requiring additional costly add-ons unlike in data warehouses. Furthermore, the main objective of the developed platform is to provide support for performing predictive analysis using heterogeneous data sources, which in this study is predicting parking space availability. The identified requirements for the data management platform based on the conducted literature review in Chapter 3 are described in the following sub-sections:

4.1.1 Data Characteristics

The developed data management platform should be capable of supporting data from diverse sources with diverse formats. The data for performing predictive analysis in this study is environmental, geographical, and sensor data. Moreover, the included data have different representation formats, such as hypertext markup language (HTML), JavaScript object notation (JSON), and geographic JavaScript object notation (GeoJSON). Furthermore, these data sources are updated with different frequencies: static, which is not updated frequently until there is a new update. Furthermore, data from two data sources are updated every hour. Whereas one data source contains real-time streams from parking sensor, this data source is updated following a push-based mechanism i.e. it is written as updated. Therefore, the developed platform should allow to collect and integrate data in meaningful manner for further usage. Apart from data sources finalized for this study, the platform will also support for collecting and processing the data from sources presented in other formats like comma-separated values (CSV), tab separated values (TSV), extensible markup language (XML), Shapefiles, and others for pursuing other research objectives.

4.1.2 Technical Requirements

The technical requirements in this study are set based on the data characteristics and the need to collect, process, and analyse data both sequentially and in real-time. More details are provided in the following sub-sections:

Data Collection Mechanism

The designed platform and chosen technology for data collection should provide different approaches to ingest data into the storage repository regardless of what format or with what frequency data comes from its origin. This may require development custom scripts to extract the data from the data origin (e.g. webpage, application programming interface (APIs), and etc.). Similarly, as described in Section 4.1.1, the platform should allow the collection of data from real-time streams, data which is less dynamic and not updated often (e.g. geographical information), and historical data.

Reliability and Fault-tolerance

The chosen technologies for ingesting data into the platform should be reliable and fault-tolerant. As the data for this study comes from different sources with varying frequencies, it is possible that during the ingestion process loss of data may occur due to different issues e.g. network connectivity, unavailability of data from the data provider, problems with custom written scripts, and others. Therefore, the designed platform should be reliable and fault-tolerant. The advancements in the field of big data have made significant improvements, there are various data ingestion frameworks available, such as Apache Flume, Apache Kafka, and Apache Nifi.

Toolset for Data Processing and Analysis

The data in real-world cannot be always accurate, it can be inaccurate, incomplete, may contain redundancy, and inconsistent. The chosen technological solutions in the implemented platform should allow to pre-process, integrate, and harmonize the data through custom scripts using standard programming libraries. There different tools available which can be used to address the issues of the veracity of data, such as Apache Storm and Apache Spark described in Chapter 3.

The data management platform should enable its user to perform the tasks of data exploration and data analysis. The data lakes with traditional approaches are more focused on batch processing, to produce more throughput. But often, the issues of variable latencies of data are not addressed, one reason is their close association with the Hadoop ecosystem. Therefore, platforms with real-time analysis capabilities are limited, raising the need for hybrid data lakes which can address both batch and real-time analysis.

Parking Prediction

The main purpose of implementing a data lake in this study is to support the prediction of parking space availability from diverse sources. Therefore, it should provide technological choices for data collection, data ingestion, data processing, data analysis, and visualization.

As the data sources for predicting available spaces are heterogeneous, which will require metadata management and data integration tasks. Metadata can be defined as, the information regarding the collected data i.e. its structure, data type, in case of measurements its units, and others. Metadata management plays an important role in the data integration process, as it contains information regarding the data in scope. Therefore, the developed platform should allow integrating data for further usage. Furthermore, if the collected data have different time representations, it will require data harmonization steps. Thus, the chosen technologies for processing and analysis shall also allow addressing these issues.

The summary of described requirements is listed in Table 1. The listed requirements from conducted literature review and based on the needs for addressing the identified parking prediction problem in Table 1 are also used to evaluate the artefact in study. The details on evaluation process are described in Chapter 7.

Table 1. The requirements for predictive analysis

Requirement ID.	**Requirements for the data management platform**
R-1	The included data sources should be available for crawling i.e. open data.
R-2	The data lake created in this study should allow the user to collect data from data providers into the platform, irrespective of their format.
R-3	Based on the different frequencies of data sources in this study and occurring network exceptions, the chosen data collection technologies should be fault-tolerant. Moreover, the platform should be available all the time from data collection.
R-4	The platform should allow the user to store data into the platform in raw format.
R-5	As the data in this study will grow over time, so the developed platform should be scalable.
R-6	The platform should allow pre-processing the stored data including cleaning, transformation, and data integration.
R-7	The platform should be developed considering the need for both batch and real-time processing and should enable the use of different standard libraries for performing the analysis.
R-8	The platform should provide support for data visualization
R-9	The platform should provide support for developing custom applications.
R-10	The developed platform should be secure and capable of handling any intrusions.

5. Architecture

The designed architecture of data management platform in this study can perform data collection, data ingestion, data exploration and analysis, and data visualization tasks for data generated from heterogeneous sources. The proposed architecture for data lake implementation is based on the Hadoop ecosystem.

Data management platform is a cluster, comprised of four virtual machines, each as a single host is networked to create a distributed cluster. The developed cluster is comprised of four virtual machines, each with two virtual cores, 100 GB storage space, and 16 GBs of RAM. Linux OS (Ubuntu) is installed and configured on each host to develop a connection between the four hosts so that they can communicate with each other. Following to that, Cloudera's CDH (Cloudera Distribution including Hadoop) containing different parcels, such as Zookeeper, Hue, Apache Flume, HDFS, Apache Spark, and others are installed to form a Hadoop cluster.

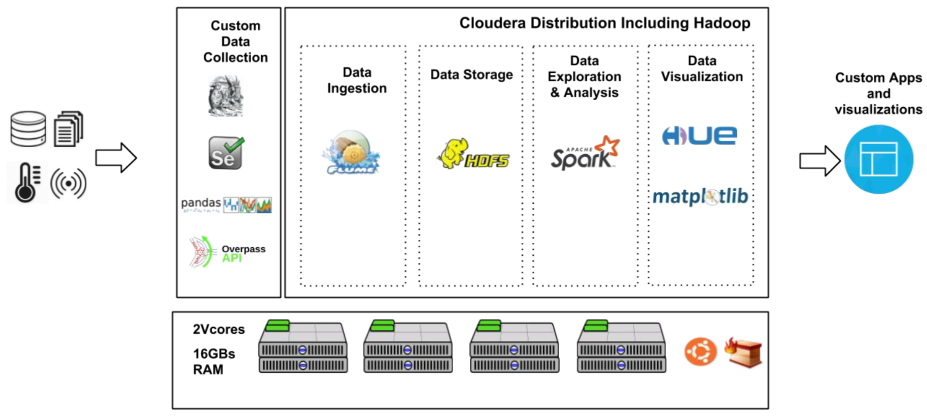


Figure 6. Architecture for data collection, data storage, and data analysis, redrawn from (Mehmood et al., 2019)

As can be seen from Figure 6, the current version of the data lake follows the data pipeline approach. The data generated from disparate sources, such as databases, static file, and sensor readings are first collected through custom written scripts using standard programming libraries. Following that, data ingestion tool (Apache Flume), injects the data to HDFS in a raw format making it reusable for future purposes. The stored data in HDFS then can be used for performing different kinds of processing and analytics tasks. The generated results from data exploration and analysis stage can be visualized using Hue and a Python-based library called Matplotlib. Additionally, designed data management also support the development of custom applications and visualization to aid in decision-making. The custom applications can be developed for specific domains e.g. policymakers are interested in the environmental activities from the past 10 years till the current for predicting the situation in coming years. The following sub-sections provided a detailed description of the implemented data management platform.

5.1 Custom Data Collection

The custom scripts for data collection are written outside the Hadoop ecosystem, as can be seen in Figure 6. The custom scripts are written for data available through third-party services and require scripts for data retrieval. Data collection for predicting available parking spaces includes Python-based scripts written extract the data from data providers. The used set of libraries for retrieving data from identified sources i.e. weather data, traffic counters data, parking data, and geographical information from open-source maps are Beautifulsoup4, Pandas, Selenium, and Overpass API. These tools are used for extracting data from webpages and APIs. In cases where human-like interaction is required to navigate through different web pages for extracting the intended data, Selenium with BeautifulSoup4 is used. Whereas, data sources with APIs, python-based Pandas is utilized for receiving the response from API in JSON or GeoJSON format. Moreover, if retrieved data requires basic restructuring and formatting, mentioned libraries are used before injected it to data lake.

5.2 Data Ingestion

The data ingestion is defined as a process of injecting data in both batch and real-time manner to a storage repository, which in this study is HDFS. In the implemented data management platform, the data ingestion process is performed using Apache Flume. As described in Section 3.2.1, it is considered as a reliable and fault-tolerant solution for inserting data to HDFS from external sources. It has three main components i.e. source, sink, and channel. The source is responsible for fetching data from a data provider, which is then passed to the sink through a specified channel to insert the collected data to HDFS. The complete architecture of Apache Flume has been described in Figure 3.

5.3 Data Storage

The data in the implemented platform is stored in HDFS, which allows storing diverse kinds of data in raw format. It is considered a reliable solution for storing large-scale data. The data in HDFS is stored in folder directories, where each folder directory contains data from disparate sources in their raw format. However, to efficiently manage and track the data, folder and file naming conventions can be followed, which can also help in maintaining the metadata of respective data sources.

5.4 Data Exploration and Analysis

The data exploration and analysis in the implemented platform is carried out using Apache Spark. As described in Section 3.2.3, it is used for analysing large-scale data in a sequential and real-time manner. The cluster-based computing architecture makes it efficient for analysing large-scale data. Due to its support for various kinds of transformations processing data becomes easier specifically when you are dealing with large datasets. As in this thesis, the predictive analysis for available parking spaces requires different kinds of processing tasks, such as data cleaning, data transformation, data integration, and data harmonization. Apache Spark provides support for data exploration and all these data processing tasks through its advance python-based libraries and Spark SQL.

Apache Spark allows its user to analyse data in scope using its spark.ml and spark.MLlib, by providing access to different statistical and machine learning methods. Apache Spark is not only efficient for solving complex data problems, but it also shortens the distance from exploratory to operational analytics. It also increases the productivity of data scientist especially, when its upper libraries are used for implementing complex algorithms and pipelines.

5.5 Data Visualization

The generated results from data exploration and analysis are delivered using Hue and python-based library Matplotlib. Both tools allow visualizing results from basic querying to advance data analytics efficiently. Hue provides a user-friendly interface to visualize the stored data in HDFS using Apache Solr and Lucene libraries (Sirisha & Kiran, 2018). Using different built-in plugins, the user can create dynamic dashboards comprised of different kind of visuals like bar and pie charts, face-based grouping, and others. In the context of this thesis, as data for predictive analysis is from four different sources, containing disparate parameters, hue can be used for getting initial insights about the data. Whereas, Matplotlib can be used for creating different types of visualization especially when performing sequential data analysis. The created visuals can be then integrated with custom applications depending upon the requirements of decision makers.

6. Implementation

The purpose of this thesis is to perform predictive analysis for available parking spaces enabled by the developed data management platform. The developed platform contains a set of big data technology tools as part of Cloudera’s CDH along with external support for configuring other tools. Its web-based graphical user interface (GUI) allows maintaining the cluster from anywhere through the internet. For example, a script for collecting data is configured and during the collection process, one of the services gets down, due to exceptions from a data provider or internal resources. The user can access the platform through a web user interface and investigate the issue for its resolution.

