Trinity College Dublin
Coláiste na Tríonóide, Baile Átha Cliath
The University of Dublin

Management Science and Information Systems Studies

Final Year Project Report

Usage of AI Techniques to Resolve Dublin Bikes Distribution Issue

Rhona Digan

March 2018

DECLARATION

I declare that the work described in this dissertation has been carried out in full compliance with the ethical research requirements of the School of Computer Science and Statistics.

I have read and I understand the plagiarism provisions in the General Regulations of the University Calendar for the current year, found at: http://www.tcd.ie/calendar

I have also completed the Online Tutorial on avoiding plagiarism 'Ready, Steady, Write', located at

http://tcd-ie.libguides.com/plagiarism/ready-steady-write

I declare that the report being submitted represents my own work and has not been taken from the work of others save where appropriately referenced in the body of the assignment.

Signed: _____

Rhona Digan
23/3/2018

ABSTRACT

The aim of this project is to propose a method to increase the efficiency of the dublinbikes bike sharing system using artificial intelligence techniques. The suggested method involves building a dataset using live dublinbikes data as well as a range of other data sources. Various predictive models were applied to this dataset. These models were evaluated using a variety of performance metrics and the most suitable model was selected. The model should enhance the bike distribution process and thus, increase the efficiency of the system.

PREFACE

This project was undertaken on behalf of Accenture. Accenture are interested in the development of a system to solve the current rebalancing issue experienced by dublinbikes. Rebalancing is the problem of redistributing the bikes within a network in order to avoid stands becoming full or empty and thus, to optimize user experience. Currently there is no model in place to regulate the redistribution of the bikes in the dublinbikes system. This results in an inefficient system where there are often full or empty bike stands at times of high demand.

Creating a rebalancing system involves two steps:
1. Predicting the demand for bikes and free spaces in the docking stands.
2. Selecting the most efficient way to relocate the bikes in order to best satisfy such a demand.

The main problem encountered was a lack of appropriate data was available. Unlike, previous studies done on bike sharing systems, no GPS data was available for dublinbikes. This means there was no way of tracking the location of each specific bike. As a result of this the project only focuses on (1) predicting the demand for bikes and free spaces in the docking stands. In order to predict the demand various predictive models were explored.

I would like to thank Darren Donoghue of Accenture for his assistance throughout the project. I would also like to thank my supervisor, Fergal Shevlin for his guidance and advice over the course of the project. I would finally like to thank the residents of 31 Lansdowne Park for their continuous support and encouragement during this project.

ACCENTURE
Usage of AI Techniques to Resolve Dublin Bikes Distribution Issue

23RD March 2018

TABLE OF CONTENTS

APPENDICES

1. INTRODUCTION

This section introduces the client, discusses the project background, states the terms of reference and summarises all of the subsequent chapters.

1.1 The Client

Accenture is a multinational consultancy firm, offering management consultancy, technology, outsourcing and analytics services to clients across the globe. Accenture's global headquarters are based in Dublin along with Accenture's Analytics Innovation Centre; designed to help accelerate the development and delivery of innovative analytics solutions. Accenture is a Fortune Global 500 company employing more than 323,000 people in 200 cities across 56 countries. Accenture analytics offers clients solutions to data, technology, and business challenges. Their comprehensive approach to analytics is fuelled by their deep industry knowledge, broad functional experience, and technology expertise.

1.2 Project Background

The following project background was provided by the client, "Machine learning and advanced analytics are hot topics in today's business environment, as is the rise of dynamic data visualisation. The value of advanced analytics gives the business insights that would have been previously unreachable. Data Science can provide solutions to many of the world's problems such as transport which is what we will attempt to do here". The aim of this project is to use machine learning and advanced analytics to improve the efficiency of the dublinbikes bike sharing program.

