Prediction of parking areas availability from parking dataset using AI/ML models

Dabir Hasan Rizvi (dhr8@aber.ac.uk)

Department of Computer Science
Aberystwyth University
Wales

September
2023

This thesis is submitted in partial fulfilment of the
requirements for the degree of Master of Science

Degree: MSc Advanced Computer Science (with Integrated Year in Industry)
Module: CHM9360
Supervisor: Dr Yasir Saleem Shaikh

ABSTRACT

Traffic congestion has caused frustration among drivers. Finding a free space to park has become challenging. This paper uses AL/ML methods to predict the space availability for a time series model. It includes the context, background, scope and its limitations. Data collection, pre-processing and data transformation. The literature review section contains the previous work done in this field to make predictions for parking area availability and their benefits, gaps and the outcomes of their research. This model utilises a parking dataset in Santander, Spain, which contains data from nearly 400 on-street parking sensors collected over nine months [1] to make predictions. The project aims to use AI/ML techniques to predict the parking spots by clustering them into areas based on the coordinates provided, thus helping the users to make informative decisions regarding which area and parking spots to use. The dataset includes information on spots available whether they are occupied or not and each parking spot's time frame of historical parking details. The methodology section covers which AI/ML models should be selected, and how they can be developed and trained. The experimental results section displays the outcomes for machine learning models like Long Short-Term Memory and Random Forest using graphs and statistics. Finally, a summary of the future scope of the project and its shortcomings is mentioned in the conclusion section.

Declaration of originality

I confirm that:

- This submission is my own work, except where clearly indicated.
- I understand that there are severe penalties for Unacceptable Academic Practice, which can lead to loss of marks or even the withholding of a degree.
- I have read the regulations on Unacceptable Academic Practice from the University's Academic Quality and Records Office (AQRO) and the relevant sections of the current Student Handbook of the Department of Computer Science.
- In submitting this work, I understand and agree to abide by the University's regulations governing these issues.

Name: Dabir Hasan Rizvi
Date: 29/09/2023

Consent to share this work

- By including my name below, I hereby agree to this thesis being made available to other students and academic staff of the Department of Computer Science, Aberystwyth University.

Name: Dabir Hasan Rizvi
Date: 29/09/2023

ACKNOWLEDGEMENTS

A special thanks to my supervisor, Dr Yasir Saleem Shaikh, for his continuous help in this journey. By arranging meetings weekly and guiding me through the whole process, taking time outside the meeting hours to explain the problems in detail and motivating me to achieve the best. I would also like to thank Dr. Faisal Rezwan for his continuous support and for providing updates throughout this semester. Thanks to Dr. Edel Sharrat and R.T. Barry for making available the LATEX document template and example pdf.

It has been a privilege to spend my last two years in such a great university. I would also like to thank all the staff who continuously guided me throughout my time at the university.

Thank You
- Dabir Hasan Rizvi

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

Finding a parking spot is a difficult task for a driver today whether we're running late for work or going shopping. The availability of parking spaces depends on various internal and external factors, leading to various environmental issues and increasing frustration among drivers. The need for a model to predict parking area availability is growing enormously. Metropolitan cities like New York or London are facing these problems as well, and even smaller towns like Aberystwyth aren't exempt from this global problem, so finding a parking spot is getting increasingly difficult. The frustration of going around the same place multiple times is the greatest sentiment shared by drivers worldwide. Drivers searching for parking spaces in urban cities cause 30% of traffic congestion [2].

The population growth has a projection of a 12% increase by 2050 in urban areas, creating a need for a solution to this problem before it gets out of hand [3]. Waerden. et al. [4] suggested the importance of parking information needed for drivers to utilise the spaces efficiently by displaying real-time data directly to the user with the help of a parking guidance system. These systems today are available in closed parking spaces, and expanding the horizons is vital before things get out of hand. This problem also causes accidents when the driver is mainly focused on finding a parking space and contributes to additional fuel consumption and pollution of the environment [5]. Additional work has been done to manage the parking spaces by installing sensors which provide information on parking area availability to the driver and getting feedback from local people who know the parking space availability, [6] which is not an efficient way to predict an available space creating a need for a model which can accurately calculate the needs of a user and provide better interpretability.

With the technology growing, there is a hope to eradicate these issues where the drivers can save time, which is precisely the aim of this paper, focusing mainly on the Santander,

located in the coastal region of Spain, where 400 on-street parking sensors were installed for nine months providing detailed information which I intend to use to design an AI/ML model to predict the availability based on this time series model.

1.2 Problem Statement

The main challenge of this paper was to work with the information provided, cleaning and pre-processing the data efficiently to solve the everlasting problem of finding a parking space in the busy streets of Santander, Spain. This paper goes through the previous work done in this field to improve the existing knowledge by designing an AI/ML model to achieve high accuracy from its predecessors, taking on a journey to research different Machine Learning models, and finding the best fit for the problem. This paper intends to answer the research questions and provide satisfactory results to predict parking availability, helping reduce frustrations among drivers and help the environment by reducing fuel consumption.

1.3 Research Questions:

- Can a machine learning model predict parking area availability based on the historical data in Santander, Spain?
- What are the benefits of modelling a machine learning model to aid the drivers in Santander, Spain?
- How to clean and design the data for a machine learning model to predict parking space availability?
- Which machine learning model to use to solve the parking area prediction problem?
- What are the technical challenges of designing such models?
- What is the importance of feature engineering in such models?

1.4 Significance of the study

Parking area availability is a real-life problem which is getting frustrating for drivers every day. My decision to choose this topic was to work on a real-world problem to help contribute to the field and learn along the way. Making the world a better place to live is the goal of all humans. There is no better way than to help the environment and increase convenience, and I aim to do that by reducing the consumption of fuel and helping to save time by designing a prediction model to solve the parking problem. Moreover, this paper intends to contribute to the continuously growing field of Smart cities.

1.5 Approach and Overview

In this paper, I first did a literature survey on existing work on this topic, which intends to go in-depth into a few approaches that contributed to this field and its learning outcomes. I went through different AI/ML techniques used to solve parking availability problems with their benefits and shortcomings, helping me with my model selection. The different models I looked into were:

- Time series forecasting model using ARIMA, SARIMA and neural network.
- Deep learning-based model using Recurrent Neural Network.
- Regression-based model working with PGI system (Parking Guidance and Information)
- Reinforcement learning model to solve complex sequential decision making.
- A Graph-to-sequence model, which also worked on a different dataet from Sanatander, Spain.
- Other models include Random Forest, Decision Trees, K-nearest neighbours, prophet etc.

Going in-depth into each section, I intended to do a deep literature survey on this topic. I explored different model options and explained my model selection based on the literature survey and other justifications mentioned in the methodology section. I designed my Machine Learning model using Long Short-Term Memory because of its ability to handle sequentially aligned information efficiently for the time series dataset and Random Forest because of its interpretability and resilience to unreliable data. I then explained the model's development and training methodology and provided experimental results to justify my choice. Even though the paper holds great potential, it is crucial to acknowledge the limitations and scope of the project explained in the conclusion section.

Chapter 2

Literature Review

2.1 Introduction

The necessity for an effective management system for parking has been a critical concern in the era of rapid urbanisation and an increase in vehicle ownership. Integration of Artificial Intelligence and Machine Learning (ML) techniques offers data-driven insights to help accurately estimate the availability of parking spots, reducing traffic congestion and saving time whilst maintaining efficient resource allocation. Having a concern with parking accessibility can be highly frustrating for drivers. An efficient model that anticipates the availability of parking spots in each area can provide a smooth integration to smart-city architecture and help society by controlling urban traffic density and carbon emissions in the environment. This literature survey summarises different parking prediction models used and their results.

2.2 Parking Space Prediction Models

2.2.1 Time Series Forecasting Model

Sağlam et al. [7], examined the significance of offering real-time information to drivers about the parking area availability closer to their destination to lower traffic congestion, aiding with a solution for urban traffic congestion. They proposed three different approaches to estimate the availability of parking spaces: (i) auto-regressive integrated moving average (ARIMA) model, (ii) seasonal auto-regressive integrated moving average (SARIMA) model, and (iii) neural networks. (2). Working with the "SFpark" dataset from San Francisco, their approach covered external factors such as the significance of the time of the day and calculating their influence on the occupancy rates. In this section, I will mainly focus on the ARIMA model approach as it works efficiently with the time-series dataset.

They examined the mean occupancy rates for different areas and time intervals. The initial discovery was to evaluate peak parking hours and their patterns. Upon discovery, lower occupancy rates were observed, during the night. Selecting 12 parking spaces for their model, they thoroughly investigated the dataset. Avoiding trends and seasonal effects and accounting for historical data, an ARIMA model was employed integrating autoregression with moving average. The NARX network provides a connection for monitoring and accounts for historical data using a series-parallel architecture.

In this section, we will discuss ARIMA findings. As shown in Figure 2.1, the researchers considered the construction of the model by mixing one and two values of p,d, q. The rows in the table depict each model structure whereas the columns visualise the parking spaces. The minimum, maximum, and average performance of each model is shown in the last three columns.

Model	P01	P02	P03	P04	P05	P06	P07	P08	P09	P10	P11	P12	min	max	avg
(1,1,1)	0.026	0.253	0.048	0.007	0.016	0.092	0.090	0.085	0.045	0.145	0.132	0.143	0.007	0.253	0.090
(1,2,1)	0.395	2.574	0.153	0.051	0.577	1.587	2.525	4.396	0.113	0.114	6.323	0.810	0.051	6.323	1.635
(1,1,2)	0.025	0.168	0.049	0.007	0.016	0.096	0.078	0.113	0.045	0.146	0.124	0.141	0.007	0.168	0.084
(1,2,2)	0.468	4.059	0.222	*n-inv*	0.585	0.135	2.317	0.390	0.531	0.528	6.565	0.066	0.066	6.565	1.442
(2,1,1)	0.020	0.111	0.049	0.007	0.019	0.099	0.068	0.098	0.043	0.149	0.129	0.104	0.007	0.149	0.075
(2,2,1)	0.703	2.323	0.168	0.049	0.603	0.859	0.393	3.593	1.833	0.135	4.011	0.633	0.049	4.011	1.275
(2,1,2)	0.015	*n-inv*	0.048	*n-inv*	0.019	0.123	0.067	0.068	0.044	0.123	0.121	0.094	0.015	0.123	0.072
(2,2,2)	0.711	1.928	0.091	0.060	0.030	1.180	0.353	0.323	1.278	0.370	1.997	0.103	0.030	1.997	0.702
min	0.015	0.111	0.048	0.007	0.016	0.092	0.067	0.068	0.043	0.114	0.121	0.066			
max	0.711	4.059	0.222	0.060	0.603	1.587	2.525	4.396	1.833	0.528	6.565	0.810			
avg	0.295	1.631	0.104	0.030	0.233	0.521	0.736	1.133	0.491	0.214	2.425	0.262			

Figure 2.1: Results obtained with ARIMA model [7]

The researchers observed better performance parameters when the occupancy rate pattern did not contain significant fluctuations for parking spaces (example: P01), as seen in the Figure above. In contrast, when the occupancy rates differ significantly between different times of the day, a large fluctuation is observed, which causes the performance values at extreme values (example: P11). Green and orange highlighted text represent the minimum and maximum values observed.

Overall, the researchers combined ARIMA and SARIMA models incorporating seasonal effects with neural networks, specifically to analyse time series techniques. They predicted parking occupancy rates to provide a comprehensive approach to solving the complexity of a parking prediction model. The goal was to compare different models and understand the patterns and benefits of employing one model over the other for the time series forecasting model. This model gives us a good indication of benefits of ARIMA for time series dataset, making it a strong candidate for selection.

2.2.2 Deep Learning Model

Deep Learning is a subset of Machine Learning and can learn from complex datasets. The models are designed to find complex patterns and relationships in datasets. They resemble a human brain neural network. The LSTM (Long Short-Term Memory) is a specific type of deep learning model frequently employed as a solution for time series prediction tasks. LSTMs are a subclass of RNN (Recurrent Neural Network) that work exceptionally well at modelling sequential data.

With an emphasis on the LSTM model, Shao et al. [8] proposed a unique framework built around RNN (Recurrent Neural Network). This model predicts parking availability, which is the fundamental idea utilising two essential performance indicators: the likelihood of a car leaving a parking space and the occupancy rate of on-street parking spots in a given area. The main contributions of this literature were to analyse the parking duration and estimation of parking occupancy using LSTM and evaluating the performance of these models. Figure 2.2 represents the proposed architecture for their model.

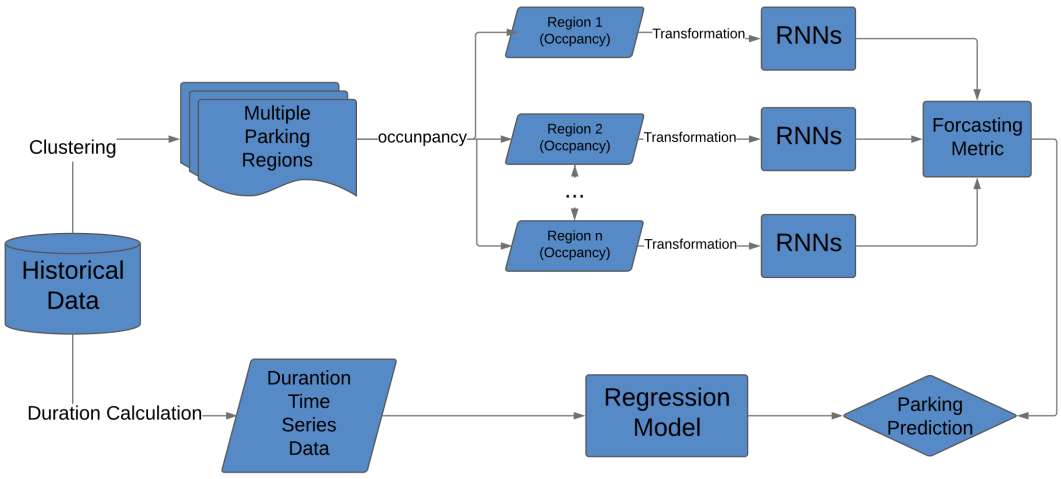


Figure 2.2: Functional architecture of the proposed parking prediction scheme [8]

The proposed framework estimated parking spot predictions using LSTM and clustering techniques. The framework consisted of two modules to predict parking occupancy duration time, where they introduced a Recurrent Neural Network (LSTM) to contemplate patterns for the occupancy rate in each region in Melbourne clustered by K-means. They observed better performance when the division of regions was clustered. Their future work was to improve the model by researching different deep learning models and comparing their performance to achieve better results. Making this model an ideal choice for model selection when the data is clustered time series.