Additionally, the platform is aimed to provide support for decision-making. As these days various cities have adopted the concept of smart cities which are creating a large amount of data from traffic sensors, environmental sensors, social platform, economic platforms, and others. The developed platform is general and applicable to other urban cases. The developed artefact mainly relies on Hadoop ecosystem, and the scripts for data collection are implemented using Python programming language. Whereas for data processing and analysis spark SQL and Python for Spark which is called pyspark are used. The completed process of data analysis to predict available parking spaces is demonstrated in Figure 7. The detailed explanation on the implementation of data analytics pipeline is mentioned in following sub-sections.

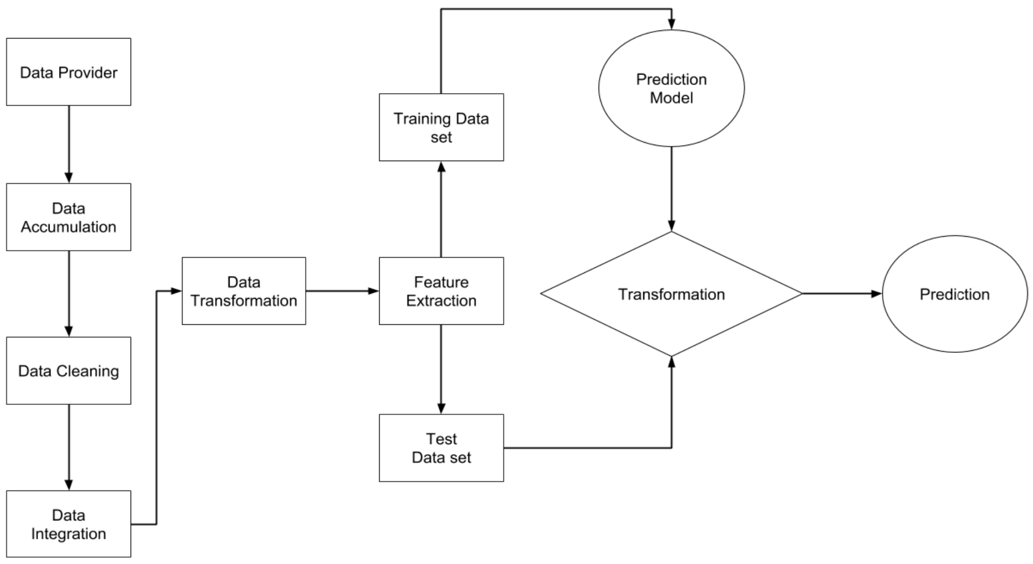


Figure 7. The flow of predictive analysis for parking prediction

6.1 Data Accumulation

The data accumulation phase involves the collection of data from identified data sources. To retrieve data from their origins, scraping scripts were developed to extract the data available in different formats using Python-based standard frameworks described in Section 5.1. The scraping scripts were designed in such manner, that for every data source, the available information after retrieval is stored in CSV format. The information data sources and for each data source which python frameworks were used, is described in Table 2.

Table 2. Information on identified data sources from Cork County

Data Source	Origin	Format	Used Framework
Weather data	Weather station	HTML	Beautifulsoup4 and Pandas
Traffic data	5 traffic counters	HTML	Selenium and Pandas
Parking data	8 parking garages	JSON	Pandas
Geographical information	OpenStreetMap	GeoJSON	Overpass and Pandas

The weather data available from one of the data providers from Ireland was collected using Beautifulsoup4 and Pandas. The scraper for this data sources operates by sending a request to uniform resource locator (URL) where the information is available. Following that, the available information at the designated URL is parsed and converted into CSV format with required formatting. The second data source containing traffic counts requires human-like interaction to reach the intended webpage, where information is available. For this purpose, a combination of Selenium and Pandas is used to retrieve data in CSV. Similarly, the third data source which provides real-time parking occupancy information as it is updated. The data from this source is available through a web API in JSON format, for retrieving the information API URL is requested and received a response is then transformed to CSV format. Moreover, the received JSON response, in this case, is quite nested which required implementation of data normalization procedures.

Finally, the fourth and last data source for predicting parking availability contains information about amenities e.g. markets, parks, government service centres, hospitals, educational institutes, and others. This information is extracted from OpenStreetMap using overpass API and Pandas. Data from this source is extracted by defining a radius of 150 meters for each parking garage, which means that information of all amenities in the proximity of 150 meters is collected. This is used to determine the impact of particular amenity on each parking garage e.g. assuming a parking garage is located near markets and parks, its occupancy rate will be high compared to others especially on weekends. To enable the analytics model for determining this impact, each amenity extracted from OpenStreetMap is associated with defined classes. For example, places like fast food, continental restaurant, and cafes are classified as food service. Similarly, if there are convenience stores, book stores, toy shop, and grocery store they are classified as a market place, the same approach is applied to other types of amenities. The defined amenity classes used for classifying different places are listed below:

- Food service (used for classifying food and eating places)
- Entertainment service (used for classifying pubs, bars, and gaming places)
- Transportation service (car rental service, travel agencies, and others)
- Educational service (school, colleges, libraries, and others.)
- Market place (shops, grocery stores, convenience store, and etc.)

- Customer service (banks, ATMs, and others)
- Recreational place (parks, and outdoor leisure places)
- Workplace (offices, business centres, and others)

After associating amenities with each identified class their ratio is calculated with the total amount of classes, to make the predictive analysis more efficient.

The data is stored using a defined approach by Gilman et al. (2018), which allows ingesting retrieved data from all the data sources to HDFS depending upon their acquisition frequencies. The used approach is described in following sub-sections.

6.1.1 Manual Data ingestion

This approach is used for data sources which is updated rarely by the data provider, such data source is inserted to HDFS using web interface Hue. In case, data needs to be formatted and transformed to CSV format, custom scripts are used and then data is ingested to HDFS. In the context of this thesis, only one data source is ingested manually i.e. geographical information from OpenStreetMap, as data on maps are not updated frequently.

6.1.2 Automated Data ingestion

The automated data ingestion approach is used for data sources, where the script is written to load data automatically. Such sources have particular schedule i.e. data is updated by data provider every minute, hourly, daily or monthly. The scraping scripts for data sources with certain schedules are configured with cron scheduler, where the cron scheduler executes the scraping scripts as defined. As soon as the data arrives in the local disk – spooling directory from data source provider, it is picked up by Flume agent configured in the cluster. A sample configuration script is provided below:

```
spoolAgent.sources = splSrc
spoolAgent.channels = splChannel
spoolAgent.sinks = sinkToHdfs
spoolAgent.sources.splSrc.channels = splChannel
spoolAgent.sinks.sinkToHdfs.channel = splChannel
spoolAgent.sources.splSrc.type = spooldir
spoolAgent.sources.splSrc1.channels = splChannel
spoolAgent.sources.splSrc.spoolDir = “”
spoolAgent.sinks.sinkToHdfs.type = hdfs
spoolAgent.sinks.sinkToHdfs.hdfs.fileType = DataStream
spoolAgent.sinks.sinkToHdfs.hdfs.path = “”
spoolAgent.channels.splChannel1.type = file
spoolAgent.channels.splChannel1.checkpointDir
spoolAgent.channels.splChannel1.dataDirs = “”
```

As it can be seen from the above-provided configuration, the spooling directory is configured, from where the source in Flume agent picks the data as it arrives, which is later passed to the HDFS sink and injected into the defined folder directory in HDFS. In this thesis, the described approach is adopted for three data sources i.e. weather data, traffic count data, and parking count data. The flow of data ingestion using Apache Flume is demonstrated below in Figure 8.

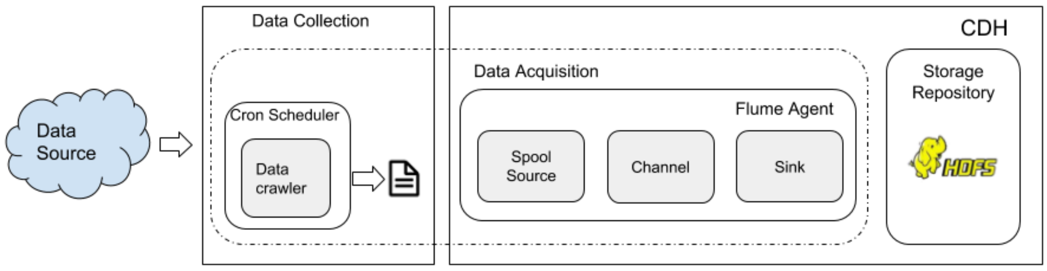


Figure 8. Data accumulation approach using Apache Flume, redrawn from (Gilman et al., 2018)

6.2 Data pre-processing

The data pre-processing techniques are used for data preparation so that the provided input data fits the chosen analytic method. The real-world data quality cannot be guaranteed, often it contains noise, inconsistency. Data preparation in data analytics is considered as an essential step, to resolve the issues of noise, inconsistency, misinformation, the presence of special characters, and formatting in the data. In this thesis, the following techniques are used to prepare data for analysis and ensure data quality.

- Data cleaning
- Data integration
- Dealing with missing values
- Data transformation

6.2.1 Data Cleaning

Data cleaning involves different operations, such as removal of errors, inconsistency, formatting issues, and duplication for improving the quality of data. According to Abedjan et al. (2016), two main types of data errors were identified from the included data sources: duplication and pattern violation. Duplication of data is referred to as the recurrence of exact same observation which has been reported earlier. Whereas the pattern violation is more related to the formatting of data, misspelling, or violation of semantic rules.

In this study, the data is first stored to HDFS and then it is closely inspected for the possible errors. After inspecting the stored data, duplication of records, the presence of special characters in numerical observations, and missing values were found. To address these errors, a combination of Spark SQL and Pandas were used. As described in Section 3.2.3, Spark SQL allows preparing data and features for analysis. The identified errors were removed using Spark SQL and Pandas in rare cases. For example, in weather data, wind measurements are always encapsulated in brackets which can make the learning process of the analytic model hectic. Moreover, in the case of parking data duplication of records were found, which required cleaning. There were also cases of missing values, where few of the observations didn't have reported parameters for weather data specifically. Additionally, one case related to duplication was also found in parking data, where the reported reading from one parking sensor was repeated for three consecutive days. After identification of all errors in data, the errors were removed from the data. A line chart was created to explore the data, Figure 9 shows the example of a maximum number of vehicles passing through one highway (N08) from the first three hundred records.

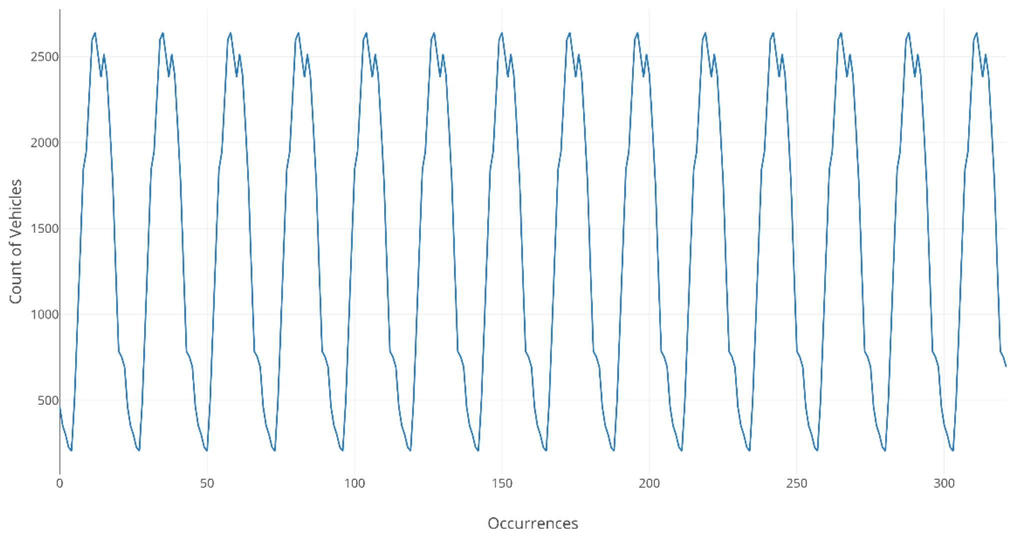


Figure 9. Visualization of cleaned traffic data

6.2.2 Data Integration

Data integration is performed to combine the data from disparate origins having different kinds of representations, it is considered as a standard procedure to process data from heterogeneous sources (Kadadi et al., 2014). However, it is difficult to integrate data in a meaningful manner, which may involve data model development, data transformation, and semantic matching (Kadadi et al., 2014).