Over 400 cities have operating bike sharing programs world-wide (wikipedia.org, 2018). Bike sharing systems are made up of a network of bike docking stands which allow users to take a bike from any stand and return it to any stand. The bikes are redistributed between stands using vehicles which carry multiple bikes. The main challenge for a bike sharing system is rebalancing the system. This is the redistribution of bikes within the network in order to avoid stands becoming full or empty. The design of an efficient rebalancing scheme has two major challenges: (i) predicting the demand for bikes and free spaces in the docking stands, and (ii) dictating the movements of the vehicles to relocate the bikes efficiently. This project focuses on the prediction of demand for bikes for the dublinbikes system. The vehicle routing problem cannot be addressed in this project as there is no data or information available on distribution vehicles. There is no rebalancing model currently in place for dublinbikes.

1.3 Terms of Reference

<u>Original Terms of Reference</u>

- Investigate the current situation of how bikes are moved around the city.
- Create a database from the live data.
- Observe the distribution/slopes of the bikes coming into or out of the bike racks and try and spot inconsistencies in the distribution as the racks become full or empty.
- Use these distributions/slopes to predict the length of the queue and length of time people are queuing for at each bike stand.
- Build multiple models to map the movements of bikes around the city.
- Compare the models and pick the best one.

Amended Terms of Reference

Throughout the course of the project the terms of reference were amended. Due to the limited data available it was not possible to predict the length of time people queue for bikes or build models to map the movements of the bikes around the city.

The following terms of reference were amended:

- Use these distributions/slopes to predict the length of the queue and length of time people are queuing for at each bike stand.
- Build multiple models to map the movements of bikes around the city.

It was decided the project would focus solely on predicting the demand for bikes within the dublinbikes network. Two new terms of reference were decided with the client:

- Use the historical bike data to find a method to calculate demand for bikes and free stands.
- Use this demand variable to predict future demand for bikes and free stands at any time for any stand.

1.4 Summary of Chapters

- Chapter 2: Recommendations and Conclusions: Outlines the overall recommendations and conclusions of the project.
- Chapter 3: Project scope and Background: Defines the problem, describes background research and outlines the approach taken.
- Chapter 4: Data Collection, Preparation and Exploration: Describes how the dublinbikes data was collected, compiled and initial exploration of the data.
- Chapter 5: Target Variable: Describes how the dublinbikes data was transformed into the net demand variable using regression, forecasting and clustering techniques.
- Chapter 6: Predictor Variables: Describes each predictor variable and how they were obtained, compiled and their limitations and assumptions. It describes how each variable is linked to the predictor variable and how the final dataset was collated in a suitable way for predictive modelling.
- Chapter 7: Predictive modelling: Analyses and compares three different predictive modelling techniques. It describes the best model for each technique evaluates the performance of the models.
- Chapter 8: Model Comparison: Compares the performance of the three models and suggests the final model.

2. CONCLUSIONS AND RECOMMENDATIONS

This chapter outlines the overall conclusions and recommendations of the project.

2.1 Conclusions

The live dublinbikes data was gathered and compiled to form a dataset. Initial analysis of the trends in the data show a major difference between weekday and weekend bike usage patterns, thus, two weekends and weekdays were analysed separately.

Initial analysis also highlighted that there was a high correlation between the patterns on bike stands located close to each other. Therefore, the stands were clustered in groups based on location.

As per the terms of reference, demand for each bike stand was calculated as the target variable. A dataset of predictor variables was created using a range of online resources and data. This was combined with the demand variable to make a useful dataset for predictive modelling. The data was cleaned and split into a training set and a testing set (70% training, 30% testing).

Three predictive models were fitted to the data and the optimal parameters for each were found using the training set. The test set was used to analyse the performance of the models. The Random Forest model was deemed to be the optimal model. It had a correlation of 0.681 and 0.81 for weekday and weekend models respectively. This gives quite an accurate prediction of net demand. The model performed well when predicting small demand values, however, it struggled to predict extreme demands.