2.2.3 Regression Based Model

The aspiration for drivers navigating urban environments is to acquire knowledge of parking availability around them. Fabian Bock and Sergio DI Martino et al. [9] aimed to a discerning parking area prediction within the PGI system (Parking Guidance and Information), where they presented a novel approach. Their goal was to increase the efficiency and accuracy of addressing the parking issues in urban areas by proposing a two-step approach to this problem:

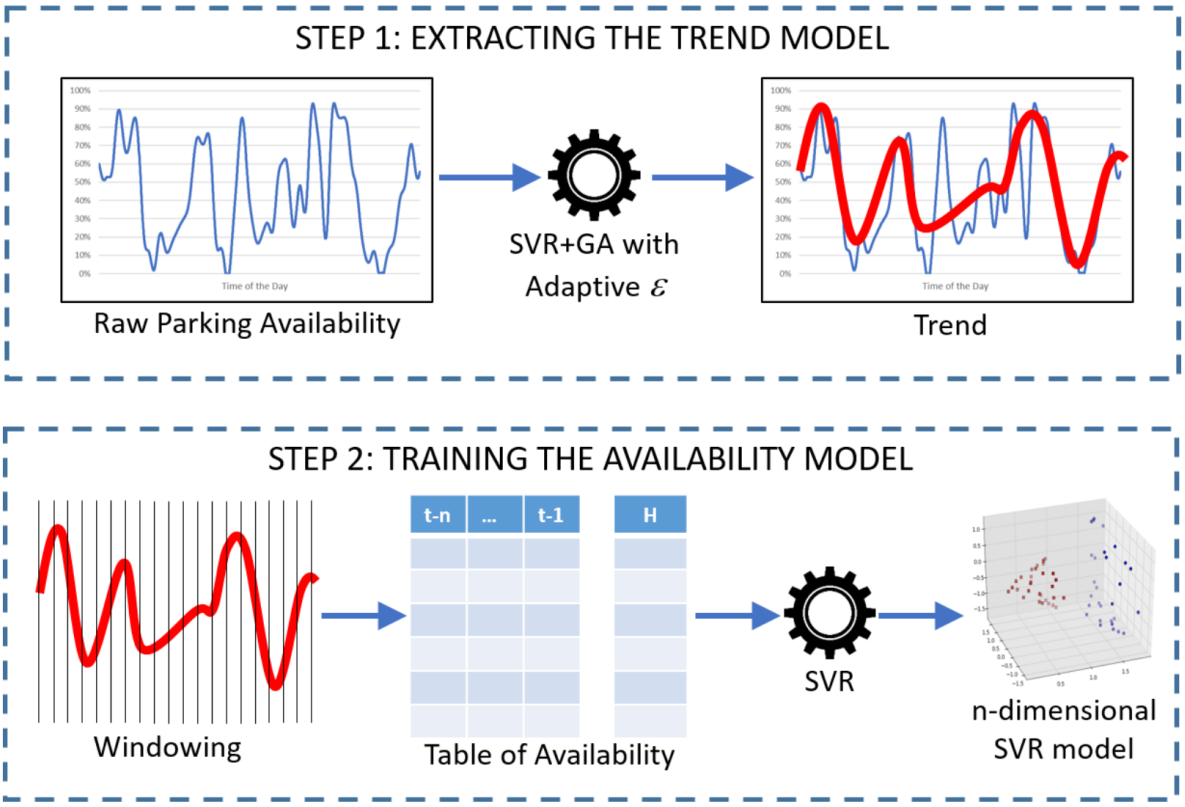
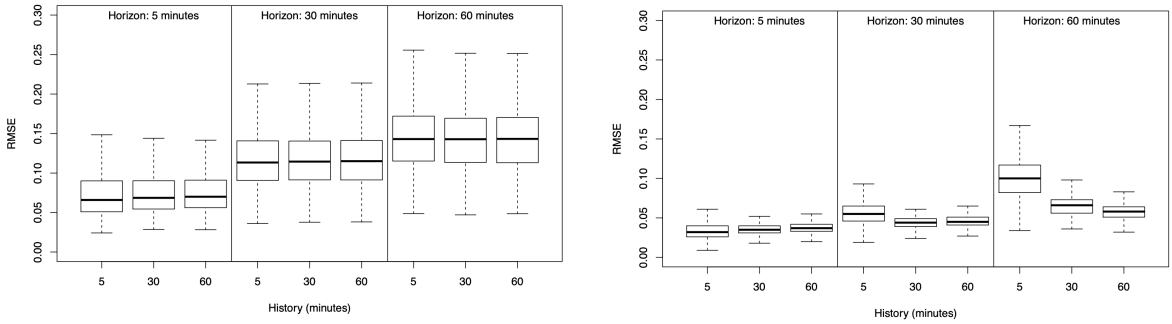


Figure 2.3: Graphical representation of the proposed 2-step approach. [9]

- Trend Extraction: a Support Vector Regression model was used to find hidden patterns from records containing noise. Employing a Genetic Algorithm to continuously find the fittest SVR parameters based on the parking areas availability for every road segment. Helping to reduce the noise from the data and underlying changes in the data. Which increases the efficiency of the model, resulting in smoother trend curves.
- Training the Prediction Model: After obtaining the desired result from trend extraction, another SVR model was trained to predict the availability of parking spaces for different time horizons.

From the dataset SFpark project located in San Franscico, this model was applied comparing their technique with baseline approaches to directly use the raw data to train models. These tests were conducted based on different prediction horizons of 5, 30 and 60 minutes as we can see in the Figure below:



(a) Performance of the trained model using the raw data

(b) Performance of the trained model using the extracted trends.

Figure 2.4: Comparison of model performance using different data [9]

The result demonstrated their approach, which consistently outperformed the baseline models and allowed the model to reduce the size of the dataset by approximately 60%. The focus of this paper was to reduce traffic congestion by proposing a 2-step solution utilising SVR (Support Vector Regression) and extraction of trends. They achieved high accuracy and a size reduction of the data, which increased the running speeds and efficiency of the proposed model. Their shortcomings and future research directions were to focus on employing clustering techniques and research more deep learning models to complement their results and achieve higher performance.

2.2.4 Reinforcement Learning Model

Reinforcement learning is a powerful tool in machine learning, showcasing solutions for complex sequential decision-making problems by having the ability to learn by trial and error using feedback from its actions. In this section, I am discussing more than parking area availability but also about utilising automated vehicles to detect the parking spots to make decisions based on the environmental conditions and solve the issue of parallelism in parking. The overview of the reinforcement learning model proposed by Zhang et al. [10] is shown in Figure 2.5, which contains two modules to track the parking slot for an electric automatic vehicle using reinforcement learning. The parking slot tracking provides a continuous estimation of the position of parking spots for reinforcement learning, which adapts to achieve end-to-end planning by utilising details of steering wheel angles and parking availability.

The general path planning is not efficient for perpendicular parking slots due to its narrow width. An end-to-end reinforcement-learning-based algorithm was proposed by Zhang et

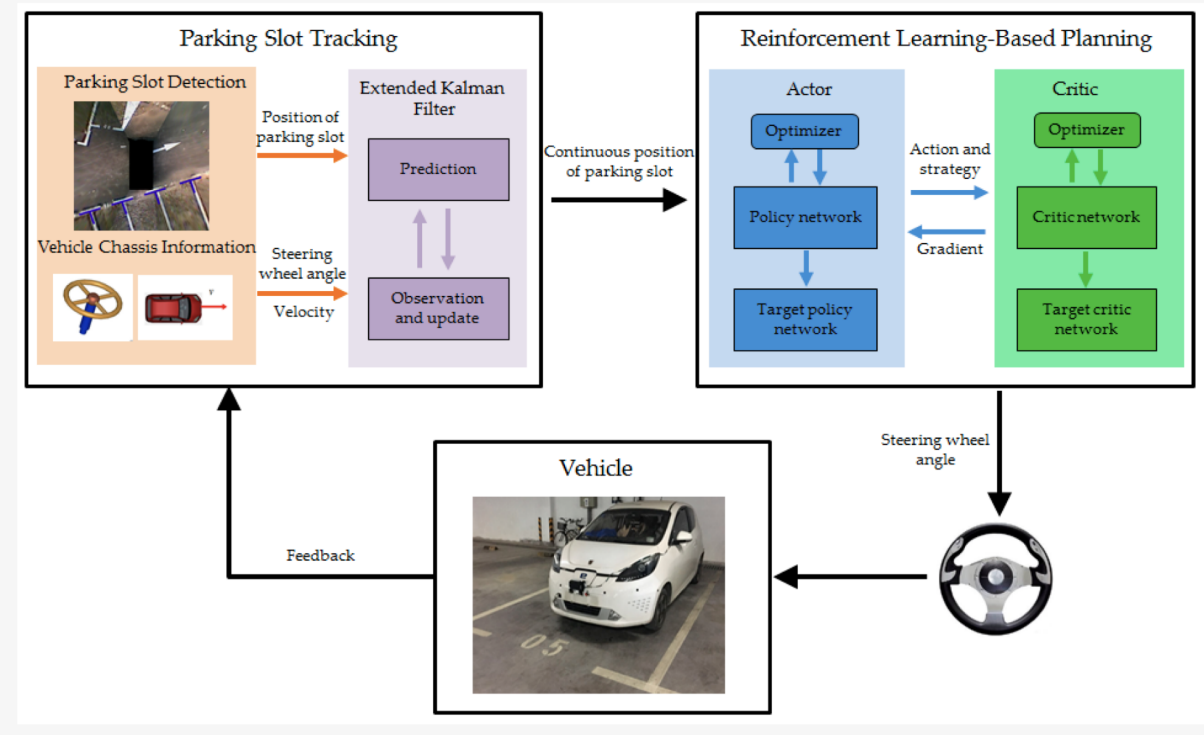


Figure 2.5: Overview of the reinforcement learning-based end-to-end parking method. [10]

al. [10], and the main contribution was to apply Deep Deterministic Policy Gradients (DDGP) to the model to continuously learn from the experience that it accumulates. To achieve regressive training of the model a parking slot tracking algorithm was installed in vehicles utilising the Kalman Filter (EKF) by creating patterns of relationship to fuse vehicle chassis information to track parking spots. The training process of the DDGP algorithm involved simulation of the environment using tools like MATLAB and PreScan. After achieving satisfactory results from simulation, the next step involved testing in a real-world environment and addressing challenges like convergence issues to improve training measures. Real-time testing was applied to the Rongwei E50 electric vehicle equipped with cameras and sensors. A comparison of 3 parking methods was done:

- Geometric method-based path planning with PID-based path tracking.
- Geometric method-based path planning with SMC-based path tracking.
- Reinforcement learning-based end-to-end parking.

Reinforcement learning outperformed and predicted the path and detected the available parking space. The utilisation of other parking spots coordinates was crucial in deciding on the area of availability.

2.2.5 Graph-to-Sequence Model

A paper proposed by Sasaki et al. [11] worked on the older dataset in Santander to predict the parking area availability for the smart city project in Santander, the same source for my dataset. Several IOT (Internet of Things) sensors, including parking sensors, were installed to retrieve real-time data. The smart city project was initiated in Santander to reduce traffic congestion and save time for urban drivers and tourists. This model is meant to solve the parking availability problem in real time, considering disruptions or any other service interruptions, and a graph-to-sequence model is used to calculate the prediction. Similar to my project goals, they tend to provide predictions for time steps. The sensors calculate the magnetic field, sending data to relay nodes and to the servers where the data was collected.
The proposed model was trained for graph-to-sequence neural network model by clustering parking spots into parking areas and minimising mean absolute loss function (MAE), the function is defined as:

$$Loss = \frac{1}{N' \cdot |s|} \sum_{i=1}^{N'} \sum_{j=1}^{|s|} abs\left(s'_{i,j} - s_{i,j}\right) \tag{2.1}$$

where, $|s|$ denotes the size of vector s (i.e., 27), and $s'_{i,j}$ and $s_{i,j}$ denote the predicted and measured values of cluster j at time step i, respectively. N' is a predefined parameter for model training, and we set N' to a random value. To reduce overfitting, we apply dropout with a probability of 0.3 to the encoding. [11]

The model consisted of a Graph Neural Network (GNN) and a Recurrent Neural Network (RNN) decoder as seen in Figure 2.6 below.