In this study, the necessity of data integration was observed after inspection of data from all four included data sources. For example, in weather data date and time were collectively reported as a single entity with hourly frequency. Whereas, in traffic data time and date are reported as two separate entities with hourly frequency. The data from parking garages were reported in real-time every minute with date and time a single entity. This clearly shows that to use data from these three data sources for further usage, they need to be integrated with similar time-frequency which can be hourly or per minute. On the other hand, for traffic data where date and time are reported separately, they needed to be merged so they can have similar nomenclature and representation. Similarly, the data from maps are collected and store based on radius from each parking garage. So, to prepare data from analysis, all these data sources needed to be integrated. The following approach was following for integration:

- Data and time from traffic data were combined using ISO-8601 format, which is also standard reporting time for parking and weather data
- A nomenclature was defined for date and time column i.e. for each data sources date and time will be reported as “dateTime” so integration can be performed smoothly

- For integrating geographical data containing a list of amenities, each parking garage name was matched with a column containing garage name in parking data
- To harmonize the time frequencies of weather, traffic, and parking data, it was finalized to integrate these data sources with per minute reporting frequency, which resulted in many null records. As weather and traffic data are reported with hourly frequency, when integrated with parking data with per minute frequency, it left various null records. This showed the need for implementing data transformation methods.

After creating the final data frame containing reported parameters from the weather, traffic, parking data, and geographical information, data transformation was performed, which is described in the next section.

6.2.3 Dealing with missing values

In cases, where few data observations are known but few are not, interpolation is done to estimate the missing observations based on the known observations.

As described in Section 6.2.2, in the context of this study, data integration is performed to harmonize different observations based on reported date and time. The weather and traffic data are integrated using per minute frequency from parking data observations, which resulted in many null values. To fill up the missing values, an estimation method called interpolation is used (Howard, 2017). The observation for each hour from weather and traffic data is sorted with matching hour in parking observations. Following that, the linear interpolation method is applied to the data frame created from data integration stage. Linear interpolation is defined as a curve fitting method to estimate and construct missing observations within the range of known observations. The formula of linear interpolation for two known values is mentioned below, where x, x_1 , y_1, y_2 are known values to estimate the unknown value y. However, the sequence can be continued till nth number, depending upon the requirements.

$$y = y_1 + \frac{(x-x_1)(y_2-y_1)}{x_2-x_1} \tag{1}$$

6.2.4 Data Transformation

In pre-processing, data transformation is done to convert or consolidate the data for making the data processing more efficient. Data transformation can be of various types, it can be linear, quadratic, and others (García et al., 2015).

The parking data is from eight different parking garages, each with different space limit e.g. total capacity of parking vehicles in one of the parking garages is 376, another parking garage is of 935 vehicles, and so on. As machine learning methods will be used for data analysis, it may be become confusing for the model to determine the impact of each parking garage because of different parking capacities. To handle this problem, the ratio of free spaces with a total number of spaces for each parking garage is taken for the analysis.

Beside that some transformation was also performed on date and time, as the data that is of each minute, for analysis to have a clear impact of each observation, reported time is shifted to the nearest quarter e.g. if a reading is observed at 12:09 it will be shifted to 12:15 and if its lower than half of the quarter (12:06) it will be shifted to 12:00. Moreover, the reported data for every observation is converted to the name of the day e.g. if a measurement reported on 26th January, then it will be represented in the final dataset as "Friday".

6.3 Integrated data set for predicting available parking spaces

After applying data pre-processing techniques explained in Section 6.2, the final integrated data set is created. The overview of integrated data set containing features to implement predictive analysis for available parking spaces is described in Table 3.

Table 3. Overview of integrated data set for predictive analysis

Category	Variable Name	Description of Variable
Date and Time	Day	Day of week (Monday – Sunday)
	Time	Hours and transformed minutes to a nearest quarter
Parking observations	Parking ratio	The ratio of parking observations against the total capacity of each parking garage
Weather condition	Wind speed	Wind speed in Knots
	Rain	Amount of rain (millimetres)
	Pressure	Pressure measurement in Hectopascals
	Atmospheric temperature	The measured temperature in Celsius
	Humidity	Percentage of humidity in the city
Traffic flow	Traffic count	Count of vehicles passing through the particular road
Geographical information	Food service	The ratio of restaurants/food places against the total number of amenities in the region
	Market place	The ratio of shops and malls etc. against the total number of amenities in the region
	Transportation service	The ratio of taxi, car rentals, and others against the total number of amenities in the region
	Entertainment service	The ratio of cinemas, theatres, and others against the total number of amenities in the region
	Educational service	The ratio of school, colleges, and others against the total number of amenities in the region
	Community service	The ratio of churches, welfare centres, and others against the total number of amenities in the region
	Customer service	The ratio of ATMs, banks, and others against the total number of amenities in the region
	Recreational place	The ratio of parks against the total number of amenities in the region
	Work place	The ratio of offices against all number of amenities

One example from the integrated data set, containing parking spaces ratio with atmospheric weather is shown in Figure 10. As it can be seen, that on Tuesday with colder temperature the parking availability decreased. Furthermore, following figure was plotted after pre-processing stage to see if there are any abnormalities in the data. The results from the Figure 10 showed the data to be pretty much organized and cleaned without any duplication present.

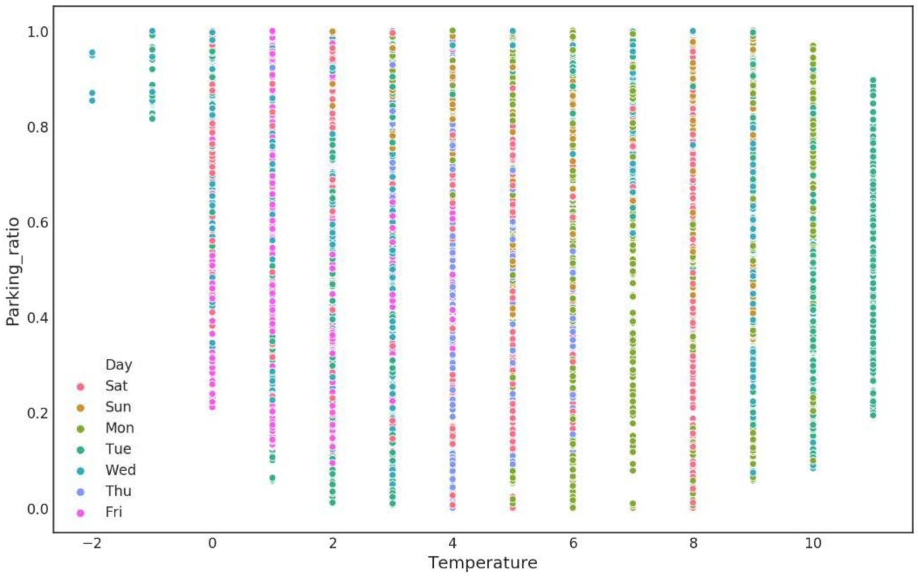


Figure 10. Data from temperature and parking spaces, where parking ratio represents the ratio of available spaces and temperature from Cork County specified by day

6.4 Predictive Analysis

The implemented platform provides support to analyse large-scale data, both in batch and real-time through Apache Spark. Its machine learning APIs (spark ml and spark MLlib) allows analysing large-scale data at low cost.

The predictive analysis for predicting available parking spaces in this study is done using Apache Spark's machine learning API (spark.ml). Using spark.ml makes the data analysis simpler because of its multi-stage pipelines, the pipelines works as the end-to-end process (Meng et al., 2016). The end-to-end process may include different data processing and analysis tasks like feature extraction, learning of models, cross-validation, tuning, and others. This makes the whole process of data analysis inexpensive and developers can even integrate their own algorithms (Meng et al., 2016).

The predictive analysis for this study has been performed using machine learning methods to predict the available parking spaces from real-world data generated from traffic counters, parking sensors, weather stations, and geospatial information from maps. As described in Chapter 3, there various statistical and machine learning methods to perform predictions. The prediction in this thesis has been performed using different regression models. The list of used regression models is listed below:

1. Linear Regression
2. Decision Trees

3. Random Forests
4. Gradient-boosted Trees

6.4.1 Identification of key traffic counter

Principal component analysis (PCA) is a statistical method used for reducing multi-dimensional data by determining the correlation present between each dimension (Bartholomew, 2010). It works by synthesizing the available observational variables and captures the most relevant information from provided observations i.e. the provided variables are ranked by the calculated covariance, following to that their eigenvalues are calculated and sorted in a descending order to find the new m dimensions. The original data with n number of dimensions is then transformed into the new dimension (Bartholomew, 2010).

In the context of this study, the PCA is performed for the data from traffic counters. The data from traffic counters is from 5 counters, placed on different locations in the city. Based on assumptions, usually traffic flows are related to each other e.g. people every day in the morning go out for work and educational purposes, so the density of traffic on road may be similar each day. Therefore, PCA is performed to find the correlation between the included 5 traffic counters. While performing the PCA, there can be a chance that one observational variable due to higher values may dominate the other observational variables. In this regard, to make sure the performed analysis is accurate, the scaling of data from traffic counters is performed. The implementation of PCA is done using Apache Spark's libraries.

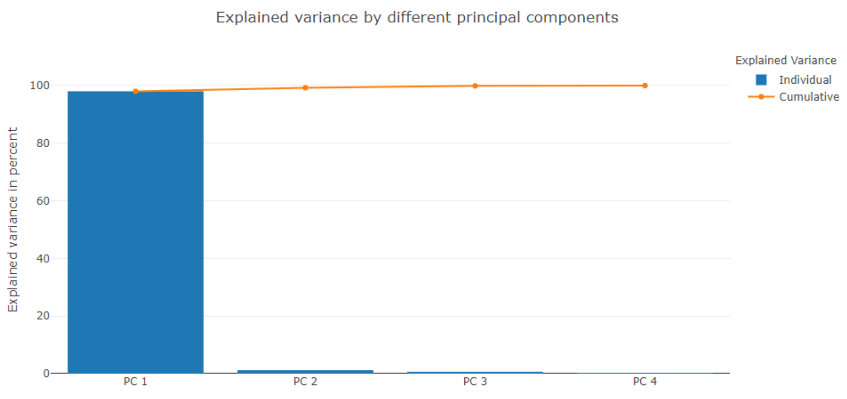


Figure 11. Principle component analysis on traffic counters

As shown in the above Figure 11, principal component 1 explains the most variance with a percentile of 97 approximately. Whereas as the other three components still contain some information, which is comparatively minimal, and can be discarded for further analysis. The last fifth component was already discarded because it was bearing almost 0 per cent of the information. Based on the results from PCA, it can be said that all of the 5 traffic counters are highly correlated, and they have pretty similar patterns of traffic flows. To pursue the predictive analysis for parking prediction, one traffic counter is used.