The Random Forest model can be used to predict net demand for any station for any hourly time period once the predictor variables are known. It can be used to tell if a stand is likely to become full or empty based on its current state. The dublinbikes vehicles have access to live data informing them of the number of bikes and free stands at all of the bike stands at any given time. If the model predicts the demand for bikes in the next hour will be greater than the current number of bikes at a station, the vehicles can put bikes in this station and vice versa for free stands. This can help avoid a surplus or shortage of bikes and thus, make the system more efficient.

The Random Forest variable importance output highlighted that weather has a significant impact on bike usage. This can be used to adapt the system when during periods of poor weather.

This method is a general method which is not specific to dublinbikes. Once a detailed investigation is done into the factors affecting bike usage, this model can be applied to any bike sharing system.

2.2 Recommendations

It is recommended to implement the Random Forest model with the database that was built throughout this project. To ensure the model is always up-to-date, the database should be

continuously updated. The predicted net demand should be used to inform the vehicles of potential full or empty bike stands so that they can be avoided. This improves on the efficiency of the current system where there is no model in place.

Given the time constraints and limited data available for this project, there are many improvements which could be made to further increase the performance of the suggested model. This could be done by improving the dataset created or, if more data becomes available, investigating other models.

Improving the Dataset

1. Data should be gathered over a longer period of time. This project only deals with 6 weeks of data which is not a sufficient representation of the system for example it is unknown if the system experiences seasonality (variance at different times of the year). Ideally at least a year of data should be used.
2. The predictor variables gathered should be more accurate. There are many assumptions made when collecting the predictor variables. Many of the assumptions could be narrowed with the availability of more detailed data. For example, the affluence variable could consider smaller areas.
3. Multiple datasets could be created with different assumptions for the predictor variables. For example, the walking distance which affects bikes usage could be tested at different values. The models for all datasets could also be compared. This was not possible due to the time constraint on the project.
4. The Variable Importance (Appendix F, p.F.1) could be used to identify the types of variables which have the most influence on net demand. More variables should be gathered and added to the dataset. Variables with little or no influence should be omitted from the dataset.

General Recommendations

5. Many assumptions are made for the purposes of this project. There should be extensive research carried out in order to increase the accuracy of these figures for, example, how far someone is willing to walk to take or return a bike (Section 3.1, p.11).
6. The movements of the redistribution vehicles should be obtained and the corresponding observations should be removed from the data before modelling. As this information is not available for this project, a reasonable guess is made to which demands are caused by the redistribution vehicles. If this information was available the net demands caused by the vehicles could be removed. The model should then perform significantly better.
7. As outlined in the literature review (Appendix C, p.C.1), the use of GPS data is essential in creating accurate dynamic solutions. As GPS data for dublinbikes is currently unavailable, this project suggests a general solution which can be applied to any bike sharing system. However, in order to build a specific solution for dublinbikes, GPS tracking should be installed in the bikes and the methods suggested in the literature review should be explored and compared with this model.

3. PROJECT SCOPE AND BACKGROUND

This chapter describes the problem, background research and the approach taken.

3.1 Problem Definition

There is no model currently in place to dictate the movements of the bikes within the dublinbikes system. The dublinbikes scheme suffers from two availability problems: (i) having bikes when and where people want to take them; (ii) having parking bays available when and where people want to leave them. Consequences of these problems include wasted time waiting and inefficient use of the bikes ultimately leading to reduced mobility around the city. In order to solve this problem, it is essential to predict the demand for taking bikes and available parking stands. A model should be created to predict these respective demands for the dublinbikes network.

3.2 Background Research

In order to better understand the current situation of the dublinbikes network, background research was conducted. This was carried out through desk research and field research.

Desk Research

The following information was available online (Dublinbikes.ie, 2018).