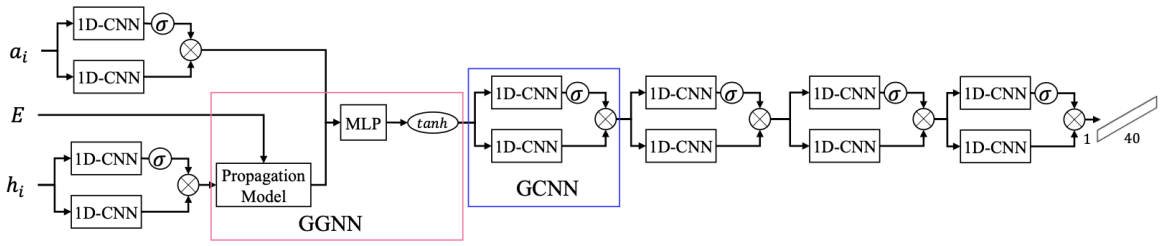


Figure 2.6: GNN encoder model. Our GNN encoder consists of one GGNN and six GCNN layers. [11]

The GNN encoder maps the inputs to a vector of defined dimensionality, whereas an RNN decoder is used to decode the output from a GNN to sequentially generate parking area predictions. Three models were tested as we can see in the Figure provided below: The mean absolute loss function was calculated for parking lots using a Graph Neural Network (GNN), Convolution Neural Network (CNN) and a Recurrent Neural Network

Cluster ID	RNN				CNN				GNN			
	15 min	30 min	60 min	120 min	15 min	30 min	60 min	120 min	15 min	30 min	60 min	120 min
0 (10)	0.749	0.734	0.769	0.769	0.468	0.497	0.540	**0.606**	**0.293**	**0.392**	**0.502**	0.640
1 (15)	0.803	0.816	0.822	0.843	0.552	0.556	0.579	0.647	**0.301**	**0.367**	**0.472**	**0.576**
2 (5)	0.404	0.392	0.416	0.429	0.088	0.095	0.117	0.155	**0.069**	**0.081**	**0.110**	**0.139**
3 (13)	0.911	0.899	0.911	0.904	0.669	0.717	0.773	0.844	**0.243**	**0.319**	**0.444**	**0.606**
4 (6)	0.265	0.291	0.302	0.301	0.126	0.148	0.175	0.231	**0.104**	**0.123**	**0.171**	**0.223**
5 (6)	1.015	0.978	0.968	0.975	0.513	0.530	0.587	0.632	**0.247**	**0.337**	**0.448**	**0.591**
6 (14)	0.806	0.786	0.791	0.803	0.579	0.645	0.675	0.806	**0.284**	**0.379**	**0.495**	**0.673**
7 (15)	0.913	0.936	0.966	1.042	0.510	0.561	0.599	0.738	**0.241**	**0.342**	**0.465**	**0.629**
8 (6)	0.631	0.703	0.765	0.847	0.415	0.539	0.640	0.769	**0.243**	**0.368**	**0.524**	**0.673**
9 (20)	1.550	1.585	1.611	1.666	0.589	0.653	0.715	0.817	**0.244**	**0.358**	**0.521**	**0.696**
10 (12)	0.560	0.543	0.549	0.540	0.279	0.288	0.342	0.449	**0.131**	**0.184**	**0.244**	**0.327**
11 (6)	0.422	0.405	0.401	0.417	0.240	0.261	0.281	0.321	**0.088**	**0.121**	**0.168**	**0.251**
12 (10)	0.982	0.974	0.953	0.966	0.722	0.733	0.756	0.808	**0.301**	**0.400**	**0.539**	**0.699**
13 (5)	0.804	0.846	0.874	0.948	0.178	0.204	**0.256**	0.371	**0.127**	**0.179**	0.265	**0.370**
14 (18)	1.976	2.214	2.331	2.372	0.669	0.708	0.894	1.115	**0.248**	**0.361**	**0.554**	**0.806**
15 (10)	0.744	0.838	1.011	1.175	0.245	0.278	0.363	0.457	**0.131**	**0.170**	**0.252**	**0.339**
16 (10)	0.806	0.886	0.912	1.001	0.314	0.331	0.399	0.490	**0.126**	**0.187**	**0.275**	**0.402**
17 (11)	0.819	0.821	0.839	0.860	0.502	0.528	0.594	0.711	**0.229**	**0.309**	**0.399**	**0.586**
18 (16)	1.203	1.197	1.193	1.167	0.701	0.725	0.758	0.821	**0.297**	**0.391**	**0.505**	**0.658**
19 (29)	1.276	1.317	1.397	1.494	0.940	0.940	1.053	**1.221**	**0.759**	**0.890**	**1.038**	1.238
20 (7)	0.581	0.623	0.787	0.892	0.157	0.182	0.278	0.389	**0.066**	**0.108**	**0.212**	**0.351**
21 (12)	0.796	0.824	0.834	0.850	0.524	0.526	0.593	0.680	**0.185**	**0.259**	**0.367**	**0.508**
22 (10)	0.882	0.929	0.980	1.108	0.835	0.818	**0.929**	**1.088**	**0.580**	**0.737**	0.944	1.211
23 (10)	1.510	1.444	1.226	1.154	0.395	0.448	0.547	0.650	**0.191**	**0.295**	**0.429**	**0.572**
24 (18)	1.352	1.165	1.004	1.036	0.572	0.653	0.736	0.907	**0.403**	**0.546**	**0.694**	**0.879**
25 (17)	3.172	3.200	3.145	3.120	0.859	0.889	1.107	1.504	**0.346**	**0.509**	**0.816**	**1.258**
26 (12)	0.925	0.956	0.985	1.057	0.567	0.628	**0.702**	**0.833**	**0.417**	**0.549**	0.707	0.929

Figure 2.7: MAE of available parking lots per street. The numbers within parentheses at Cluster ID denote the numbers of parking lots in clusters. The bold font indicates the best accuracy. [11]

(RNN). The table provides the results of these operations and generally GNN encoder has the best performance. The findings in this paper showed that GNN and RNN encoders are more robust to changes in the training size of the data. This paper was proposed by Sasaki et al. [11] concludes that the GNN model accurately predicted the parking area availability in Santander, Spain. Future Work such as online learning methods, user feedback and the addition of features such as weather conditions and traffic volumes was proposed by the authors.

2.2.6 Other Parking Space Prediction Models

Jelen et al. [12] proposed the solution of parking area availability and the impact on traffic congestion and greenhouse gas emissions. Evaluation of prediction metrics was done in this paper aiming to improve parking spot detection utilising different machine learning models such as CatBoost, and Random Forest. They utilised parking sensor data to evaluate a result indicating the performance metrics of the machine learning model which outperformed the baseline model, the R-square score was achieved in the range from 84.31% to 88.83% indicating the goodness of fit of the model. Whereas the CatBoost

model using contextually enriched data has the highest prediction metric surpassing the Random Forest model by 1.7% which incorporates the importance of contextual data for efficient results.

Integrating external factors to model a prediction of parking area availability was performed by Inam et al. [13]. The model analysed the impact of pedestrian activity, weather conditions and traffic towards the parking prediction. A comprehensive analysis of ML and deep learning technology was constructed with Random Forest emerging as the top performer, achieving an average accuracy of 81% and UC of 0.18. Decision Trees and K-Nearest neighbours although being fewer complex models, outperformed the majority of the complex models. Indicating the importance of contextual data in the prediction accuracy and providing a fast and reliable approach to the parking problems.

Kuhail et al. [14] took on the challenge of predicting the availability of parking spaces in densely populated areas such as the United States. According to his research findings, approximately 12 minutes are generally wasted per day by an average urban driver in the United States to find a parking area. Working with user-generated data and sensor-based data were used in this approach. However, the limitations of this project included user participation to provide data and the infrastructure cost to set the sensors to address these issues, they predicted a forecasting model by utilising historical parking data from Kansas City and Melbourne, analysis of car counts in every five-minute interval over a period of a year. ARIMA, LSTM and Prophet were employed for the time series forecasting model by featuring external variables such as weather and finding patterns in the dataset to achieve a result with an absolute error of 1.6 vehicles for Melbourne and 0.78 vehicles for Kansas City when using LSTM architecture for a week of parking prediction.

Predicting parking availability has become an important research field in machine learning aiming to eradicate traffic congestion and reduce environmental impact. The challenges generally faced in these types of predictions generally include data privacy, scalability, and interpretation of the models. The techniques that can be used in the future to solve these problems are:

- Addressing data privacy concerns and implementing robust data anonymisation.
- Improving data quality and availability.
- Building easily scalable prediction models to adapt seamlessly with large-scale datasets.
- To increase the interpretability of models that offer high accuracy.

This literature survey aided me in making my decision for my model selection. I have seen various models used on time series forecasting model where Random Forest and

Long Short-Term Memory stands out in most cases providing an efficient and scalable model.

Chapter 3

Data Preperation

Collecting and analysing information (or data) from numerous sources to address issues related to research problems, assessing results, offering solutions, and estimating trends in data collection [15]. Whereas data pre-processing is a modification of data into a specific format to process the data for mining, predicting models and various other data science operations more effectively and quickly [16]. The foundation elements to create an accurate and reliable prediction model using Machine Learning techniques for parking area availability is to apply data collection and pre-processing at the beginning of the development pipeline. In this section, I want to discuss the complex aspects of these procedures and their benefits.

3.1 Data Collection

The initial step to set up the project was to source the data. Dr Saleem Saleem Shaikh took the initiative to provide me with a dataset which he had worked with before and wanted me to expand its potential. The dataset contained information from around 400 on-street parking sensors collected over a 9-month period from October 2017 to May 2018. These IOT (Internet of Things) sensors calculate magnetic field, sending data to relay nodes and to the servers where the data was collected and retrieved as we can see in Figure 3.1 below [11].

Figure 3.1: Fig.1: Parking sensors in Santander. (top) locations of parking lots with cluster IDs, (bottom-left) a parking sensor with a car, (bottom-center) the parking sensor, and (bottom-right) a parking panel to guide drivers. [11]

According to Saleem et al. [17], the origin of the dataset was initiated while working on the H2020 WISE-IoT project [18], which was an H2020 EU-KR project. The sensors stored data in the NGSI (Next Generation Service Interface) context broker. Saleem et al. [17] designed an algorithm to retrieve and store the data at regular intervals to create consistency throughout the data. The dataset had 146,289 records, providing various information as we can see from Figure 3.2 below:

The total number of records in the given dataset is: 146289

Out[3]:

	row_id	parking_spot_id	start_time	end_time	status	duration	last_updation_time
0	1	urn:entity:santander:parking:parkingSpot:3601	2017-10-18 09:57:04	2017-10-18 16:12:45	occupied	22541	2017-10-18 16:12:45
1	2	urn:entity:santander:parking:parkingSpot:3602	2017-10-18 09:57:02	2017-10-18 16:00:30	occupied	21808	2017-10-18 16:00:30
2	3	urn:entity:santander:parking:parkingSpot:3603	2017-10-18 09:24:10	2017-10-18 12:06:10	free	9720	2017-10-18 12:06:10
3	4	urn:entity:santander:parking:parkingSpot:3604	2017-10-18 09:22:30	2017-10-18 13:17:46	occupied	14116	2017-10-18 13:17:46
4	5	urn:entity:santander:parking:parkingSpot:3605	2017-10-18 10:01:12	2017-10-18 13:17:46	free	11794	2017-10-18 13:17:46
...	...	...	...	...	...	...	...
146284	147123	urn:entity:santander:parking:parkingSpot:3621	2018-05-20 07:20:13	2018-05-20 07:20:13	free	0	2018-05-20 07:14:28
146285	147124	urn:entity:santander:parking:parkingSpot:3745	2018-05-20 07:20:13	2018-05-20 07:20:13	occupied	0	2018-05-20 07:15:15
146286	147125	urn:entity:santander:parking:parkingSpot:3710	2018-05-20 07:25:24	2018-05-20 07:25:24	occupied	0	2018-05-20 07:20:06
146287	147126	urn:entity:santander:parking:parkingSpot:3904	2018-05-20 07:44:40	2018-05-20 07:44:40	occupied	0	2018-05-20 07:38:54
146288	147127	urn:entity:santander:parking:parkingSpot:3630	2018-05-20 07:49:26	2018-05-20 07:49:26	free	0	2018-05-20 07:46:28

146289 rows × 7 columns

Figure 3.2: Original Dataset

The data provided the following information [19]:

- 'row_id': It is a unique identifier for each row in the dataset used for indexing and referencing purposes.
- 'parking_spot_id': A Uniform Resource Name (URN) format to specify a unique identifier for different parking spots in the Santander area represented in the range from 3600 – 3923.
- 'start_time' and 'end_time': These timestamps indicate a period in which the parking spot was occupied or free. "start_time" indicates the beginning of the parking event. Whereas the "end_time" determines the end of the parking event.
- 'status': The status column determined the status of the parking spot ID in each period, depicting whether the parking spot was "occupied" or "free" in each time slot.
- 'duration': It is a calculation parameter, calculating the difference in second(s) for each given event between the start of an event and the end of an event. In other words, the difference between start_time and end_time in seconds is calculated in the duration column, representing the length of time between these two intervals for a given status.
- 'last_updation_time': The timestamp demonstrates when the information about this dataset for a given row was last updated (or) changed.

3.2 Exploratory Data Analysis (EDA)

This process is carried out to analyse and investigate records and to summarise their key features. It gives a guideline to manipulate data sources to achieve desirable results. Increasing the chances to discover patterns, spot anomalies and test a hypothesis, exploratory data analysis is a crucial step to understanding the data.

The exploratory data analysis for time columns is presented in Table 3.1, in this table we present an overview of important statistical measurements for time-related columns providing an insight into the distribution and characteristics of these columns.

Table 3.1: Summary Statistics for Time Variables

	start_time	**end_time**	**last_updation_time**
count	146289	146289	146289
mean	2018-01-09 21:24:35.836713472	2018-01-10 12:45:43.852538368	2018-01-10 12:20:36.542337536
min	1970-01-01 01:00:08	1970-01-01 01:00:08	1970-01-01 01:00:08
25%	2017-11-25 04:02:40	2017-11-25 09:22:24	2017-11-25 08:50:25
50%	2017-12-31 06:52:34	2017-12-31 14:52:16	2017-12-31 14:43:05
75%	2018-02-12 13:31:17	2018-02-12 16:17:03	2018-02-12 16:11:28
max	2018-05-20 07:49:26	2018-05-20 07:49:26	2018-05-20 07:46:28

- Count: There are records of 146,289 data points indicating the size of the dataset.
- Mean – These values highlight the central tenancy for each time stamp. Revealing the average time for each time column.
- Minimum (Min): This section gave us an insight into the earliest records present in the dataset. Upon visualisation we realised the earliest timestamps backdate to January 1, 1970, at 01:00:08 AM, indicating a fault in the data since the data was recorded between October 2017 and May 2018. Helping us to improve the data and handling these incorrect values which was done during the data cleaning process.
- 25th Percentile (Q1): Indicating time stamps below which 25% of records are present. Providing us with an insight into the lower range of the dataset.
- 50th Percentile (Q2or Median): Indicates the middle point of the distribution.
- 75th Percentile (Q3): Indicating time stamps below which 75% of records are present. Providing us with an insight into the higher range of the dataset.
- Maximum (Max): Unlike the minimum corrupted minimum section, the maximum section indicated the latest recorded timestamp which is correct as the data was recorded until May 2018.