6.4.2 Parking prediction using Linear Regression

The linear regression model works by determining the relationship between variables e.g. there are two variables X and Y, if there is an increase in variable X, Y will also increase or increase in X caused a decrease in Y (Bruce & Bruce, 2017). This can also be measured by using correlation i.e. how both variables are related to each other. However, both approaches are different in nature, correlation is used to determine the strength of the relationship between both variables whereas regression approach actually helps in quantifying their relationship nature (Bruce & Bruce, 2017).

The linear regression using estimation determines the impact of variable Y on variable X i.e. if there is a change in X variable it changes Y (Bruce & Bruce, 2017). The equation of linear regression is mentioned below:

$$Y=b_0+ b_1 X \tag{2}$$

As it can be seen from the above-mentioned equation, Y represents the value to be predicted, b_0 is intercepted constant and b_1 is calculated slop for variable X.

Model Construction

In this study for parking predictions, X variable consists of data from traffic counters, weather stations, and geospatial data containing information about amenities where Y variable contains data from parking sensors (which is being predicted). The linear regression model was implemented using a spark.ml package.

The integrated dataset ref (Section 6.3) is partitioned using a random split approach with 70% as training set and 30% as a test set. After the partition of data, the data is used to prepare a feature set containing data for weather conditions, traffic counts, and data from amenities in the city, which are used for other regression models as well. The extracted feature set is used as the X variable whereas variable Y contains data from parking sensors.

The model is implemented using a 10-fold cross-validation technique for predicting parking space availability. With this technique, the partitioned training set is converted to ten sub-sets, where the model is trained on 9 subsets and 1 subset is used for validation purpose. The cross-validation technique iterates over 9 subsets to train the model for performing predictions and validates the model with 1 subset left for validation. The main benefit behind using this approach is that all observations present in the data are utilized for training and validation, and each observation is validated. After the completion of the process, results are analysed to get the best-trained model, which can be the later used against test set partitioned initially.

The initial predicted results from the model were not plausible. Because of this, each variable was plotted against parking sensor data to see an abnormality in the data. The observations from wind speed and rain were a bit skewed. To resolve the skewness in wind speed and rain observations, different data transformation approaches were applied such as log transformation, square root transformation and cube root transformation. Among all these transformation techniques, the results from cube root transformation helped in normalizing the data. Following that, the training process was again repeated to compute the predictions. The computed predictions from linear regression model are presented in the Figure 12, where label represents the real data from parking sensors:

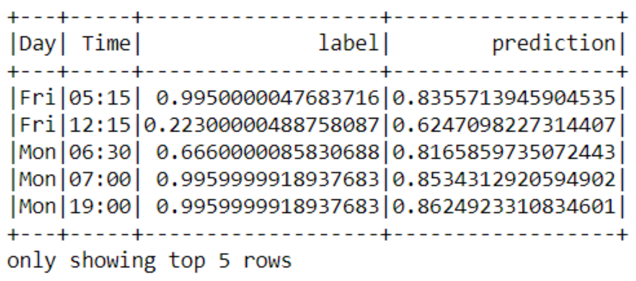

```
+---+-----+------------------+------------------+
|Day| Time|             label|        prediction|
+---+-----+------------------+------------------+
|Fri|05:15| 0.9950000047683716|0.8355713945904535|
|Fri|12:15|0.22300000488758087|0.6247098227314407|
|Mon|06:30| 0.6660000085830688|0.8165859735072443|
|Mon|07:00| 0.9959999918937683|0.8534312920594902|
|Mon|19:00| 0.9959999918937683|0.8624923310834601|
+---+-----+------------------+------------------+
only showing top 5 rows
```

Figure 12. Prediction results from the Linear Regression Model

6.4.3 Parking prediction using Decision Trees

The decision trees are used for solving both classification and regression problems. The decision trees are used for regression problems, where the relationship between independent and dependent variables is non-linear and linear models are not able to capture the relationship (Tamminen, Laurinen, & Röning, 1996). When using decision trees for classification or regression problems, no additional data transformation is required rather data can be used in raw format (Lytvynenko, 2016).

In cases, where variable to be predicted is dependent on continuous values (numerical values), these can be solved using decision tree regression method (Lytvynenko, 2016). The decision tree regression – regression trees are binary in nature. The tree is constructed by creating a partition of the input data into smaller subsets using the independent variables (Tamminen et al., 1996). The independent variables in this study are values from weather data, traffic data, and data containing information regarding amenities. The regression trees are constructed recursively with top-down and divide and conquer approach (Lytvynenko, 2016). After splitting the input data into subsets, the next step is to find the nodes that are possibly similar to a variable to be predicted (Tamminen et al., 1996). The complete regression tree is implemented by splitting all nodes in an iterative manner, to get the maximum decreased value of R(T), where R(T) contains the total sum of all sums of squares from all formed nodes (Tamminen et al., 1996). The completed formula is mentioned below, where x is given features for tree construction, t represents each node, y_i is the target variable (label) for a particular instance, $\bar{y}$ is the mean of y, N is the count of instances:

$$R(T)=\frac{1}{N}\sum_{x_i \in t}(y_i-\bar{y}(t))^2 \qquad (3)$$

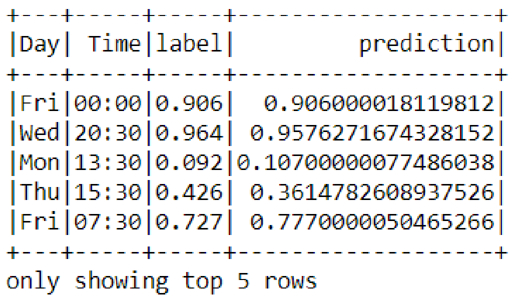
The decision trees are capable of providing most important features from the provided features i.e. after the tree is constructed, the data is divided into two groups (continuous or categorical), which makes it easier to determine the most important features (Tamminen et al., 1996). There are more advantages of using decision trees for solving regression problems e.g. decision trees can tackle missing values present in the input data by traversing the split and ranks it with the probability of falling into each category.

Model Construction

The implementation of decision tree regression for parking prediction is done using a spark.ml package. Using spark.ml package, the feature construction and training of the model can be performed in one go as an end-to-end process. The input data for decision tree for parking prediction is taken after data pre-processing (ref Section 6.3), however, no additional data transformation is performed unlike in the linear regression model.

After getting the data from data pre-processing stages, a ten-fold cross-validation approach-based machine learning pipeline was constructed to perform predictive analysis for free parking spaces. While constructing the ten-fold cross-validation pipeline, an array for depth of tree and number of trees was constructed to find the best suitable model. The array for depth of tree contained values from 1 to 11 and array for a number of trees contained values maximum of 20. The training of the model was performed using described parameters, after completion of training. The left subset for testing (just like it was done linear regression) is transformed for prediction.

The results from decision tree regression are mentioned in Figure 13, where the label represents the original observational values of parking sensors from eight parking garages. The presented results are generated from the best model, which was formed with a depth of 11 and 20 number of trees.



```
+---+-----+-----+-------------------+
|Day| Time|label|         prediction|
+---+-----+-----+-------------------+
|Fri|00:00|0.906|  0.906000018119812|
|Wed|20:30|0.964| 0.9576271674328152|
|Mon|13:30|0.092|0.10700000077486038|
|Thu|15:30|0.426| 0.3614782608937526|
|Fri|07:30|0.727| 0.7770000050465266|
+---+-----+-----+-------------------+
only showing top 5 rows
```

Figure 13. Prediction results from Decision Tree Regression

6.4.4 Parking prediction using Random Forests

The random forests have been developed based on ensemble learning approach that aggregates the results from multiple created classifiers (Liaw & Wiener, 2002). There are two main methods in ensemble learning boosting and bagging. Using the boosting method, the subsequent tree assigns extra weight to values which are incorrectly predicted by previous predictors. Following that, the final prediction is created by taking vote from assigned weights (Liaw & Wiener, 2002). Whereas, with bagging each tree is formed independently through bootstrap samples, without having any dependency on previously created trees. After all, the trees are constructed, the prediction is computed by taking the majority vote (Liaw & Wiener, 2002).

The main purpose of using bagging approach is to produce an unbiased model with reduced variance (Breiman, 2001). The random forests are built on the idea of bagging with additional randomness attributes e.g. in standard trees, the nodes are partitioned by making the best choice from input variables, whereas in random forests each node is split by randomly choosing the best from a subset of predictors in respective node (Liaw & Wiener, 2002).

The random forests algorithm works by creating a bootstrap sample to providing training data, and then trees are constructed from that bootstrapped data in a recursive manner (Breiman, 2001). The algorithm randomly selects the most suitable variable from the chosen data and splits into further sub-nodes (Breiman, 2001). After the random forest's trees are formed, the prediction can be performed against the target variable. The complete formula for random forests regression is shown below, where x represents the point where prediction is to be performed, b = 1, B is the number of instances (trees), T_b is bootstrapped data to grow random forest tree, and $\hat{f}_{rf}^{B}$ is the predicted value for given x:

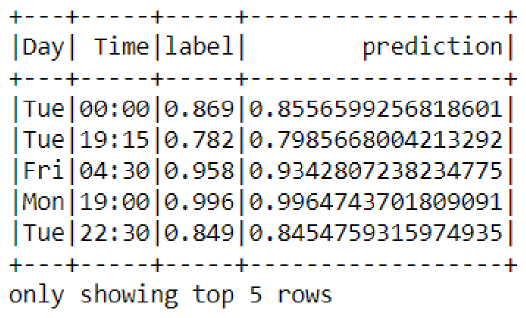
$$\hat{f}_{rf}^{B}(x)=\frac{1}{B}\sum_{b=1}^{B} T_b\,(x) \tag{4}$$

The random forests do not require any additional data transformation for performing prediction.

Model Construction

To predict the available parking spaces, data is taken after it has been prepared from data pre-processing stage, described in Section 6.2. The prepared integrated data (ref Section 6.3) is split as 70 % training set and 30 test set. After data is randomly partitioned, ten-fold cross-validation approach is applied to train the random forests regression model. The ten-fold cross-validation approach is implemented using a spark.ml pipelines, which is comprised of feature extraction and training of the model.

The pipeline works by first extracting the features from the training set, after the construction of the feature set, it is passed to the model for training. The training of the model is done by testing it for maximum depth of 1 to 11 and the maximum number of trees 20. After the completion of the training process, the best model is chosen for computing predictions on left data as the test set. The best model was produced with a depth of 11, a maximum number of trees 20. The prediction results are shown in Figure 14.



```
+---+-----+-----+------------------+
|Day| Time|label|        prediction|
+---+-----+-----+------------------+
|Tue|00:00|0.869|0.8556599256818601|
|Tue|19:15|0.782|0.7985668004213292|
|Fri|04:30|0.958|0.9342807238234775|
|Mon|19:00|0.996|0.9964743701809091|
|Tue|22:30|0.849|0.8454759315974935|
+---+-----+-----+------------------+
only showing top 5 rows
```

Figure 14. Prediction results from Random Forests Regression Trees

6.4.5 Parking prediction using Gradient-boosted Trees

The gradient-boosted trees are a machine learning technique developed on ensemble learning approach like random forests. However, gradient-boosted trees are constructed following a boosting approach instead of bagging approach. It has gained significant popularity due to its accuracy and efficiency for both classification and regression problems (Si et al., 2017). The main idea of gradient-boosted trees is to combine multiple weak predictors to produce a more robust model with greater accuracy (Dabiri & Abbas, 2018).

The trees in the gradient-boosted method are built in a forward manner, each tree is created incrementally to cover up the weaknesses of the previously created tree (Sangani, Erickson, & Al Hasan, 2017). For each iteration, the sum of individual predictions is taken for computing overall prediction – D(x), a weight is given to weak predictor which indicated the error rate of computed predictions, and then they are reweighted to determine computed incorrect predictions (Sangani et al., 2017). The next constructed tree will try to minimize the error left by the previous tree, which is F(x) in the equation (4). The final prediction is improved by retraining the model to reconstruct the difference between current prediction and the target function, which is called residual i.e. R(x). The complete equation for this process is mentioned below:

$$D(x) = d_{tree1}(x)+d_{tree2}(x)+\ldots \tag{5}$$

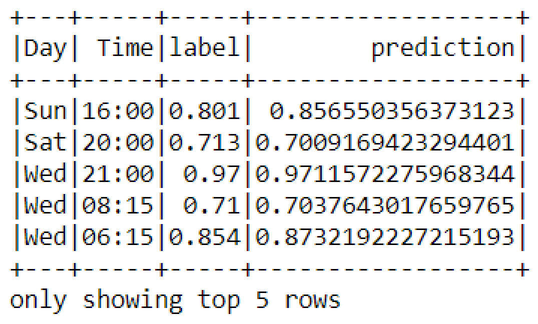
$$F(x)=D(x)+ d_{tree3}(x)$$

$$R(x)=F(x)-D(x)$$

Each iteration to construct gradient-boosting tree is fitted to the computed residuals R(x). The described process is repeated until enough regression trees are added, resulting in a small error (Persson, Bacher, Shiga, & Madsen, 2017). The overfitting of the model is avoided by scaling the input received from each constructed tree with factor n (0-1) (Persson et al., 2017).

Model construction

Similar to other tree implementations, no additional data transformation techniques are applied to data. The integrated data set is picked after data pre-processing stage, described in Section 6.3. The data is randomly partitioned as 70% training set and 30 % test set. After splitting the data, machine learning pipeline for gradient-boosted regression tree is constructed. The pipeline consists of two stages, in first stage features are extracted from the training set. Following that, the ten-fold cross-validation technique is used for training purpose, the model is trained by providing an array of maximum depth and maximum tree to get the best model. The input values for maximum depth and a maximum number of trees are similar random forest, described in Section 6.4.4. As soon as the best model is produced, the trained model is transformed on the left test set initially to generate predictions. The prediction results from gradient-boosted trees are shown in Figure 15.



```
+---+-----+-----+------------------+
|Day| Time|label|        prediction|
+---+-----+-----+------------------+
|Sun|16:00|0.801| 0.856550356373123|
|Sat|20:00|0.713|0.7009169423294401|
|Wed|21:00| 0.97|0.9711572275968344|
|Wed|08:15| 0.71|0.7037643017659765|
|Wed|06:15|0.854|0.8732192227215193|
+---+-----+-----+------------------+
only showing top 5 rows
```

Figure 15. Prediction results from Gradient-boosted Regression Trees

7. Evaluation

The artefact in this thesis is of two folds, first one is to develop a data lake for data collection, data ingestion, data processing, data analysis, and data visualization. Secondly, utilize the developed platform to perform predictive analysis for available parking spaces using heterogeneous data sources. The artefact requirements were identified based on conducted literature review and for predicting available parking spaces, which are describe in Chapter 4. The development of this artefact is done in iterative cycles, which means that if requirements of one cycle are not met then the next cycle cannot be pursued.

The developed artefact is evaluated in two stages. The first stage is to validate the developed artefact against the defined requirements in Chapter 4. Evaluation in the second stage is more related to the performed predictive analysis. As the implemented machine learning techniques for predictive analysis are regression models, where different regression models are experimented to predict the available parking spaces. The used machine learning techniques are evaluated using regression evaluator metrics such as root mean square error (RMSE), R-square, and mean absolute error (MAE). However, apart from these evaluation procedures, the developed artefact is tested after completion of each cycle. Additionally, as the artefact was implemented in iterative cycles along with DSR method, the evaluation of the implemented study is also highlighted using the evaluation patterns by Sonnenberg and vom Brocke (2012).

7.1 Validation against the Requirements

The developed artefact is evaluated by analysing the defined requirements, how the developed artefact corresponds to the set requirements. The set requirements after implementation are marked with a status, depicting whether it has been completed, has been partially completed or has not been completed. Each requirement is thoroughly examined and analysed for providing the status and what needs to be done for improvement.

Table 4. Evaluation against requirements

Requirements ID = Status (Completed (C)/ Partially completed (PC)/ Incomplete (I)				
R-1 = C	R-2 = C	R-3 = PC	R-4 = C	R-5 = C
R-6 = C	R-7 = C	R-8 = C	R-9 = C	R-10 = PC

R-1: The included data sources should be available for crawling i.e. open data.
Status: Completed
Evaluation: The identified data sources are open data, which means that they can be re-used for any purpose. The data from these sources is provided either through APIs or web pages. The access points for data sources are also public enabling user to download it any time. The data from these sources can be scraped for further usage, however, it requires extra effort for developing custom scripts.

R-2: The artefact to be implemented for this study should allow the user to collect data from data providers into the platform, irrespective of their format.
Status: Completed

Evaluation: The developed platform is capable of collecting data from a heterogeneous source with diverse representation formats. The current version of the platform allows the user to develop custom scripts to extract data programmatically. Moreover, the data sources with APIs can be configured with a push-based mechanism to collect data as soon as it's updated.

R-3: Based on the different frequencies of data sources in this study and occurring network exceptions, the chosen data collection technologies should be fault-tolerant. Moreover, the platform should be available all the time from data collection.
Status: Partially completed
Evaluation: The chosen technological solution – Apache Flume to accumulate data from a heterogeneous source with different acquisition frequencies is reliable and fault-tolerant. In case of failures, the lost data can be recovered from a created checkpoint by Flume agent. The custom written scripts to extract data from data provider is configured with cron scheduler to store files in spool directory, from where Flume agents fetch the data and write it HDFS. However, the accuracy of data still cannot be guaranteed e.g. if data provider changes the structure of intended data, advanced techniques and solutions are required to address this challenge. The current approach for data ingestion can be improved programmatically in custom scripts, that if the structure of data is changed, it should notify the user. Moreover, adding Apache Kafka can also add value to current implementation through its schema validation functionalities.

R-4: The platform should allow the user to store data into a platform in raw format through data ingestions tools and manually.
Status: Completed
Evaluation: The current version of the platform allows the user to ingest data into HDFS using Apache Flume. Moreover, the platform also provides support to integrate other data ingestion tools like Apache Kafka and Apache Nifi. To accommodate the requirement of inserting data manually, Hue was configured. It provides support to add files regardless of their format to HDFS through its user-friendly interface. Both approaches were tested, and current data in the platform is ingested using these approaches.

R-5: As the data in this study will grow over time, so the developed platform should be scalable.
Status: Completed
Evaluation: The current storage capacity of the platform is sufficient to accommodate the data for this study. However, as the data will grow with time and in future, if more data is to be integrated, the platform can be scaled in terms of storage, computational capacity, and others.

R-6: The platform should allow pre-processing the stored data in the platform including cleaning, transformation and others.
Status: Completed
Evaluation: The added data processing and analysis solution – Apache Spark allows performing various kinds of pre-processing such as feature extraction, data cleaning, data transformation, data normalization, and others. To verify the integrated solution, data included in this study was processed using its user-friendly python-based libraries, which fulfilled the requirements. Moreover, the platform is capable of integrating other processing solutions.

R-7: The platform should be developed considering the need for both batch and real-time processing and should enable the use of different standard libraries for performing the analysis.

Status: Completed
Evaluation: The integrated solution of data analysis – Apache Spark allows to perform both batch and real-time data analysis. Its ML libraries allow the user to analyse various set of problems, such as classification, pattern recognition, clustering, and regression. The predictive analysis in this study has been performed using its machine learning pipelines, which allows to process and analyse data in one go.

R-8: The platform should provide support for data visualization.
Status: Completed
Evaluation: The support for data visualization in this study is provided through Hue and Matplotlib. Both tools experimented, and they fulfil the requirements.

R-9: The platform should provide support for developing custom applications.
Status: Completed
Evaluation: The platform supports the development of custom applications. Through the use of REST APIs, the data and results can be delivered to third-party users depending upon their needs.

R-10: The developed platform should be secure and capable of handling any intrusions.
Status: Partially completed
Evaluation: To ensure the security of the platform, only users with whitelisted IPs can access the platform. This still requires further research to integrate more advanced techniques for securing the platform.

7.2 Evaluation of Predictive Analysis

In this study, the predictive analysis for available parking spaces has been performed using multiple regression models. The most popular metrics which are used to evaluate regression models are a MAE, RMSE, and R-squared (Botchkarev, 2018; Swalin, 2018). The MAE is calculated by measuring the difference between the set of predictions and actual observed values. The difference for each individual prediction and observed value is calculated first, and then their mean is calculated, given that each difference has equal weight (Swalin, 2018). On the other hand, RMSE is the standard deviation of calculated differences between predicted and residuals. Lastly, the R-squared metric is used for determining how well the chosen independent variables explain the dependent variable (target variable) (Swalin, 2018). The equation to calculate these evaluation metrics for regression models is shown below, where N is the count of given observations, y_i is the actual observation, $\hat{y}_i$ represents the predicted value, and $\bar{y}$ is computed mean for y:

$$\text{MAE}=\sum_{i=0}^{N-1} |y_i-\hat{y}_i| \tag{6}$$

$$\text{RMSE}=\sqrt{\sum_{i=0}^{N-1} \frac{(y_i-\hat{y}_i)^2}{N}}$$

$$R^2=1-\frac{\sum_{i=0}^{N-1}(y_i-\hat{y}_i)^2}{\sum_{i=0}^{N-1}(y_i-\bar{y})^2}$$

In this study, different regression models are applied to predict available parking spaces. For each applied regression model, the regression model evaluation metrics are calculated, which are described in Table 5.