3.2.1 Data Visualisation

The process of presenting data and information in a visual format is called Data Visualisation. Graphs, charts, and images are used to visualise data. It is a crucial step to understand the dataset accurately and provides us with information regarding its modelling capabilities. The goal of this step is to locate, identify, format, and convey data in the best way feasible. Some benefits of visualising data include:

- Quick method to understand the problems to make faster decisions.
- Easier means of distributing information to a general audience.
- Handling outliers and missing data.
- Identifying Anomalies and Patterns within data. [20]

3.2.1.1 Visualising Parking Status:

The first step was to visualise the status column in the dataset to determine the consistency in the data. As seen in Figure 3.3, the count of 'occupied' and 'free' status was distributed evenly, giving a good indication of consistency in the data and also depicting a continuous change in status as one can observe in any parking bays.

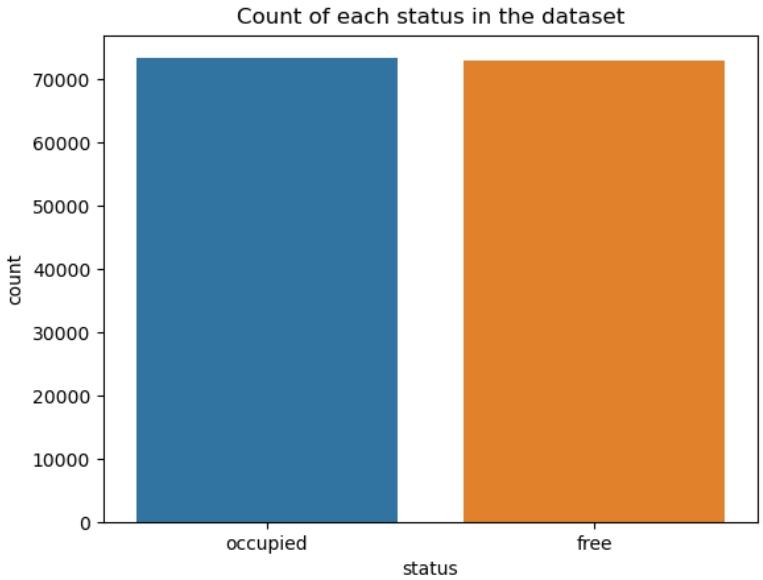


Figure 3.3: Status Count

3.2.1.2 Visualising Parking ID:

The importance of visualising the parking ID was to determine the use of each parking spot and further explain the parking patterns. First, I tried to display the count of unique

identifiers 'parking_id' which came up to 290 different spots present in the dataset. The graph below represents the count of records for each ID in the dataset. The spike in the graph represents more instances of id having more information indicating frequent changes in status for those parking spots.

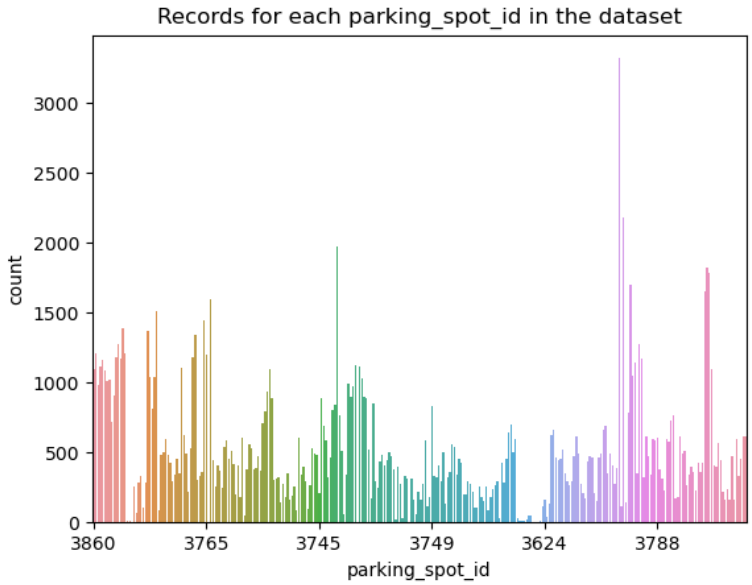


Figure 3.4: Parking ID count

Few Parking ID's indicated insufficient data, I tried to print those ids as seen in the Table 3.2 below

Table 3.2: Parking Spot Statistics

Row_id	**Parking_spot_id**	**Count_of_records**
0	3860	3323
1	3862	2180
2	3716	1973
3	3903	1822
4	3904	1784
⋮	⋮	⋮
285	3656	4
286	3618	4
287	3811	3
288	3812	3
289	3813	3

This table shows the inefficiency of the records for a few parking IDs as there is insufficient information and might not be the best practice to use these IDs for modelling as it can reduce the efficiency of the data.

3.2.1.3 Visualising Time Columns

The 'last_updation_time' represents the timestamp where the information about this dataset for a given row was last updated (or) changed. Whereas the 'start_time' and 'end_time' represent the duration of a particular status in each period. Upon discussion with my mentor, the goal was to have the 'end_time' and 'last_updation' time similar during the collection of the data, we decided to visualise the data to verify the correctness of the data for this condition and if there are any changes between those two parameters and we observed that some rows had different records of 'end_time' and 'last_updation' time. As we can see in the Figure 3.5 below:

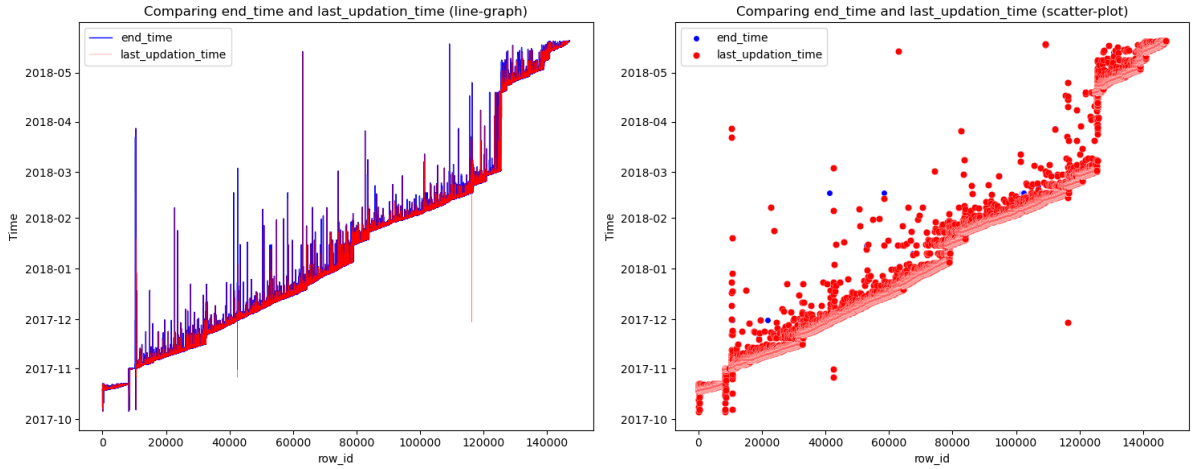


Figure 3.5: Comparing Time Columns

Most of the records were similar confirming their accuracy, we further wanted to calculate the difference between these two parameters and check for any outliers. My next step involved calculating the time difference between these two parameters storing it in a new column and displaying a scatter plot to check for any outliers as minor differences can be ignored but handling outliers will be necessary. As seen in the graph below the goal was to represent the difference between 'last_updation_time' and 'end_time' for each record. Only a few instances of record showed a large deviation in the graph, and I set a threshold of 10 minutes and found a total of 1892 records where the difference between these two parameters was greater than 10 minutes indicating a small fraction ¡1% of outliers. These outliers were dealt with during the pre-processing of data.

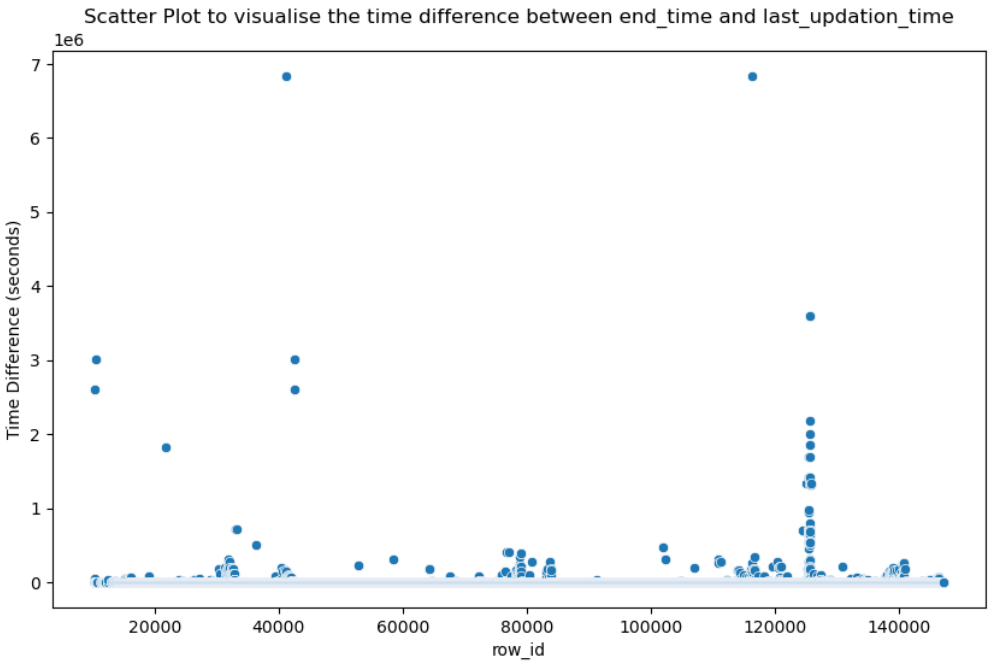


Figure 3.6: Comparing Time Difference

3.2.1.4 Visualising Duration

The duration column is the difference between 'start_time' and 'end_time' indicating the total time for a particular status at a given parking spot. The duration column was provided in the dataset, and to check its validity I recalculated the absolute difference between the 'start_time' and 'end_time' for each record and compared it to the given duration. I observed a total of 306 wrong calculations present in the records. I further calculated the difference between the 'calculated_duration' and provided duration to check the deviation and it can be seen in the Figure 3.7 below.

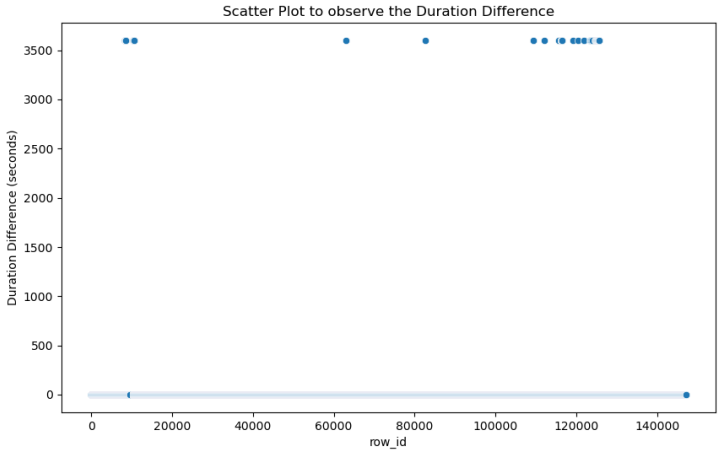


Figure 3.7: Comparing Duration Difference

As we can see in the Figure **??**, all the records of duration difference were set exactly to 3600 s indicating a slight error in the calculation of the original data. I then replaced the duration column with the newly calculated duration to have accurate records.

3.3 Data Cleaning and Pre-processing

The process of organising raw data within a dataset to increase its consistency by identifying, modifying, and editing to make it suitable for data analysis is called data cleaning (or) data cleansing. The process entails increasing efficiency without compromising the beneficial data. Incomplete and irrelevant data is eliminated or replaced with new data in a specific format for analysis. These errors in data occur during data entry and transmission due to loss or conflicting dictionary meanings of similar variables. The goal is to increase the data's validity, accuracy, and dependability to irradicate erroneous, fraudulent, and duplicate records in the dataset [21].

The main benefits of clean data efficiently are:

- To maintain data integrity which is defined as the overall accuracy, completeness,

and consistency of data. It also represents the security of the data to follow regulatory compliance for safety such as GDPR (General Data Protection Regulation) [22]

- Ensuring easy location of entries by discovering and removing errors to keep a consistent and usable record. This process is carried out to maintain up-to-date entries and make them easily accessible.
- Improving decision-making to improve the overall quality of data to reach greater accuracy. Analysing desired output and a better of targeted audience is a crucial metric when cleaning data. This causes lower errors subsequently improving efficiency.
- Increasing the quality of data by cleaning from multiple sources making it simpler to format and use in multiple cases. It increases the quality of data and overall accuracy for machine learning models. This process eradicates the older and irrelevant data and keeps only the updated desired records saving time when searching for records and increasing efficiency.
- Removing all unused and unnecessary data points from records. This is to prevent from creating mistakes and misinterpretation of data.
- Ensuring to have backups and spaces to store the data on cloud and local ensuring the safety and security of data.
- Importantly to ensure consistency throughout the data sets to conduct further analysis.
- Creating a logical relationship to avoid contradictions and gaps which ensures the organisation of data into segments creating an ease of access.
- To make sure the data is easily readable to make sense of data for the target audience.

3.3.1 Searching Noise in Data

The initial step was to search for noise in the dataset. Data noise is the undesirable and irrelevant information present in the records. Eliminating these noises will enhance the accuracy of analysis by removing the influence of irrelevant information. It is also necessary to increase the consistency throughout the records ensuring high data quality which increases the reliability of the dataset. While designing Machine Learning models, the removal of noise ensures the models learn meaningful patterns and relationships.
The substring in the 'parking_spot_id' column contained noise; the dataset used a Uniform Resource Name (URN) format to specify identifiers the information we needed to

extract was the unique parking identifiers. Removing additional substring from the 'parking_spot_id' column was necessary for better visualisation of data.

3.3.2 Data Formatting

Formatting refers to the process of organising and structuring in a consistent and standardised format to streamline data analysis and reduce errors further enhancing data quality and ensuring compatibility. Moreover, it promotes the understanding of data structure and is integral for data pre-processing. After eliminating the data noise, the next crucial step was to reformat the records. The focus was to convert the objects into the desired format. Formatting of the dataset is the I first converted the time columns. Subsequently, transforming all the columns as we can see in Table 3.3 into the desired format. The table below illustrates these transformations:

Table 3.3: Data Format Conversion

Columns	**Old Format**	**New Format**
row_id	int64	int64
parking_spot_id	object	int64
start_time	object	datetime64[ns]
end_time	object	datetime64[ns]
status	object	Boolean
duration	int64	float64
last_updation_time	object	datetime64[ns]

These conversions were essential to ensure the suitability of the data for analysis and interpretation.