Table 5. Results from evaluation metrics for experimented models

Metric Name	Linear Regression	Decision Tree Regression	Random Forests	Gradient-boosted Regression Tree
MAE	0.156025	0.0273134	0.0319873	0.0132115
RMSE	0.197572	0.0550783	0.0518326	0.0302239
R-squared	0.311599	0.946407	0.952537	0.983862

Apart from evaluating the implemented regression models for predicting available parking spaces, to have a closer look at produced predictions from each model. The scatter plots are plotted to determine how the predicted values and observed values are inter-related to each other. The generated plots for each model are shown below:

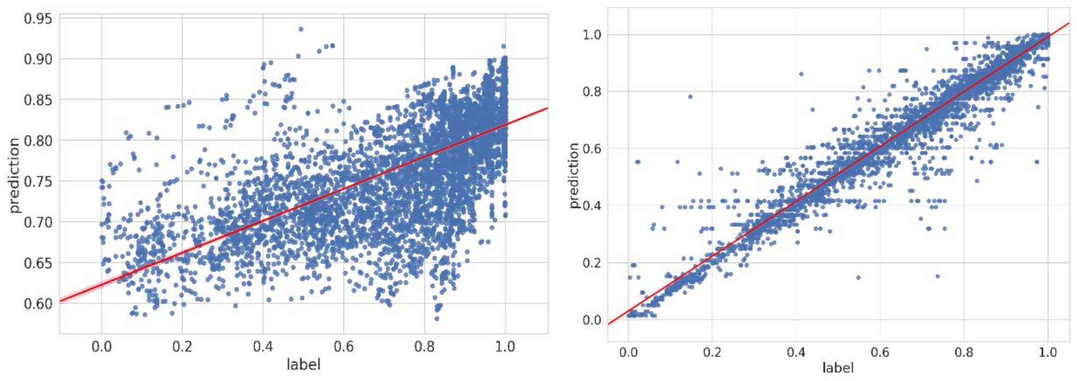


Figure 16. Scatter plots for evaluation of predicted values (left) Linear Regression Model (Right) Decision Tree Regression Model

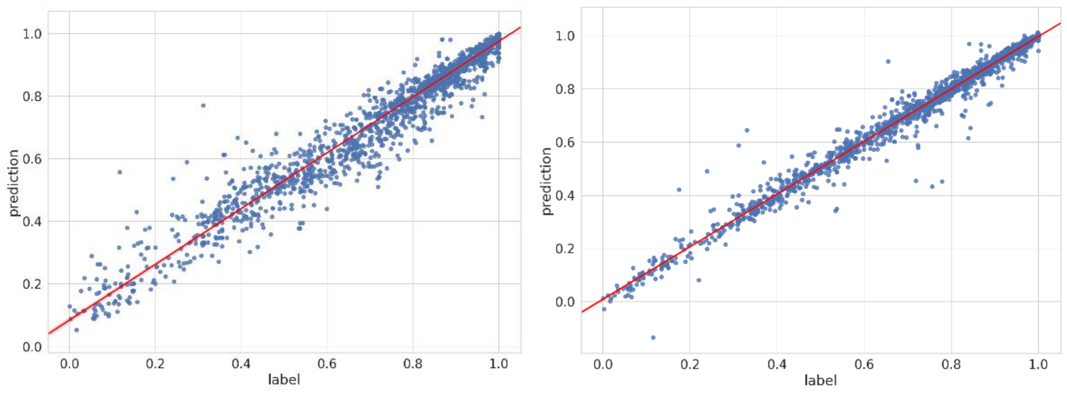


Figure 17. Scatter plots for evaluation of predicted values (left) Random Forests (Right) Gradient-boosted Regression Tree

7.3 Evaluation activities in DSR

The artefact was implemented in iterative cycles along with DSR method, the evaluation of the employed method is also highlighted using the evaluation patterns by Sonnenberg and vom Brocke (2012). The design process has been evaluated using evaluation activities presented by Sonnenberg and vom Brocke (2012). The evaluation activities are described in Section 2.4. Sonnenberg and vom Brocke (2012) have suggested that evaluation should be done in two phases: ex-ante evaluation and ex post evaluation. In ex-ante evaluation the identified problem and design for the intended artefact is evaluated. Whereas, in ex post evaluation the actual implementation of artefact and its usage is evaluated. Based on the provided evaluation activities the developed research has been evaluated, which is described in following sub-sections.

7.3.1 Evaluation activity 1

The first evaluation activity is focused on validating the identified research problem whether it is an actual problem which needs to be addressed. As a research problem in this study has two main themes, the first one was to perform a predictive analysis for available parking spaces on gathered data for identified parking occupancy problem from Cork County. The identified problem was verified from a local administrative person and a news article published by Roche, (2018). Hence, these findings led to pursue this study. Secondly, to design a data lake to enable the process of data accumulation, processing, analysing, and disseminating the generated results for predicting available parking spaces using heterogeneous data sources. This was cross-checked with existing literature and it showed that there is a need to address challenges of data collection, data processing, and data analysis for large-scale data and data produced from heterogeneous sources. Moreover, difficulties of data integration and data processing from collected data were also reported from a review conducted by Lim et al. (2018).

7.3.2 Evaluation activity 2

The evaluation activity is performed as part of ex-ante evaluation, right after the identified research problem is evaluated. The main objective of this activity is to evaluate the proposed design for addressing the research problem. After evaluation of first activity, clear high-level requirements were outlined to create the design for the artefact to be constructed. Based on the identified requirements from conducted literature review and parking occupancy problem in Cork County, the design of artefact was constructed, which is described in Chapter 5 and was checked against the defined requirements. The defined requirements in Chapter 4 were strictly followed to construct the artefact.

7.3.3 Evaluation activity 3

The third evaluation activity is conducted as a post evaluation after the artefact is constructed to demonstrate its functionality. The developed artefact was evaluated against the defined requirements, as described in Section 7.1. Furthermore, to evaluate whether developed artefact – a data lake – can perform the required operations, to conduct predictive analysis. The data from four different data sources was collected and processed to predict available parking spaces. The predictive analysis was performed using one of the integrated big data solutions in the data lake. The generated results from the analysis were evaluated using standard machine learning evaluation techniques, as highlighted in Section 7.2.

After this evaluation activity, the developed artefact is pushed for usage in a practical environment. However, the implemented data analytics technique has been evaluated in an artificial setting which requires further field test in Cork County, after deployment in real-environment. Therefore, the fourth evaluation activity was not conducted because it is done after the artefact is used in practice. This has been left as future work.

8. Discussion

In this chapter, the identified research questions mentioned in Chapter 1 are answered. Furthermore, the benefits of the artefact along with few limitations are also described, and how can these limitations be improved for producing better results. The implemented artefact in this study has two main parts, first one is a data lake to accumulate data into a storage repository from heterogeneous sources and enable the user to perform different kinds of analysis and provide results using the same platform. Second, part is focused on identified research problem i.e. to perform a predictive analysis for available parking spaces. Besides that, the artefact is implemented using DSR method, to employ this method guidelines provided by Hevner et al. (2004) are followed. The evaluation of followed guidelines is also discussed.

8.1 Discussion on research questions

RQ1: In what ways can big data analytics be used in predicting available parking spaces using machine learning techniques?

The main objective of this thesis was to perform a predictive analysis for available parking spaces from Cork County. To address the identified problem of available parking spaces from Cork County, a data lake was developed. However, the developed data management platform can also be used for other research objectives. Furthermore, the described limitations from previous research are also considered while implementing the parking prediction model, as described in Section 3.5.

In previous research studies on parking prediction topic, all factors were not taken into account to predict available parking spaces. Moreover, implemented prediction model by Badii et al. (2018) provided better results by using data about traffic flows, weather condition, and parking garages. The implemented model was integrated with a smart city application, as described in Section 3.4. However, the geographical information (information about markets, parks, and others) was not considered. Additionally, the generated predictions were limited to 24 hours, 1 hours, and 30 minutes. Whereas, the parking prediction in this study has been performed using different data sources such as weather conditions, traffic flows, real-time parking data, and geographical information from maps. The reason behind using this approach was to predict parking spaces by analysing the current situation of parking spaces as well as the factors which influence the parking occupancy. For example, it is possible that in good weather conditions, people may go out more frequently to markets or parks than in bad weather situations. Moreover, the implemented model is capable of providing prediction for next 15 minutes, which is more feasible and can be improved in future.

The analysis has been done using four regression models i.e. linear regression, decision tree regression, random forests, and gradient-boosted regression trees. Before performing the analysis, PCA was performed which showed that the included traffic counters were highly related to each other. Therefore, data from only one traffic counter was used. Upon experimenting stated regression models, gradient-boosted algorithm produced the best results. Initially, the linear regression model was applied but the results were not accurate due to the complex relationship present between the chosen data sources. As can be seen from Figure 16 (left), the plotted predicted values against original observations do not show a strong correlation. Additionally, to determine the accuracy of linear regression, evaluation methods were applied to the model which showed satisfactory results but still required improvement. The evaluation results described in Table 5 show that the chosen independent variables were not able to explain the target variable very much with R-square of 0.31. Whereas if the same evaluation metric is compared with other models, with decision trees the score was 0.94, random forests 0.95, and gradient-boosted 0.98 produced best results than other models. Similarly, for each model RMSE was calculated to analyse the average error magnitude given a range of 0-1.

The results among all the models showed that again the best-received results closer to 0 were from gradient-boosted regression trees, ref Table 5. This can be further compared by analysing the scatter plot for predicted values and observed values of parking sensors. From Figure 17 (right), it can be seen that from gradient-boosted regression trees, almost each generated prediction is highly correlated with original observation, stating the most accurate predictions. The evaluated RMSE can further be validated through MAE from Table 5, which shows that the average magnitude of error from gradient-boosted is the lowest. In the context of the identified parking prediction problem in this thesis, gradient-boosted regression trees model showed the most accurate results for predicting available parking spaces, ref Table 5 and Figure 17 (right).

Based on the findings from the linear regression model, which showed a complex relationship between the chosen data sources. More non-linear methods like ARIMA, FNN, and ANN can also be applied to the stated problem. If results from this study are compared with a study conducted by J. Li et al. (2018), the implemented prediction model in this study provides better results using heterogeneous data. Whereas, the implemented model by J.Li et al. (2018) used only parking data, weather condition, and data about holidays. The RMSE error produced from gradient-boosted regression trees is 0.03, whereas they used long short-term memory network (LSTM), which produced RMSE of 5.42 from predicted results. However, the implemented LSTM model by J. Li et al. (2018) is developed on time series data whereas the best model of this study disregards all temporal dependencies. Additionally, it depends on the chosen data sources, local situations, and applied data preparation techniques.

The current implementation of predictive analysis in this study still requires further testing (field test) to ensure its accuracy in real-world. Further experimentation for parking prediction will be conducted after a field test, which is left for the future.

RQ2: What kind of technological solutions can be used for designing a data management platform enabling parking spaces prediction?

The data management has become a complicated process due to increased velocity and variety of data. Every day new data is created from various sources such as social networking platforms, sensors, IoT, stock exchanges, and many others. The generated data from such sources are increasing each day, therefore creating the demand for analysing the data to address routine problems of mankind. Despite, the advances in the field of data science managing such diverse data is still a challenge. Along with this, the concept of smart cities is being adopted by various cities in the world to improve government services and their citizen's livelihood, and smart cities are using data generated from such sources. Often, some government service offices use legacy databases which are difficult to be integrated with modern solutions. To address all such challenges, an artefact has been developed in this thesis.

The developed artefact is capable of accumulating data from heterogeneous sources, which can be HTML pages, CSV, TSV, JSON, GeoJSON, and shapefiles. Furthermore, the developed data lake also allows storing data that is provided manually from data providers. The chosen technologies such as Selenium, Pandas, and beautifulsoup4, with support of Apache Flume allows to store data in the platform directly. Furthermore, the chosen approach addresses the limitations of making data reusable from disparate sources, which has been highlighted by (Lim et al., 2018).

The support for storing large-scale data for batch and sequential data processing has been addressed using HDFS, which allows storing data in a raw format without the need for additional processing. In this way, the stored data in raw format can be used by different users with different requirements for producing effective results.

From the literature, it was identified that Apache Spark allows to process and analyse large-scale data available in the diverse format in both batch and real-time manner. After clearly analysing its benefits and limitations, Apache Spark was integrated to enable data processing and analysis tasks. The main challenge of processing large-scale data from heterogeneous sources is to integrate data in a meaningful manner. Despite, many implemented big data platform in smart cities, challenges of data integration, data privacy, and meeting end-users requirements have been reported from conducted reviews (Lim et al., 2018). While designing the platform these challenges were taken into consideration, and Apache Spark was found to be suitable.

The other reported challenge in literature is visualizing large-scale data, which may not contain noise in generated visuals. This has been accommodated by configuring Hue and Matplotlib. In a few cases, it is possible that the results could be used by third-party services. The current version of the platform allows to port data results using REST API, and by configuring solutions like HBase. To ensure the security of the platform, currently only whitelisted IPs are allowed to access.

The chosen technologies for implementation, however, do have some limitations such as, the data ingestion tool – Apache Flume requires additional disk space if spool source is used for injecting data to HDFS. However, it provides different alternative approaches to directly ingest data to HDFS without storing the data to the local disk first. Additionally, alternative solutions like Apache Kafka and Apache Nifi can be experimented to address issues of schema validation and scalability. The configured visualization at this point are general in nature but more advanced visualization solutions like Kibana and Grafana can still experiment. Lastly, the security of the platform has been done using a standard firewall, which can still be improved in future.

8.2 Design Guidelines

In this thesis, the DSR method and seven guidelines provided by Hevner et al. (2004) were used as the research method. In this section, the applied guidelines are discussed i.e. how the presented design guidelines by Hevner et al. (2004) are met for this thesis.

8.2.1 Design as an artefact

The implemented artefact in this study enables the process of predicting available or free parking spaces using heterogeneous data sources from Cork County. The developed data lake allows to accumulate data from heterogeneous sources with different acquisition frequencies. Moreover, it is capable of providing support for the operations of data storage and different kinds of analysis. In addition, the platform also allows providing visually aided results along with support for developing custom applications. While designing the data management platform, the drawn requirements from literature review described in Chapter 4 were strictly followed to address the existing limitations described in Section 3.5.

8.2.2 Problem Relevance

The identified research problem represents the need to understand the challenges of collecting data from heterogeneous sources and data produced within smart cities. Therefore, to store such data into a single repository in raw formats for further usages without integrating additional costly add-ons. Furthermore, the problem related to parking spaces in Cork County is an actual problem which needs attention. Designing a data management platform to address such urban issues especially, vehicle parking was a suitable solution, which can help governing bodies and policy-makers to have evidence-based insights into the actual needs of the city.

8.2.3 Design Evaluation

The developed artefact was evaluated thoroughly in two ways. Firstly, the developed data management platform was evaluated against the defined requirements i.e. by analysing how well the defined requirements for the platform are met. Secondly, the applied machine learning techniques for predictive analysis were evaluated using standard evaluation metrics such as MAE, RMSE, and R-square.

8.2.4 Research Contributions

There are two main contributions to this study, the first contribution is related to the identified urban problem of finding available parking spaces. It has been addressed using machine learning techniques to predict future situations about parking occupancy so that the generated information can be used for designing parking management strategies. The second contribution is the development of data lake for accumulating, processing, analysing, and visualization data from disparate sources. Apart from predicting available parking spaces, the developed data management platform can be used to address various urban problems.

8.2.5 Research Rigor

This study was conducted using DSR methodology along with seven guidelines provided by Hevner et al. (2004). The knowledge base for this study was previously conducted researches on the topic of data lakes, big data, big data pipelines, smart cities and smart city platforms, and machine learning methods. The artefact was implemented and evaluated rigorously by following the requirements and created a knowledge base.

8.2.6 Design as a Search Process

The knowledge base was used for selecting different big data technology tools to design and implemented a data lake in this thesis. However, an efficient data lake which addresses all existing challenges in big data and smart city platforms will have a wider scope than this thesis. The available open source big data tools were utilized to design the platform. Secondly, the predictive analysis has been done using different machine learning techniques, and the technique provided better results than earlier studies, which will be used in practice.

8.2.7 Communication of Research

The thesis is documented in a way that both technical and non-technical readers can take advantage of it. To maintain the flow of the reader, the main purpose of the thesis has been highlighted in almost every chapter. However, still, some domain-specific terminologies have been used, which require a bit prior knowledge from the reader. The limitations of reviewed studies on the topic of parking prediction and data lakes were also explained, and how the highlighted challenges were addressed in this study (ref Section 3.5, Chapter 4, and Section 8.1).

9. Limitations and Future work

In this thesis, the artefact was built to predict available parking spaces using disparate sources. The developed artefact is a general data management platform which also provides support for data collection from diverse origins in both batch and real-time manner and utilizes the gathered data for further usage. However, just like every other thing, the developed artefact does have some limitations.

The current version of the platform is capable of accumulating data from heterogeneous sources, each with different acquisition frequency. The data ingestion process is accommodated using Apache Flume, which requires additional disk storage for storing the extracted data first to the spooling directory. This can, however, be managed by constructing a custom data ingestion source, which will inject data directly to HDFS. Secondly, the change in the structure of data is often done by data providers. In such cases, more advanced data management approaches are required like schema validation. In future, alternative solutions like Apache Kafka and Apache Nifi can be integrated into the platform for addressing such issues.

The configured data visualization tools in the platform are capable of addressing the current needs, however, more advanced solutions like Kibana and Grafana can also be integrated for delivering interactive results. The other major limitation of the data management platform is ensuring its security. The platform is currently, secured using a standard firewall and only whitelisted IPs are given access, and still, the platform can be vulnerable to different kinds of attacks. Therefore, more advanced approaches are needed to ensure the integrity of the platform and data in it.

The developed artefact enables performing predictive analysis for estimating the future situation regarding free parking spaces. The parking space availability has been predicted using different machine learning models, each model was tested in an artificial setting using evaluation metrics. Among applied machine learning models, gradient-boosted regression tree provided the most accurate results. However, the developed prediction model still needs to be integrated with an end-user application, where the results can be synced in real-time. Moreover, the implemented prediction model also requires field testing to ensure its applicability in real-world.

The described limitations will be addressed in future so that more researches can be performed using the developed data lake, and the prediction model for estimating available parking spaces could be integrated into the end-user application. This can help vehicle drivers in decision-making and will allow governing bodies in drawing better parking management strategies.

10. Conclusion

This study presents the approach to predict parking space availability, supported by a data lake solution. The implemented data lake contains different big data technology solutions for accumulating, processing, analysing, and visualising large-scale data from heterogeneous sources. The developed platform has been constructed by following the created high-level requirements. The current version provides hybrid functionality for data collection, data processing, and data analysis in both batch and real-time. In addition, based on the diverse nature of data origins, the data can be extracted through custom scripts and directly using the flume agent. Currently, the evaluation of the platform has been done against the defined requirements. In future, after addressing the current limitations, it can be compared to other similar data management platforms as well.

The developed artefact was used to address the identified research problem for predicting available parking spaces. The results from this study depict that, traffic flows from the selected region are highly correlated and can have a similar impact to their respective parking garages. The prediction process was implemented using different machine learning regression models. Due to the less linear relationship between selected data sources i.e. weather conditions, traffic data, amenities data (geographical information), and the ratio of observed parking occupancy, gradient-boosted regression trees presented the most accurate results. The developed model will further be integrated with an end-user application to support the vehicle drivers in decision-making and policy-makers for drawing better management strategies.

The data from smart cities can be utilized for addressing more urban issues similar to parking availability using the developed platform. The work on this research study will be continued in future to improve the implemented data lake and integrate the developed prediction model to a user application.

References

Abedjan, Z., Chu, X., Deng, D., Fernandez, R. C., Ilyas, I. F., Ouzzani, M., … Tang, N. (2016). Detecting Data Errors: Where are we and what needs to be done? In *Proceedings of the VLDB Endowment* (Vol. 9, pp. 993–1004). https://doi.org/10.14778/2994509.2994518

Awad, M., & Khanna, R. (2015). *Efficient Learning Machines*. Apress, Berkeley, CA. https://doi.org/https://doi.org/10.1007/978-1-4302-5990-9_11

Badii, C., Nesi, P., & Paoli, I. (2018). Predicting Available Parking Slots on Critical and Regular Services by Exploiting a Range of Open Data. *IEEE Access*, *6*, 44059–44071. https://doi.org/10.1109/ACCESS.2018.2864157

Bartholomew, D. J. (2010). Principal components analysis. *International Encyclopedia of Education*, 374–377. https://doi.org/10.1016/B978-0-08-044894-7.01358-0

Botchkarev, A. (2018). Performance Metrics (Error Measures) in Machine Learning Regression, Forecasting and Prognostics: Properties and Typology, 1–37. Retrieved from http://arxiv.org/abs/1809.03006

Breiman, L. (2001). Random forests. In *Machine learning* (Vol. 45, pp. 5–32). https://doi.org/10.1023/A:1010933404324

Brohi, S. N., Bamiah, M., & Brohi, M. N. (2018). Big Data in Smart Cities: a Systematic Mapping Review. *Journal of Engineering Science and Technology*, *13*(7), 2246–2270. Retrieved from https://www.scopus.com/inward/record.uri?eid=2-s2.0-85051514659&partnerID=40&md5=52a85e003bea6a0b2570d3c19800d572

Bruce, A., & Bruce, P. (2017). *Practical statistics for data scientists: 50 essential concepts*. O'Reilly Media, Inc.

Cenedese, A., Zanella, A., Vangelista, L., & Zorzi, M. (2014). Padova smart City: An urban Internet of Things experimentation. In *Proceeding of IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks 2014, WoWMoM 2014* (pp. 1–6). IEEE. https://doi.org/10.1109/WoWMoM.2014.6918931

Chen, X. (2014). Parking Occupancy Prediction and Pattern Analysis.

Cheng, B., Longo, S., Cirillo, F., Bauer, M., & Kovacs, E. (2015). Building a Big Data Platform for Smart Cities: Experience and Lessons from Santander. In *Proceedings - 2015 IEEE International Congress on Big Data, BigData Congress 2015* (pp. 592–599). IEEE. https://doi.org/10.1109/BigDataCongress.2015.91

Dabiri, S., & Abbas, M. (2018). Evaluation of the Gradient Boosting of Regression Trees Method on Estimating the Car Following Behavior. *Transportation Research Record*, *01*(540), 1–7. https://doi.org/10.1177/0361198118772689

Dey, A. (2016). Machine Learning Algorithms: A Review. *International Journal of Computer Science and Information Technologies*, *7*(3), 1174–1179. Retrieved from www.ijcsit.com

Fang, H. (2015). Managing data lakes in big data era: What's a data lake and why has it became popular in data management ecosystem. In *2015 IEEE International Conference on Cyber Technology in Automation, Control and Intelligent Systems, IEEE-CYBER 2015* (pp. 820–824). IEEE. https://doi.org/10.1109/CYBER.2015.7288049

Farid, M., Roatis, A., Ilyas, I. F., Hoffmann, H.-F., & Chu, X. (2016). CLAMS: Bringing Quality to Data Lakes. In *Proceedings of the 2016 International Conference on Management of Data - SIGMOD '16* (pp. 2089–2092). https://doi.org/10.1145/2882903.2899391

García, S., Luengo, J., & Herrera, F. (2015). *Data Preprocessing in Data Mining. Intelligent Systems Reference Library* (Vol. 72). https://doi.org/10.1007/978-3-319-10247-4

Gilman, E., Kostakos, P., Cortes, M., Mehmood, H., Byrne, A., Valta, K., … Kumar, C. (2018). *First version of the framework for the collection, cleaning, integration & anonymization of big data*.

Hai, R., Geisler, S., & Quix, C. (2016). Constance: An Intelligent Data Lake System. In *Proceedings of the 2016 International Conference on Management of Data - SIGMOD '16* (pp. 2097–2100). https://doi.org/10.1145/2882903.2899389

Hajirahimova, M., & Ismayilova, M. (2018). Big Data Visualization: Existing Approaches and Problems. In *Problems of Information Technology* (Vol. 09, pp. 65–74). https://doi.org/10.25045/jpit.v09.i1.07

Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design Science in Information Systems Research, *32*(4), 725–730.