3.3.3 Removing Incorrect Values

Removing incorrect records was the next concern in the data cleaning pipeline. This step is necessary to ensure accuracy and reliability in the records as incorrect records can cause anomalies, outliers and errors during analysis and pre-processing. It is also necessary to maintain the integrity and ensuring reliable and trustworthy records. While visualising the data, as described in the earlier section, the data was recorded between October 2017 and May 2018. I ran a script to check for time columns outside the period and I came across 119 records. These records were corrupted as they were outdated to 1970, which clearly indicated incorrect and inconsistent data. After removing these records, the next step was to check for missing and null values.

3.3.4 Handling Missing and Null Values

The absence of data in records is denoted as 'NaN' or 'null' signifying missing and null values respectively in the dataset. These values adversely impact the quality of the data by creating inaccuracies and biases in analysing. It is vital to remove or handle these errors to reduce distortion and potentially avoid errors. Data imputation is the best technique if removing the null or missing values causes a significant loss in data. I ran a script to check for any missing or null records in the dataset, I did observe some time gaps between records, which I will discuss in the data transformation section later, which I solved using data imputation but there were no null values or missing values in the records ensuring consistency and reliability of the data.

3.3.5 Handling Outliers

Outliers are abnormal or extreme values in the dataset. They differ from the majority of records in the dataset, and they should be dealt with as they can distort statistical analysis and manipulate the assumptions. It has the potential to skew results by influencing the calculation of central tendencies and dispersion such as mean and standard deviation. Outliers are caused during the collection process and ignoring can cause misinterpretation of the data causing biased estimates. Visualisation of graphs and plots can be difficult with outliers present causing difficulties in finding relationships and patterns [23].

Outliers were observed for time columns and were solved by elimination as they occurred as incorrect data. I further calculated and visualised outliers for the remaining columns as we can see in Figure 3.8 .
There were no outliers present for other rows but there were outliers present for the duration column. Upon research and discussion with my mentor, the main cause for these outliers was discovered due to construction in different parking regions between October 2017 and May 2018. I further visualised the area and 'parking_id' with the duration column confirming the pattern explaining different regions being impacted during the construction. Only a few of the parking IDs were influenced during this construction and outliers were observed during those dates. As only a few areas were impacted, we decided to further visualise the data in the sections below to check the impact of these outliers and what the best solution would be.

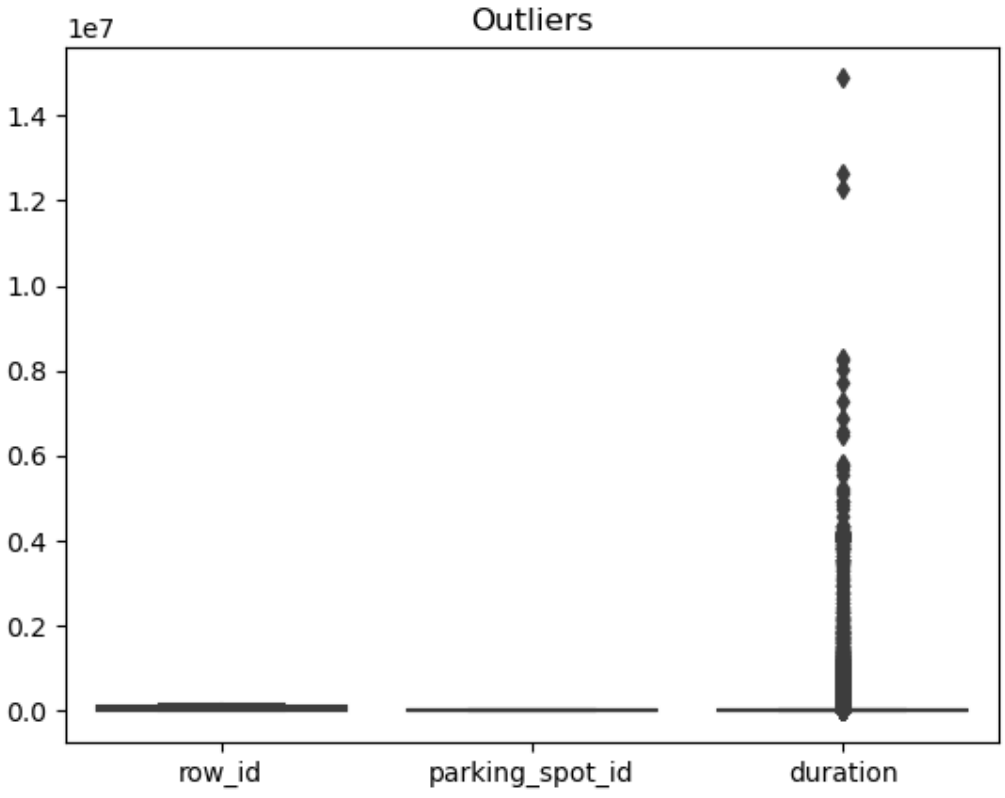


Figure 3.8: Outliers

3.3.6 Data Pre-Processing

The process of converting raw data into a desirable defined format to easily and effectively process data mining, machine learning and other data science tasks is called data pre-processing. It is a crucial step to increase consistency effectively increasing its accuracy and reliability. There are two key features of data pre-processing:

- Data validation: To achieve the best results, the process involves analysing and evaluating raw data to establish its validity and accuracy.
- Data imputation: Process of working with missing values and rectifying errors in data during the validation process. [24]

3.3.6.1 Pre-processing status columns:

During the pre-processing of the status column, I wanted a Boolean value. I set the occupied status as 1 (True) and the free status as 0 (Free) replacing the column completely.

3.3.6.2 Pre-processing Time Columns:

As explained during analysis we observed different values in the 'end_time' column and 'last_updation' time column. The situation was resolved in this section.

3.3.7 Pre-processing Duration Column:

As we can see in the Table 3.4, few parking ids had fewer records and thus had fewer durations for these records, my next step involved calculating the total duration for each 'parking_id' as seen in the table below:

Table 3.4: Total Duration for Parking Spots

Parking_spot_id	**Total Duration (seconds)**
3811	0
3813	0
3812	0
3807	115,504
3808	355,922
⋮	⋮
3912	18,505,571
3663	18,693,238
3825	18,709,816
3913	18,835,562
3799	19,093,613

This table shows us the records for every ID and as we can see few IDs have no duration which does not contribute to the model at all and can cause bias after discussing with my mentor, we decided to get rid of all rows with 0 duration. Furthermore, it causes to remove few IDs with no contribution to the data completely (e.g., 3811,3812,3813) as we can see in the table above. After pre-processing this was the data that we achieved:

Table 3.5: Table after data pre-processing

row_id	parking_spot_id	start_time	end_time	status	calculated_duration
1	3601	2017-10-18 09:57:04	2017-10-18 16:12:45	1	22,541.0
2	3602	2017-10-18 09:57:02	2017-10-18 16:00:30	1	21,808.0
3	3603	2017-10-18 09:24:10	2017-10-18 12:06:10	0	9,720.0
4	3604	2017-10-18 09:22:30	2017-10-18 13:17:46	1	14,116.0
5	3605	2017-10-18 10:01:12	2017-10-18 13:17:46	0	11,794.0
⋮					

3.3.8 Data Transformation

The process of converting raw data into a validated, cleaner, and converted into ready-to-use format is called data transformation. Data transformation involves these steps which fall under ETL (extract/transform/load) process:

- Data discovery - Using profiling scripts and graphs to understand the structure and characteristics of the data.
- Data mapping – Creating relations and patterns between data field sources is called data mapping.
- Code generation – Script used to transform the data into a desirable format.
- Execution of the code – Process of engaging in the transformation process.
- Review – Confirming the output and whether the data meets transformation requirements giving us an insight into whether any errors are present in the records.

Data Transformation helps us achieve a higher quality of data by reducing errors. Meanwhile, increasing retrieval times and requiring fewer resources to manipulate data helps us organise and manage efficiently [25].

After Cleaning the data, data transformation was vital to organise the data in a desirable format. At first, we needed to group the table based on parking spots and sort them based on time columns, creating a chain of sequential format, and getting it ready for pre-processing to convert it into time series format.

After imputing data, the next step was to transform the data into a time series format. This required us to design a function which accepts 3 parameters – the data frame, time interval and threshold percentage. The steps taken to achieve this transformation were:

- Data discovery - Using profiling scripts and graphs to understand the structure and characteristics of the data.
- Iterating Through the data to gather all the vital information from the data to organise a time series model.
- Defining Time Interval to provide it as a parameter represented by 'interval_minutes'. This was used to sample and organise the data in intervals within the time series.
- Determining the interval status to initially hold the status of the current event if it meets a threshold condition.
- Dividing the time columns into time intervals by calculating interval start time and end time ensuring it aligns with the interval and threshold parameters.

The crucial part was to apply the threshold condition to check the status of the current event by calculating the status time represented by the duration of each status within the time interval. The interval status varies based on the threshold percentage condition, Interval status remains the same as the current status if it meets the occupancy threshold percentage specified by the user. Otherwise, it is switched to its complement. Construction of the model into a new data frame representing the original data transformed into a time series based on the specified time interval and threshold conditions. Example Usage: To demonstrate the change into a time series model, a 20-minute time interval was provided as a parameter with a threshold percentage of 80% based on a literature survey.

Table 3.6: Setting a threshold percentage and arranging the data based on 20 minute intervals

parking_spot_id	**start_time**	**end_time**	**status**
3600	2017-10-18 10:40	2017-10-18 11:00	1
3600	2017-10-18 11:00	2017-10-18 11:20	1
3600	2017-10-18 11:20	2017-10-18 11:40	1
3600	2017-10-18 11:40	2017-10-18 12:00	1
3600	2017-10-18 12:00	2017-10-18 12:20	1
⋮	⋮	⋮	⋮
3600	2017-11-01 10:52	2017-11-01 11:12	1
3600	2017-11-01 11:12	2017-11-01 11:32	0

A time series model was formatted with an interval of 20 minutes as we can see in the figure above, the status was calculated based on the threshold percentage of 80% provided by the user.

3.3.9 Dimensionality Reduction

The reduction of features or attributes in a dataset and retaining all the vital information is called dimensionality reduction. The main purpose of using this technique is to simplify overly complex data for better visualisation, interpretation, and ease of understanding. Dimensionality reduction increases the processing speed and requires minimum computational resources therefore increasing the efficiency of the model. It can help reduce overfitting by removing noise and irrelevant features as the model generally performs well on existing training data and is not able to perform well on newly trained data. Feature engineering is more meaningful and gives a better insight into dimensionally reduced data creating a better interpretation of the model. Overall, it is. A valuable tool for improving the quality and increasing the interpretability for analysis, modelling, and decision-making.

In our case the main features that needed dimensionality reduction from the original data were 'last_updation_time' and duration, as the model was changed into a time series model, the duration gets set in a time interval during this process and the calculation becomes consistent creating no distinct features and increasing the size of the data without providing any useful information. For the latter column, the calculations were based on the start time and end time of the model. Therefore, removing a need for the 'last_updation_time' column.

3.3.10 Feature Selection and Engineering

The selection, manipulation, and transformation of raw data into features to be used in AI/ML models is called feature engineering. To increase the efficiency of the model it is necessary to design and train better features. A feature is defined as a measurable input that aids the model's prediction. In other words, the addition of features to the raw data gives meaning and creates patterns and relationships to help the model to understand the data better. The main goal is to simplify and enhance model accuracy by speeding up the transformation process. Feature Selection is a step to prepare data for modelling which involves the most relevant features to improve model performance and reduce noise by enhancing model interpretability [26].

Creating features for modelling was the next step towards building an efficient model. The data was formatted after imputation and transformation. I tried to enhance the richness and suitability of the data by adding temporal patterns, and duration information and creating features for time columns. Since the model was in a time series format of 20-minute durations, temporal patterns such as the hour of the day, the day of the week, month and year were a crucial extraction from the data as we can see while evaluating

results (refer to figure where you show the output of feature engineering) allowed the model to leverage these temporal patterns. Also adding a weekend indicator helped the model to differentiate different patterns in the dataset by incorporating a binary feature to determine the condition of the week.

The idea to add such features captured the underlying patterns and variations that impact parking spot occupancy status. Providing context for the model to increase accuracy in prediction and understanding the dynamics of parking usage over a period of time.
The feature engineered data is shown in Figure **??**

Out[57]:

	parking_spot_id	start_time	end_time	status	hour_of_day	day_of_week	month	year	duration_minutes	is_weekend
0	3600	2017-10-18 10:40:00	2017-10-18 11:00:00	1	10	2	10	2017	20.0	0
1	3600	2017-10-18 11:00:00	2017-10-18 11:20:00	1	11	2	10	2017	20.0	0
2	3600	2017-10-18 11:20:00	2017-10-18 11:40:00	1	11	2	10	2017	20.0	0
3	3600	2017-10-18 11:40:00	2017-10-18 12:00:00	1	11	2	10	2017	20.0	0
4	3600	2017-10-18 12:00:00	2017-10-18 12:20:00	1	12	2	10	2017	20.0	0
...	...	...	...	...	...	...	...	...	...	...
4024522	3923	2018-05-19 21:29:00	2018-05-19 21:49:00	1	21	5	5	2018	20.0	1
4024523	3923	2018-05-19 21:49:00	2018-05-19 22:09:00	1	21	5	5	2018	20.0	1
4024524	3923	2018-05-19 22:09:00	2018-05-19 22:29:00	1	22	5	5	2018	20.0	1
4024525	3923	2018-05-19 22:29:00	2018-05-19 22:49:00	1	22	5	5	2018	20.0	1
4024526	3923	2018-05-19 22:49:00	2018-05-19 23:09:00	0	22	5	5	2018	20.0	1

4024527 rows × 10 columns

Figure 3.9: Feature Engineered Time Series Data

3.3.11 Data Imputation

The process of using alternate values when there is a missing record is referred to as unit imputation in statistics. Data imputation is a process of preserving the data and substituting any missing values with a different value. Removing data is not the best practice as it reduces the features and causes gaps which substantially cause a bias and impair analysis [27].

After converting the data into a sequential time series model, I checked for any gaps in the data and there were 280 gaps present in the time series model. Solving this gap issue was crucial to reducing any biases and creating a consistent pattern in the data frame. I had to address this by filling gaps with the interpolation technique. I created a function to go through a loop for each spot at a time and to check for a time frame gap between any records and when a gap was detected a new row was generated with the gap set as a time frame between the current row and next row and the status for that time frame was assigned by using random imputation to reduce the biases.

Chapter 4

Methodology

4.1 Selection of AI/ML Models

4.1.1 Exploration of Models

Artificial Intelligence models are designed to solve complex problems by simulating human behaviour. They utilise machine learning techniques to mimic logical decision-making from the available information. Machine Learning models are a subset of AI models. A Machine Learning model gives the machine the capacity to learn without being explicitly programmed. In this section, I will discuss about the various AI/ML models I explored:

- Deep Neural Network
- Recurrent Neural Network
- Linear regression
- Logistic regression
- Decision trees
- Random Forest

4.1.1.1 Deep Neural Network (DNNs)

Being the most popular model used in AI/ML predictions, the model is influenced by the human brain and its neural networks. They are generally used to process large datasets by combining multiple inputs and providing a single output [28]. Artificial Neural Networks (ANN) use layers of artificial neurons to process data but it differs from Deep Neural Networks (DNN) regarding the depth of the architecture. As we can see in Figure 4.1, DNN consists of two or more hidden layers stacked in a sequence whereas ANN has a single hidden layer or multiple shallow hidden layers. This depth in architecture replicates the human brain network allowing DNNs to learn intrinsically from the data [29]. Since

we have a large dataset which would require multiple layers of neurons and work well with sequential data which makes it is a great choice for our time series forecasting model.

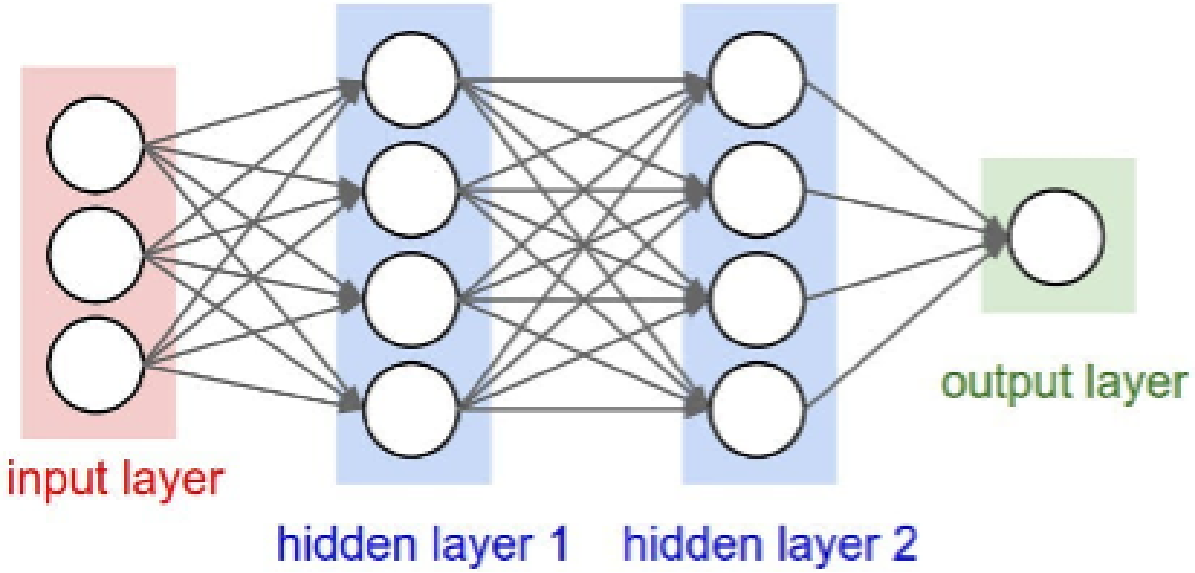


Figure 4.1: Two or more hidden layers comprise a Deep Neural Network. [29]

4.1.1.2 Recurrent Neural Network (RNNs)

RNNs are capable of handling sequential input by preserving hidden states and retaining information from prior experience. The output of the prior step is then fed as an input in the current step. Neural networks generally predict independent variables but when dependent information is needed for prediction for example, If we need to predict the next letters in a word, the previous letters of the words are needed to make those predictions creating feedback from prior outputs as an input. RNNs main attribute is its hidden layers which store prior information as we can see in Figure 4.2 below and this state is known as the memory state [30].

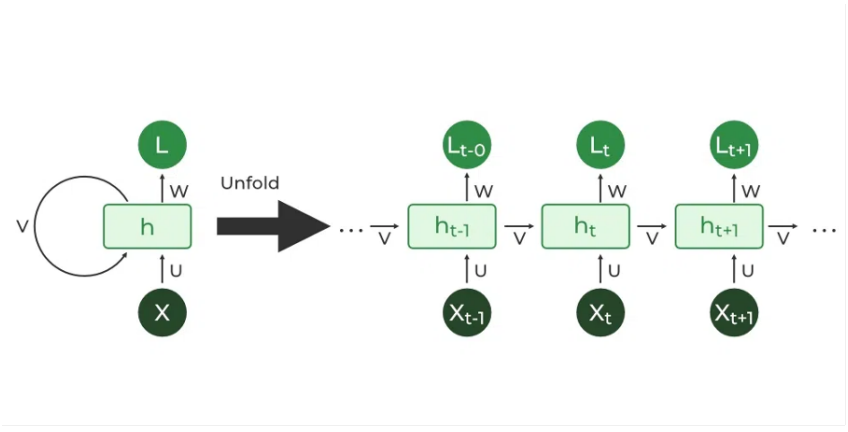


Figure 4.2: Recurrent Neural Netowrk. [30]

RNNs excel in prediction tasks where prior memory is vital to making future decisions which makes it a perfect fit for the time series model.

4.1.1.3 Linear Regression

It is one of the most popular statistical models present today. It is a supervised learning model [28]. Supervised learning is a machine learning paradigm where an algorithm is programmed to learn from a dataset to make predictions. These models have faster training convergence and are designed to find patterns between the input and output variables. The prediction capabilities of these types of models enable them to estimate the value of the dependent variable based on the value of the independent variable [28]. Since our model is a time series-based model containing statistical values, it would be a good choice to use linear regression.

4.1.1.4 Logistic Regression

It is a similar model to linear regression, but it is used to solve classification-based problems. They are generally used to solve binary classifications at predicting values based on a set of independent variables which is not what I was looking for to solve a time series forecasting model [28].

4.1.1.5 Decision Trees

These are derived models which are straightforward and highly efficient utilising data from past decisions for evaluation and prediction. Regression and classification problems are the best-used cases for these types of models. This is a simple and efficient model which could be deployed for our prediction [28].

4.1.1.6 Random Forests

Random forest is the combination of multiple decision trees. All trees have independent capabilities to predict their own result and combining the prediction can highly increase accuracy during the final prediction causing random forests as a great candidate for developing the time series prediction model for parking area availability [28].

4.1.2 Criteria for Model Selection

The journey to choose correctly started with investigating different AI/ML models. This process is time-consuming, and I had to review various literature, consult domain experts and conduct experiments to make the right choice to get the optimum prediction and set criteria which align with my goals and objectives for the dataset. The main criteria I considered during my model selections were:

- Accuracy: My goal was to get the most efficient and accurate prediction which involved the selection of a model with a track record of achieving higher accuracy on time-series prediction models.
- Training Time: Considering my computational resources and the time constraint of working on a big model within these two months, I had to select a model that could utilise these requirements.
- Data Type: The nature of the time series model data played a vital role in my choice of selecting the model. This was set after the Pre-processing of the data.
- Interpretability: As the prediction was required for parking spots as well as parking areas, I had to create an interface for users to input parking IDs and areas which needed an understanding of the decision-making process creating a vital component in my decision-making.

4.1.3 Justification for Model Choice

For urban parking management, the time series forecasting model played an important role in maximising resource allocation. Accurate measurements can help to reduce traffic, reduce search times and increase the efficiency of the parking spot. The literature review section covers the literature reasoning for my selection. In this section considering all the criteria I mentioned above, I decided to train the model using Long Short-Term Memory LSTM and Random Forest outlining the advantages and disadvantages of my selection. Random Forests: The robust ensemble Model was an ideal fit for the time series model. The reason for my selection of this model is:

- Resilience to unreliable data: Data on a time series model can be corrupted due to several factors, including sensor malfunctioning and unexpected circumstances. This model exhibits an inherent tolerance for unreliable data by combining multiple decision tree predictions effectively minimising the influence of errors.
- Competence with the diverse dataset: The parking dataset consisted of a blend of numeric, time and categorical variables. Making Random Forest an excellent choice which can handle and effortlessly execute tasks involving different varieties in the dataset. Revealing them as an all-rounder for estimating the intrinsic nature surrounding parking availability forecasts.
- Comprehensibility: Understanding the patterns in forecasting models plays a crucial role when it comes to managing this industry conscientiously, easing informed decisions based on interpretable statistics is necessary for time series forecasting-based models. Random Forest's ability to allow precise quantification of important features is considered which creates an environment where operators control over

determining which parameters are most efficient facilitating logical-proactive steps towards optimal functionality. Interpretability of the parking areas and ID was a vital condition and choosing a random forest simplified model was more favourable to create an interface for users to visualise the table and graph format.

- Efficiency: Patterns includes real-time bulk analysis of large datasets and random forest has the capacity to handle large dataset and a wide variety of features making them suitable for batch processing of the time series model of parking dataset. Their ability to be versatile and parallelise exploits modern hardware efficiently.
- Ensemble Learning: Random forest is a combination of multiple decision trees, it helps reduce overfitting and provide stability and accurate predictions. They are great at generalising the patterns observed in time series data.

Long Short-Term Memory: I wanted to develop one model based on a Neural Network and LSTM, a type of recurrent neural network that was the best choice for a time series forecasting model when capturing complicated temporary dependencies. My decision to choose LSTM for the neural network approach is described below:

- Handling sequentially aligned information: The dataset contained sequential dependencies where past observations played a vital role in predicting future occupancy. LSTM in its architecture as discussed in the prior section about recurrent neural networks has memory cells to capture dependencies from prior observations, making it ideal for modelling the time series dataset where historic patterns influence future predictions.
- Long Range Dependencies LSTM is known to capture long-range dependencies, that are present in our dataset. For example, The parking space availability at a given time is influenced by patterns from historical data. The ability to feed the old output as an input in the hidden layer allows to capture of complex patterns which cannot be observed in other models.
- Variable Sequence Length: LSTM has the feature to handle irregular time intervals, which was sorted in my case by data imputation but having features that can accommodate missing data points without resampling is perfect for a time series model.
- Feature Engineering: The deep architecture extracts all relevant features from the data reducing the need to manually configure feature engineering and selection. This capability is crucial when dealing with large-scale time series data.

  Adaptability: The ability to adapt to varying forecasting horizons, creating flexibility that enables the model to tailor prediction to specific needs.

Conclusion:
The decision to choose LSTM and Random forests as the models for time series parking availability prediction is justified based on their advantages and compatibility with the dataset we have. While LSTM excels in detecting complex patterns in the time series dataset, Random Forest provides speed, resilience, and great interpretability. Using both models we can take advantage of the best aspects of neural network and ensemble learning models ensuring efficiency, accuracy, and reliability in the estimation of best parking solutions [28].

4.2 Model Development and Training

Having completed the model selection, the next crucial step is to develop and train the time series forecasting model to optimise the resource allocation. This section outlines the steps involved in developing and training the model focusing on key aspects such as data splitting, model training, hyperparameter tuning, feature engineering and utilising confusion matrices and K-Fold cross-validation techniques for model evaluation and validation. The calculations and results carried out from these configurations are mentioned in the next section.

4.2.1 Data Splitting for Training and Testing

The splitting of data into different subsets is called data splitting. This step is crucial to have an unbiased evaluation of the models by dividing the dataset into three subsets:

- Training Set: The portion of data used to learn fundamental patterns and relationships from the time series data.
- Validation Set: Model performance during training is monitored and hyperparameters are modified with the use of a validation set which prevents overfitting and provides the optimal selection of hyperparameter configurations.
- Test Set: This section of the data is stored for evaluation metrics to calculate the model's performance providing an unbiased evaluation of the capability of the model to generalise to unseen data.

The size of the dataset determines the choice of split ratio which is mentioned in the next chapter of this journal. In our case of limited data, techniques such as k-fold cross-validation are employed to ensure a robust evaluation which will be discussed in the latter section.

4.2.2 Training the Model

To train a general model, the model is mixed with the training set and adjusting its parameters to reduce error. LSTM and Random Forest differ significantly in this step due to their underlying architectures.

The robust ensemble model is constructed by multiple decision trees which are built using a subset of training data and the features which are selected at random at each split. The prediction is the average of each decision tree's prediction helping it to naturally handle parallelism and help with resource allocation on my MacBook which has multiple-core processors.

Whereas LSTM involved the use of a gradient-based optimisation algorithm (in my case I used Adam) whose adaptability in configuring learning rates for time series data makes it an ideal choice. During the training process, the model processes the data sequentially by utilising historical data through recurrent layers and utilises BPTT (Backpropagation Through Time) to adjust the weights.

4.2.3 Hyperparameter Configurations

They are crucial to determine the performance of the model. These configurations are external to the model and the valuation cannot be estimated from the dataset. They are generally used to estimate the parameters in the model and are set using heuristics. Deploying these configurations can help increase model accuracy and generalisation [31]. For my model, I estimated the results and evaluated the model for LSTM but calculated Random Forest Hyperparameters. The set of hyperparameters in Random Forest comprises the size of the decision tree, calculating the depth for each tree and the minimum samples needed to split a node. These configurations can solve overfitting and improve the overall performance of the data which can be seen when compared with the LSTM model which didn't have any hyperparameter configuration. The results of these metrics are discussed in the next chapter. Randomised search was employed instead of grid search to find the optimal hyperparameter settings for our model as it helps with resource allocation.

4.2.4 Feature Engineering and Selection

The focus was to train the model using Random Forest to eventually predict parking area availability and feature engineering for Random Forest is a crucial aspect of model development unlike for LSTM.