Howard, J. P. (2017). Interpolation and Extrapolation. In *Computational Methods for Numerical Analysis with R* (pp. 95–132). New York: Chapman and Hall/CRC.

https://doi.org/10.1201/9781315120195-4

Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. *Computing in Science and Engineering*. https://doi.org/10.1109/MCSE.2007.55

Ionita, A., Pomp, A., Cochez, M., Meisen, T., & Decker, S. (2018). Where to Park?: Predicting Free Parking Spots in Unmonitored City Areas. In *Proceedings of the 8th International Conference on Web Intelligence, Mining and Semantics* (p. 22:1--22:12). https://doi.org/10.1145/3227609.3227648

Iqbal, R., Doctor, F., More, B., Mahmud, S., & Yousuf, U. (2017). Big Data analytics and Computational Intelligence for Cyber-Physical Systems: Recent trends and state of the art applications. *Future Generation Computer Systems*. https://doi.org/10.1016/j.future.2017.10.021

Ishizuka, Y., Chen, W., & Paik, I. (2016). Workflow transformation for real-time big data processing. *Proceedings - 2016 IEEE International Congress on Big Data, BigData Congress 2016*, 315–318. https://doi.org/10.1109/BigDataCongress.2016.47

Kadadi, A., Agrawal, R., Nyamful, C., & Atiq, R. (2014). Challenges of data integration and interoperability in big data. In *Proceedings - 2014 IEEE International Conference on Big Data, IEEE Big Data 2014*. https://doi.org/10.1109/BigData.2014.7004486

Kamala, V. R., & Marygladence, L. (2015). An optimal approach for social data analysis in Big Data. *4th IEEE Sponsored International Conference on Computation of Power, Energy, Information and Communication, ICCPEIC 2015*, (January 2015), 205–208. https://doi.org/10.1109/ICCPEIC.2015.7259464

Katal, A., Wazid, M., & Goudar, R. (2013). Big Data : Issues , Challenges , Tools and Good Practices. In *Sixth International Conference on Contemproray Computing (IC3)* (pp. 404–409). https://doi.org/10.1109/IC3.2013.6612229

Li, J., Li, J., & Zhang, H. (2018). Deep Learning Based Parking Prediction on Cloud Platform. In *2018 4th International Conference on Big Data Computing and Communications (BIGCOM)* (pp. 132–137). https://doi.org/10.1109/BIGCOM.2018.00028

Li, K., Deolalikar, V., & Pradhan, N. (2015). Big data gathering and mining pipelines for CRM using open-source. *Proceedings - 2015 IEEE International Conference on Big Data, IEEE Big Data 2015*, 2936–2938. https://doi.org/10.1109/BigData.2015.7364128

Li, X., & Wang, H. (2007). The model of the active differential data synchronization for the heterogeneous data source integration systems. *Proceedings of the 2007 1st*

International Symposium on Information Technologies and Applications in Education, ISITAE 2007, 572–574. https://doi.org/10.1109/ISITAE.2007.4409351

Liaw, A., & Wiener, M. (2002). Classification and Regression by randomForest. In *R news* (Vol. 2, pp. 18–22). https://doi.org/10.1177/154405910408300516

Lim, C., Kim, K. J., & Maglio, P. P. (2018). Smart cities with big data: Reference models, challenges, and considerations. *Cities*, *82*, 86–99. https://doi.org/10.1016/j.cities.2018.04.011

Liu, X., Iftikhar, N., & Xie, X. (2014). Survey of real-time processing systems for big data. In *Proceedings of the 18th International Database Engineering & Applications Symposium on - IDEAS '14*. https://doi.org/10.1145/2628194.2628251

Liu, Z., Jiang, B., & Heer, J. (2013). ImMens: Real-time visual querying of big data. In *Computer Graphics Forum* (Vol. 32, pp. 421–430). https://doi.org/10.1111/cgf.12129

Lytvynenko, T. I. (2016). Problem of data analysis and forecasting using decision trees method. In *CEUR Workshop Proceedings* (Vol. 1631, pp. 220–226).

Maarala, A. I., Rautiainen, M., Salmi, M., Pirttikangas, S., & Riekki, J. (2015). Low latency analytics for streaming traffic data with Apache Spark. In *Proceedings - 2015 IEEE International Conference on Big Data, IEEE Big Data 2015* (pp. 2855–2858). https://doi.org/10.1109/BigData.2015.7364101

Madera, C., & Laurent, A. (2016). The next information architecture evolution: The data lake wave. In *8th International Conference on Management of Digital EcoSystems, MEDES 2016* (pp. 174–180). https://doi.org/10.1145/3012071.3012077

Marron, J. S. (2017). Big Data in context and robustness against heterogeneity. *Econometrics and Statistics*, *2*, 73–80. https://doi.org/10.1016/j.ecosta.2016.06.001

Mehmood, H., Gilman, E., Cortes, M., Kostakos, P., Byrne, A., Valta, K., … Riekki, J. (2019). Implementing big data lake for heterogeneous data sources.

Meng, X., Bradley, J., Street, S., Francisco, S., Sparks, E., Berkeley, U. C., … Hall, S. (2016). MLlib : Machine Learning in Apache Spark. *Journal of Machine Learning Research 17*, *17*, 1–7. https://doi.org/10.1145/2882903.2912565

O'Leary, D. E. (2014). Embedding AI and crowdsourcing in the big data lake. In *IEEE Intelligent Systems* (Vol. 29, pp. 70–73). IEEE. https://doi.org/10.1109/MIS.2014.82

Persson, C., Bacher, P., Shiga, T., & Madsen, H. (2017). Multi-site solar power forecasting using gradient boosted regression trees. *Solar Energy*, *150*, 423–436. https://doi.org/10.1016/j.solener.2017.04.066

Philip Chen, C. L., & Zhang, C. Y. (2014). Data-intensive applications, challenges, techniques and technologies: A survey on Big Data. *Information Sciences*, *275*, 314–347. https://doi.org/10.1016/j.ins.2014.01.015

Praveena, M., & Jaiganesh, V. (2017). A Literature Review on Supervised Machine Learning Algorithms and Boosting Process. *International Journal of Computer Applications*, *169*(8), 32–35. https://doi.org/10.5120/ijca2017914816

Price, S., & Flach, P. A. (2013). A Higher-Order Data Flow Model for Heterogeneous Big Data (pp. 569–574).

Ranjan, R. (2014). Streaming Big Data Processing in Datacenter Clouds. *IEEE Cloud Computing*, *1*(1), 78–83. https://doi.org/10.1109/MCC.2014.22

Roche, B. (2018). Cork city businesses face extinction if parking not solved, traders warn. Retrieved from https://www.irishtimes.com/news/ireland/irish-news/cork-city-businesses-face-extinction-if-parking-not-solved-traders-warn-1.3455437

Salloum, S., Dautov, R., Chen, X., Peng, P. X., & Huang, J. Z. (2016). Big data analytics on Apache Spark. *International Journal of Data Science and Analytics*, *1*(3–4), 145–164. https://doi.org/10.1007/s41060-016-0027-9

Sanchez, L., Muñoz, L., Galache, J. A., Sotres, P., Santana, J. R., Gutierrez, V., … Pfisterer, D. (2014). SmartSantander: IoT experimentation over a smart city testbed. *Computer Networks*, *61*, 217–238. https://doi.org/10.1016/j.bjp.2013.12.020

Sangani, D., Erickson, K., & Al Hasan, M. (2017). Predicting Zillow Estimation Error Using Linear Regression and Gradient Boosting. In *Proceedings - 14th IEEE International Conference on Mobile Ad Hoc and Sensor Systems, MASS 2017* (pp. 530–534). https://doi.org/10.1109/MASS.2017.88

Santana, E. F. Z., Chaves, A. P., Gerosa, M. A., Kon, F., & Milojicic, D. S. (2017). Software Platforms for Smart Cities. In *ACM Computing Surveys* (Vol. 50, pp. 1–37). https://doi.org/10.1145/3124391

Shin, J. H., Jun, H. B., & Kim, J. G. (2018). Dynamic control of intelligent parking guidance using neural network predictive control. *Computers and Industrial Engineering*, *120*(April), 15–30. https://doi.org/10.1016/j.cie.2018.04.023

Si, S., Zhang, H., Keerthi, S. S., Mahajan, D., Dhillon, I. S., & Hsieh, C.-J. (2017).

Gradient Boosted Decision Trees for High Dimensional Sparse Output. In *Proceedings of the 34 th International Conference on Machine Learning, Sydney, Australia, PMLR 70, 2017* (Vol. 70, pp. 3182–3190). Retrieved from http://proceedings.mlr.press/v70/si17a.html

Sirisha, N., & Kiran, K. V.D. (2018). Stock exchange analysis using Hadoop user experience (Hue). In *Proceedings of the International Conference on Intelligent Sustainable Systems, ICISS 2017* (pp. 1141–1144). IEEE. https://doi.org/10.1109/ISS1.2017.8389363

Sivarajah, U., Kamal, M. M., Irani, Z., & Weerakkody, V. (2017). Critical analysis of Big Data challenges and analytical methods. *Journal of Business Research*, *70*, 263–286. https://doi.org/10.1016/j.jbusres.2016.08.001

Sonnenberg, C., & vom Brocke, J. (2012). Reconsidering the Build-Evaluate Pattern in Design Science Research (pp. 381–397).

Swalin, A. (2018). Choosing the Right Metric for Evaluating Machine Learning Models — Part 2. Retrieved March 10, 2019, from medium.com/usf-msds/choosing-the-right-metric-for-evaluating-machine-learning-models-part-2-86d5649a5428

Tamminen, S., Laurinen, P., & Röning, J. (1996). Comparing Regression Trees With Neural Networks in Aerobic Fitness Approximation.

THEIN, K. M. M. (2014). Apache Kafka: Next Generation Distributed Messaging System, *03*(47), 9478–9483. https://doi.org/10.1007/s11606-014-2920-z

Xiao, J., Lou, Y., & Frisby, J. (2018). How likely am I to find parking? – A practical model-based framework for predicting parking availability. *Transportation Research Part B: Methodological*, *112*, 19–39. https://doi.org/10.1016/j.trb.2018.04.001

Yur 'evich Gorodov, E., & Vasil 'evich Gubarev, V. (2013). Analytical Review of Data Visualization Methods in Application to Big Data. *Journal of Electrical and Computer Engineering*, *969458*(7). https://doi.org/10.1155/2013/969458

Zhang, Y., Cao, T., Li, S., Tian, X., Yuan, L., Jia, H., & Vasilakos, A. V. (2016). Parallel Processing Systems for Big Data: A Survey. *Proceedings of the IEEE*, *104*(11), 2114–2136. https://doi.org/10.1109/JPROC.2016.2591592

Zhao, X., Garg, S., Queiroz, C., & Buyya, R. (2017). *A Taxonomy and Survey of Stream Processing Systems*. *Software Architecture for Big Data and the Cloud* (1st ed.). Elsevier Inc. https://doi.org/10.1016/B978-0-12-805467-3.00011-9

Zheng, Y., Rajasegarar, S., & Leckie, C. (2015). Parking availability prediction for sensor-enabled car parks in smart cities. *2015 IEEE 10th International Conference on Intelligent Sensors, Sensor Networks and Information Processing, ISSNIP 2015*, (April), 7–9. https://doi.org/10.1109/ISSNIP.2015.7106902