- Feature Engineering for these models involved converting the status (categorical value) into binary vectors of 1 or 0 indicating 'occupied' or 'free' status respectively.

- Scaling the numerical features in the data to create patterns and converting them into similar formats was crucial for consistency throughout the data.
- New Features were created based on the time series model such as the hour of the day, the day of the week, month, year, duration and whether the day is a weekend or not.

The influence of these feature creations is seen in Figure (x mention image from results). The quality of these features significantly impacts the performance of Random Forest. Providing additional features from the time series data increases the prediction accuracy and reduces errors.

4.3 Confusion Matrix and Cross-Validation Techniques

Confusion Matrix: It is a tool to assess the performance of the model and an indication of the accurate prediction of classifiers. It also provides information about the incorrect classifier and where the model got confused [32]. The confusion matrix for the Random Forest model is calculated in the results section of this paper.

Classification Report: To get more insight into the model's performance the confusion matrix calculates the following performance metrics [33]:

- **Accuracy:** It is defined as the ratio of accurately predicted instances to the total number of instances, giving an overall measure of predictive accuracy as seen in the formula below:

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}}$$

- **Precision:** Indicates the correctly predicted cases which came true. It quantifies the ability of a model to understand all relevant positive instances and is presented by the formula mentioned below:

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

- **Recall (Sensitivity or True Positive Rate):** It indicates the amount of actual positive cases predicted correctly by our model and is as follows:

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

- **F1-Score:** The F1-score is inversely proportional to recall. It is a harmonic mean of precision and recall, providing an ideal measure of the model's performance:

$$\text{F1-Score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

Cross-Validation Techniques: For my model, I used the K-Fold Cross-validation technique, which estimated the performance of the Random Forest model and assessed its general ability. It reduces the overfitting concerns and provides a detailed model evaluation by dividing the dataset into a 'k' subset or 'folds'. The formula for mean and standard deviation of performance metrics across each 'k' iteration is expressed as follows [34]:

$$\text{Mean} = \frac{1}{K} \sum_{i=1}^{K} \text{Score}_i$$

$$\text{Standard Deviation} = \sqrt{\frac{1}{K} \sum_{i=1}^{K} (\text{Score}_i - \text{Mean})^2}$$

The working of K-Fold Cross-Validation is described below:

- Data Splitting -Partitioning of data into a 'k' subset of equal-sized folds where each fold consists of a balanced depiction of the records.
- Training and Testing – The training and testing of models are carried out 'k' times, each fold is used in subsequent iterations as a test set while the remaining 'k-1' folds are evaluated as training sets.
- Performance Evaluation: The model's performance is evaluated in each iteration using metrics such as accuracy, precision, recall or F1-score.
- Average Metrics: Averaging the performance metrics obtained from kfolds obtains a more robust estimate of the model's performance of the model.
- Model Selection and Tuning: According to the results a comparison of hypermeter settings is done and the best-performing model is selected.

Developing the time series forecasting model using LSTM and Random Forest involved a comprehensive selection process through exploration of options, consideration of hyperparameter settings and continuous evaluation using confusion matrices and K-Fold Cross-

Validation techniques. Ensuring the suitability of the model for my case, the results for these parameters are shown in the next chapter and this section covered the overview of the methodology I used and the reasons behind it [34].

Chapter 5

Experimental Results

This section aims to critically evaluate the development process with future scope and shortcomings in the project.

5.1 Experimental Results

The Dataset used in the model collected data from 400 on-street parking sensors in Santander, Spain. The initial step was to visualise the data and to find patterns to understand the structure of the data. Performing exploratory analysis and visualising the data in multiple ways, I analysed the dynamics of the data. Then, the next step involved pre-processing and data imputation. I pre-processed the data step-by-step, as mentioned in the data pre-processing and cleaning section. Designing a function to set a threshold percentage and restructuring the data based on time intervals was one of the biggest challenges in the development process. I had to completely reformat an enormous dataset to achieve the desirable time series format. Filling gaps was another obstacle when I was transforming the data. There is a better approach to fill the gaps in the data than the random imputation approach I took since I was running short on time and decided to go with that approach. After Feature Engineering, the next step was to train the model. In this section, I intend to compare performance of both models.

5.1.1 Model Performance Metrics

To predict the parking area availability, I employed two models: a Recurrent Neural Network model – LSTM and a Random Forest model. Each model had its own benefits and drawbacks, in this section I intend to visualise the performance metrics and compare both the models. The accuracy results for both the models are shown in the table below: The LSTM model, known for its resilience to unreliable data achieved a test accuracy of 77.71%. Whereas Random Forest achieved a test accuracy of 88.59%, indicating constraints on LSTM models to predict long-term variation. Random Forest has a better

Table 5.1: Comparison of Model Performance

Metric	LSTM	Random Forest
Accuracy	0.7735	0.8860
Mean Square Error	0.1627	0.1612
R-square Error	0.3203	0.3256

probability of performing against the test data in long-term variation. Random forest achieved a higher r-squared value and a lower mean square error, indicating a better fit for the data by capturing more underlying patterns unlike the LSTM model, and the model's prediction is more accurate to actual values. With an overall high performance and a better fit, Random Forest is a better choice for the time series data.
The training and validation loss and accuracy is shown in Figure 5.1, indicating the learning trends of the model.

Figure 5.1: Training and Validation performance of LSTM.

5.1.2 Hyperparameter Optimisation methods

I did hyperparameter tuning only for Random Forest to find the best hyperparameters. Since the computational power was very high, evaluating hyperparameters for both models on the cloud and locally was very stressful. However, I found the best hyperparameter through a Randomized search, fine-tuning the Random Forest model to achieve high test scores in the table above. The best hyperparameter configuration is shown in the figure below: My decision to use randomised search over grid search was due to its requirement of less computation resources while generating great results making it suitable for large-sized data.

5.1.3 Other Important Metrics

From the Confusion matrix and Classification Report Metrics these readings were obtained, shown in the Table 5.2. Random forest outperforms LSTM on all metrics. Al-

Table 5.2: Performance Metrics for Classification

Metric	LSTM	Random Forest
Precision for Class 0	0.85	0.92
Recall for Class 0	0.52	0.78
F1-score for Class 0	0.65	0.84
Precision for Class 1	0.75	0.87
Recall for Class 1	0.94	0.96
F1-score for Class 1	0.83	0.91
Overall Accuracy	0.7735	0.8860

though LSTM is good at identifying occupied parking spots with its lower precision of class 0 and lower accuracy compared to random forest, it makes it slightly unsuitable. I believe with the right Hyperparameter of LSTM, it can match the performance of Random Forest as I have found the best Hyperparameter for Random Forest but not for LSTM.

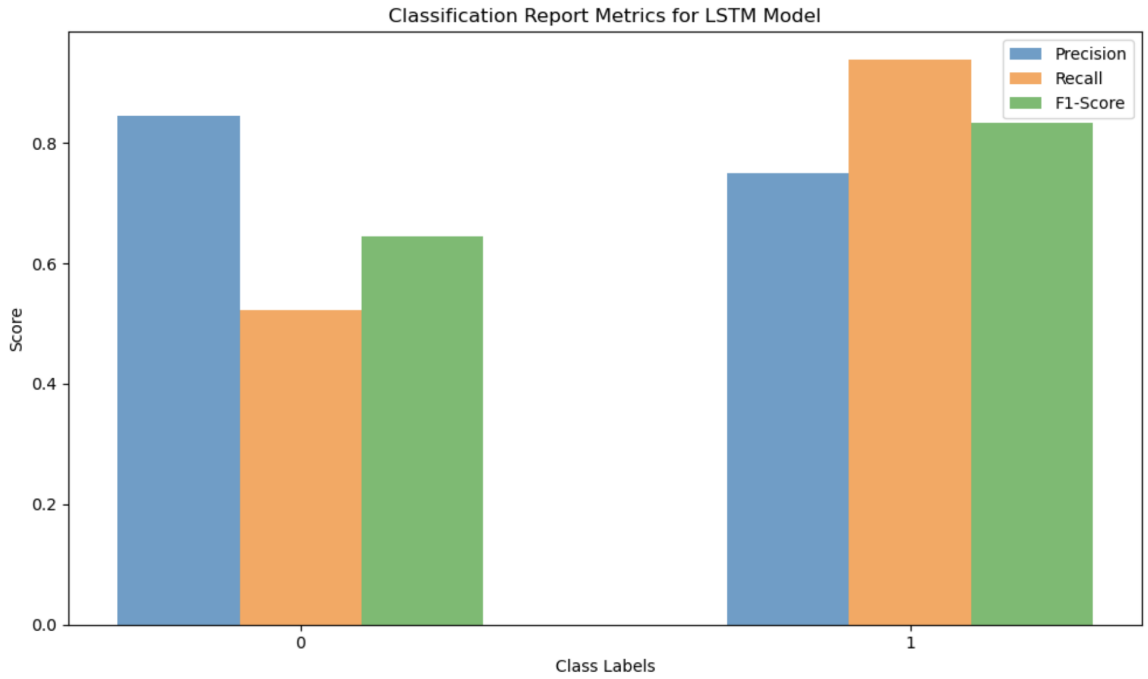


Figure 5.2: Classification Report Metrics for LSTM

Importance of each feature when modeling Random Forest is shown in Figure 5.3

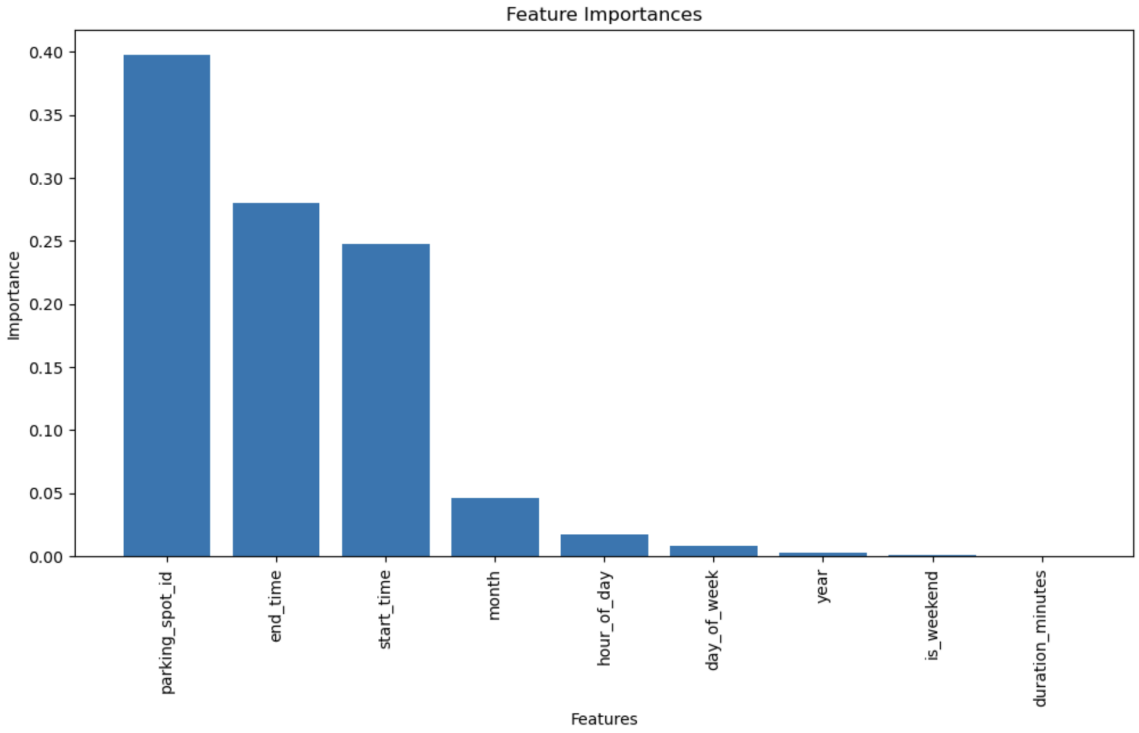


Figure 5.3: Feature Importance Metric

The prediction results for modelling Parking Spot ID and Parking Area are shown below. One of the shortcomings of the project due to time constraints was to integrate coordinates and visualise using Streamlit. I have designed two separate prediction model that requires the user to input a parking spot ID and parking area to get a prediction of the percentage of parking area availability for the model. I was provided with a JSON object containing details about the coordinates and area of each parking spot. I integrated the area into the time series dataset. But the next step involves to make a better interface.

```
Enter Parking Spot ID to print predictions (or enter -1 to exit): 3600
              Start Time            End Time  Predictions
0    2017-10-18 10:40:00 2017-10-18 11:20:00     0.988000
1    2017-10-18 11:20:00 2017-10-18 12:00:00     0.988500
2    2017-10-18 12:00:00 2017-10-18 12:40:00     1.000000
3    2017-10-18 12:40:00 2017-10-18 13:20:00     1.000000
4    2017-10-18 13:20:00 2017-10-18 14:00:00     0.994253
...                  ...                 ...          ...
8352 2018-05-19 16:03:00 2018-05-19 16:43:00     0.746345
8353 2018-05-19 16:43:00 2018-05-19 17:23:00     0.746345
8354 2018-05-19 17:23:00 2018-05-19 18:03:00     0.640340
8355 2018-05-19 18:03:00 2018-05-19 18:43:00     0.727202
8356 2018-05-19 18:43:00 2018-05-19 19:23:00     0.727202

[8357 rows x 3 columns]
Enter Parking Spot ID to print predictions (or enter -1 to exit): 3607
               Start Time            End Time  Predictions
58529 2017-10-18 07:25:00 2017-10-18 08:05:00     0.354921
58530 2017-10-18 08:05:00 2017-10-18 08:45:00     0.626595
58531 2017-10-18 08:45:00 2017-10-18 09:25:00     0.626595
58532 2017-10-18 09:25:00 2017-10-18 10:05:00     0.351702
58533 2017-10-18 10:05:00 2017-10-18 10:45:00     0.643417
...                   ...                 ...          ...
66848 2018-05-20 03:28:00 2018-05-20 04:08:00     0.592829
66849 2018-05-20 04:08:00 2018-05-20 04:48:00     0.605890
66850 2018-05-20 04:48:00 2018-05-20 05:28:00     0.605890
66851 2018-05-20 05:28:00 2018-05-20 06:08:00     0.750023
66852 2018-05-20 06:08:00 2018-05-20 06:48:00     0.514547
```

Figure 5.4: Parking ID Prediction

```
Enter Area (e.g., area-R) to print predictions (or enter 'EXIT' to quit): area-r
Model not found for Area area-r
Enter Area (e.g., area-R) to print predictions (or enter 'EXIT' to quit): area-R
                     Time  Predictions
1454824 2017-10-18 06:45:00     0.728713
1454825 2017-10-18 07:25:00     0.728713
1454826 2017-10-18 08:05:00     0.728713
1454827 2017-10-18 08:45:00     0.728713
1454828 2017-10-18 09:25:00     0.728713
...                     ...          ...
2061848 2018-05-19 20:49:00     0.574708
2061849 2018-05-19 21:24:00     0.574708
2061850 2018-05-19 21:29:00     0.574708
2061851 2018-05-19 22:09:00     0.574708
2061852 2018-05-19 22:49:00     0.574708

[47600 rows x 2 columns]
Enter Area (e.g., area-R) to print predictions (or enter 'EXIT' to quit): area-A
                     Time  Predictions
83218  2017-10-18 10:00:00     0.524335
83219  2017-10-18 10:40:00     0.524335
83220  2017-10-18 11:20:00     0.524335
83221  2017-10-18 12:00:00     0.524335
83222  2017-10-18 12:40:00     0.524335
...                    ...          ...
125167 2018-05-18 10:17:00     0.522034
125168 2018-05-18 10:25:00     0.522034
125169 2018-05-18 10:34:00     0.522034
125170 2018-05-18 11:14:00     0.522034
125171 2018-05-18 11:54:00     0.522034
```

Figure 5.5: Parking Area Prediction

Chapter 6

Critical Evaluation

6.1 Answering Research Questions

- Can a machine learning model predict parking area availability based on the historical data in Santander, Spain?- I was able to design a machine learning model to predict the parking area availability for the time series dataset. The data was collected from on-street parking sensors in Santander, Spain. I employed Long Short-Term Memory and Random Forest to achieve great results as mentioned in the experimental results section of the paper.
- What are the benefits of modelling a machine learning model to aid the drivers in Santander, Spain? - Drivers are getting frustrated every day in search of parking spaces in Santander, Spain and it has become a global problem. The model is designed with a vision to save time, fuel and avoid traffic congestions by solving parking problems in the city. It can also help optimise resources efficiently and integration of smart city architecture.
- How to clean and design the data for a machine learning model to predict parking space availability? - Cleaning of the data involved several steps as mentioned in the data cleaning section of this paper. I was able to clean the data efficiently and transform it into time series to get the most optimum output.
- Which machine learning model to use to solve the parking area prediction problem? - The choice of machine learning model depends on various factors, I have provided a reasoning for my choice in depth in the model evaluation and selection section of the paper. In brief, LSTM ability to handle large sequential datasets and Random forests great comprehensibility and help reduce overfitting of data was one of several reasons why I chose to go ahead with these models.
- What are the technical challenges of designing such models? - I had various technical challenge when designing the model. Handling large unorganised dataset, cleaning

and pre-processing of such data took a lot of time in the development process. Evaluation of different metrics and finding the best model was a learning curve for me. Selection of evaluation metrics was not an easy task either, I tried the best to solve all the problems but the model stile can improve and has a lot of potential.

- What is the importance of feature engineering in such models? - Feature Engineering was a crucial step when I was pre-processing and transforming the data. As they play an important role and extracting feature metrics which enhances the readability of data by providing additional context.

6.2 Conclusion

Smart Cities is one of the fastest-growing topics in the field of Machine Learning. Parking availability prediction is at the core of society's problem today, causing traffic congestion, accidents and polluting the environment. The best solution to solve these issues is to integrate a smart city architecture where the drivers have information about the parking space availability saving an enormous amount of time and resources.

In this paper, I aimed to learn about parking prediction availability and contribute to the field. I first introduced the problem statement with an overview of my approach where I performed multiple literature surveys exploring different models, to understand the reasoning behind a choice of a model for a specific case. I was provided with a dataset and a JSON file with the parking history in the region of Santander, Spain and with regular meetings with my supervisor, having a version control (GITLAB) and meeting meetings to keep up-to-date progress.

I was given the tasks to clean, pre-process and transform the data into a time series model. This was the main part of the project and with the constant support of my supervisor, I was able to achieve great results. Then I had to decide on the model choice. The literature survey helped me understand which approach yields good results and after reading about similar-sized data and their performances across different models, I was increasingly confident and excited to make a decision on the model choice and work with it. I then decided to go ahead with LSTM and Random Forest, working with hyperparameters and different classification metrics to yield the best result. I was able to model my data and produce outputs.

6.3 Limitation and Future Work

Although, I was able to finish most of the tasks. With my limited knowledge in this topic, time constraints and resource management This project had few limitations which I wish to improve further. The limitations and future work for Parking Area Avaialbility are:

- Evaluating the performance of various other AI/ML models and comparing them to find the best solution.
- Although my data cleaning was satisfactory, my random imputation techniques for gaps need more to be desired. Surveying for better solutions to fix gaps in data and transforming it.
- Utilising different type of data and using clustering techniques to find the best solution to the problem.
- Ability to find better feature points and learn patterns from models to achieve these metrics which requires contiunous trial and error
- The model is only able to predict parking id prediction and are prediction for specified input. I would like to expand its scope further and create much more metrics for configuration
- Including of external factors such as weather condition or a public holiday to influence my model will increase its efficiency enormously.
- Using streamlit to interface the prediction model, utilising the coordinates to display the location of the prediction in a more interpretable manner.
- Finding more patterns and finding better hyperparameters. Due to time constraints and resource limitation, I was able to only model Random foresst with a hyperparameter configuration using randomised search.
- Reduce overfitting of model, due to not having the best hyperparameter, I had a slight overfitting issue with LSTM mdodel.
- Explore Grid Search and compare the results with randomised search as grid search is slower but has better parameters and due to resource and time constraints, I wasn't able to perform grid search instead I adapted to a quicker hyperparameter using randomised search.

With all these limitations and scope for futrue work, I intend to research further in this topic and try to increase efficiency of the model.

Bibliography

[1] Yasir Shaikh. "Details - msc project." [Accessed 28-09-2023], teaching.dcs.aber.ac.uk. (2023), [Online]. Available: `https://teaching.dcs.aber.ac.uk/mscpm/Suggestions/Details/45`.

[2] D. Shoup, "Free parking or free markets," in *Parking and the City*, Routledge, 2018, pp. 270–275.

[3] V. Paidi, "Short-term prediction of parking availability in an open parking lot," *Journal of Intelligent Systems*, vol. 31, no. 1, pp. 541–554, 2022.

[4] P. Van Der Waerden, H. Timmermans, and P. Barzeele, "Car drivers' preferences regarding location and contents of parking guidance systems: Stated choice approach," *Transportation research record*, vol. 2245, no. 1, pp. 63–69, 2011.

[5] A. Koster, A. Oliveira, O. Volpato, V. Delvequio, and F. Koch, "Recognition and recommendation of parking places," in *Ibero-American Conference on Artificial Intelligence*, Springer, 2014, pp. 675–685.

[6] W.-J. Park, B.-S. Kim, D.-E. Seo, D.-S. Kim, and K.-H. Lee, "Parking space detection using ultrasonic sensor in parking assistance system," in *2008 IEEE intelligent vehicles symposium*, IEEE, 2008, pp. 1039–1044.

[7] A. S. SAĞLAM and F. ÇAVDUR, "Prediction of parking space availability using arima and neural networks," *Endüstri Mühendisliği*, vol. 34, no. 1, pp. 86–108, 2023.

[8] W. Shao, Y. Zhang, B. Guo, K. Qin, J. Chan, and F. D. Salim, "Parking availability prediction with long short term memory model," in *Green, Pervasive, and Cloud Computing: 13th International Conference, GPC 2018, Hangzhou, China, May 11-13, 2018, Revised Selected Papers 13*, Springer, 2019, pp. 124–137.

[9] F. Bock, S. Di Martino, and A. Origlia, "A 2-step approach to improve data-driven parking availability predictions," in *Proceedings of the 10th ACM SIGSPATIAL workshop on computational transportation science*, 2017, pp. 13–18.

[10] P. Zhang, L. Xiong, Z. Yu, *et al.*, "Reinforcement learning-based end-to-end parking for automatic parking system," *Sensors*, vol. 19, no. 18, p. 3996, 2019.

[11] Y. Sasaki, J. Takayama, J. R. Santana, S. Yamasaki, T. Okuno, and M. Onizuka, "Predicting parking lot availability by graph-to-sequence model: A case study with smartsantander," in *2023 24th IEEE International Conference on Mobile Data Management (MDM)*, IEEE, 2023, pp. 73–80.

[12] G. Jelen, V. Podobnik, and J. Babic, "Contextual prediction of parking spot availability: A step towards sustainable parking," *Journal of cleaner production*, vol. 312, p. 127 684, 2021.

[13] S. Inam, A. Mahmood, S. Khatoon, M. Alshamari, and N. Nawaz, "Multisource data integration and comparative analysis of machine learning models for on-street parking prediction," *Sustainability*, vol. 14, no. 12, p. 7317, 2022.

[14] M. A. Kuhail, M. Boorlu, N. Padarthi, and C. Rottinghaus, "Parking availability forecasting model," in *2019 IEEE International Smart Cities Conference (ISC2)*, 2019, pp. 619–625. DOI: 10.1109/ISC246665.2019.9071688.

[15] Simplilearn. "What is data collection: Methods, types, tools." [Accessed 28-09-2023], simplilearn.com. (2023), [Online]. Available: https://www.simplilearn.com/what-is-data-collection-article.

[16] G. Lawton. "Data preprocessing: Definition, key steps and concepts." [Accessed 28-09-2023], techtarget.com. (2023), [Online]. Available: https://www.techtarget.com/searchdatamanagement/definition/data-preprocessing#:~:text=Data%20preprocessing%20transforms%20the%20data,pipeline%20to%20ensure%20accurate%20results.

[17] F. M. Awan, Y. Saleem, R. Minerva, and N. Crespi, "A comparative analysis of machine/deep learning models for parking space availability prediction," *Sensors*, vol. 20, no. 1, p. 322, 2020.

[18] Smartsantander. "Santander facility." [Accessed 28-09-2023], smartsantander.eu. (2023), [Online]. Available: https://www.smartsantander.eu/index.php/testbeds/item/132-santander-summary.

[19] Y. Saleem, P. Sotres, S. Fricker, *et al.*, "Iotrec: The iot recommender for smart parking system," *IEEE Transactions on Emerging Topics in Computing*, vol. 10, no. 1, pp. 280–296, 2020.

[20] Simplilearn. "Data visualization: Why it is one of the top data skills for 2023." [Accessed 28-09-2023], simplilearn.com. (2023), [Online]. Available: https://www.simplilearn.com/data-visualization-article.

[21] I. E. Team. "Data cleaning: Definition, importance and how-to guide." [Accessed 28-09-2023], Indeed. (2023), [Online]. Available: https://sg.indeed.com/career-advice/career-development/data-cleaning.

[22] Talend. "What is data integrity and why is it important?" [Accessed 28-09-2023], talend.com. (2023), [Online]. Available: https://www.talend.com/uk/resources/what-is-data-integrity/.

[23] J. Frost. "Guidelines for removing and handling outliers in data." [Accessed 28-09-2023], statisticsbyjim.com. (2023), [Online]. Available: https://statisticsbyjim.com/basics/remove-outliers/.

[24] I. E. Team. "What is data preprocessing? (with importance and examples." [Accessed 28-09-2023], Indeed. (2023), [Online]. Available: https://ca.indeed.com/career-advice/career-development/data-preprocessing#:~:text=The%20following%20are%20some%20benefits,It%20makes%20data%20consistent.

[25] M. K. Pratt. "What is data transformation? definition, types, and benefits." [Accessed 28-09-2023], techtarget.com. (2023), [Online]. Available: https://www.techtarget.com/searchdatamanagement/definition/data-transformation.

[26] H. Patel. "What is feature engineering — importance, tools and techniques for machine learning." [Accessed 28-09-2023], towardsdatascience.com. (2021), [Online]. Available: https://towardsdatascience.com/what-is-feature-engineering-importance-tools-and-techniques-for-machine-learning-2080b0269f10.

[27] Simplilearn. "Introduction to data imputation." [Accessed 28-09-2023], simplilearn.com. (2023), [Online]. Available: https://www.simplilearn.com/data-imputation-article.

[28] J. Tarud. "Ai models: How does it work?" [Accessed 28-09-2023], koombea.com. (2023), [Online]. Available: https://www.koombea.com/blog/ai-models/.

[29] J. Johnson. "What's a deep neural network? deep nets explained." [Accessed 28-09-2023], bmc.com. (2020), [Online]. Available: https://www.bmc.com/blogs/deep-neural-network/.

[30] GeeksforGeeks. "Introduction to recurrent neural network." [Accessed 28-09-2023], geeksforgeeks.org. (2023), [Online]. Available: https://www.geeksforgeeks.org/introduction-to-recurrent-neural-network/.

[31] S. Paul. "Hyperparameter optimization tuning for machine learning (ml)." [Accessed 28-09-2023], datacamp.com. (2018), [Online]. Available: https://www.datacamp.com/tutorial/parameter-optimization-machine-learning-models.

[32] J. Kreiger. "Evaluating a random forest model." [Accessed 28-09-2023], medium.com. (2020), [Online]. Available: https://medium.com/analytics-vidhya/evaluating-a-random-forest-model-9d165595ad56.

[33] A. Bhandari. "Understanding interpreting confusion matrix for machine learning (updated 2023)." [Accessed 28-09-2023], analyticsvidhya.com. (2023), [Online]. Available: https://www.analyticsvidhya.com/blog/2020/04/confusion-matrix-machine-learning/.

[34] S. Pandian. "K-fold cross validation technique and its essentials." [Accessed 28-09-2023], analyticsvidhya.com. (2022), [Online]. Available: https://www.analyticsvidhya.com/blog/2022/02/k-fold-cross-validation-technique-and-its-essentials/.