- Dublinbikes has 101 stands across Dublin City Centre.
- Each stand has a between 15 and 40 parking bays.
- The stands are open to take bikes from 05:00-00:30 and bikes can be returned at any times.
- Dublinbikes is run by the marketing company JCDeceaux in return for advertising on the bikes. The official name for dublinbikes is Just Eat dublinbikes. It is referred to ‘dublinbikes’ for the purposes of the project.
- An application is available on iOS and android to show users how many bikes and free parking stands are available at each station.

Field Research

The bikes system was used and observed by the author over the course of the project.

- There are a number of vehicles used to move the bikes form one stand to another in order to manage supply and demand. Each vehicle has the capacity to hold 18 bikes.
- During peak usage hours, there were often full bike stands with additional bikes chained to nearby railings. This leads to users are paying for time when they are not using the bikes, thus, decreasing customer satisfaction and the efficiency of the system.

Literature Research

A literature review was carried out on Rebalancing Bike-Sharing Systems. The following were the main conclusions:

- There are two types of solutions to the rebalancing problem – static cases and dynamic cases. The static case involves rebalancing the bikes when the system is closed i.e. users cannot interact with the system during rebalancing. The dynamic case involves rebalancing the bikes when the system is open.

- There is extensive literature available surrounding static rebalancing. However, there is limited literature surrounding dynamic rebalancing. This is unfortunate as the dublinbikes problem is a dynamic rebalancing problem.
- Many studies have been conducted to find the most efficient rebalancing solution. Some of these solutions include forecasting, networking graphs and nearest neighbour algorithms.
- GPS tracking data is essential in order to apply any of the suggested solutions in the literature. As GPS data is not available for dublinbikes, none of these solutions are feasible for the dublinbikes problem. New approaches should be considered.

3.3 Approach

After the initial research, it was clear that there is a problem with supply and demand of the bikes not matching up. In order to find a solution, three approaches were considered:

1. Networking - Networking has been used to as an approach to bike sharing problems such as 'Volcanoes and Black Holes Detection model' for the New York citibike bikes problem (Hong et al., 2015). After further examination, it was discovered that networking would not work for this problem. In order to make a network, the GPS data tracking the movements of each bike would be required. The dublinbikes data only provides data on how many bikes are at each station at a given time.
2. Predictive modelling – This is not possible with the original dublinbikes data as there are only three variables that change in each observation; time, available bikes and free stands. Available bikes and free stands are exactly correlated (Section 5.1 p.10) so this only leaves two relevant variables. It is not possible to build a predictive model with two variables.
3. Supply and demand model – Basic supply and demand models were investigated. However, these would not satisfy the client requirement of using machine learning techniques.

The approach taken was to use predictive modelling by gathering independent variables based on external factors which affect bike usage. A suitable dataset was compiled. The calculation of the target variable, net demand, is described in Chapter 5. The collection of the predictor variables is described in Chapter 6. The model is used by inputting the predictor variables and it will output net demand for that stand for that hour. Multiple prediction models were built. These models were compared and the optimal one was chosen as the final model. This satisfies the main client requirement of the project to use machine learning techniques in an attempt to improve efficiency of the dublinbikes system.

4. DATA COLLECTION, PREPARATION AND EXPLORATION

This chapter presents an outline of the data, describes how the data was gathered and a description of the initial exploration of the live data.

4.1 Data Outline

Two types of data are provided by JCDecaux which is available online (Developer.jcdecaux.com, 2018) in the following format:

1. Static Data

Table 4.1.1 - Description of Static Data

Variable name	**Description**
station_name	Name of the bike stand
station_id	ID of the bike stand
longitude	Longitude of bike stand
latitude	Latitude of bike stand

2. Live Data

Table 4.1.2 - Description of Live Data

Variable name	**Description**
station_name	Name of the bike stand
station_id	ID of the bike stand
longitude	Longitude of bike stand
latitude	Latitude of bike stand
status	Indicates whether the stand is open or closed
bike_stands	Number of operational stands at the station
available_bike_stands	The number of available bike stands at this station
available_bikes	The number of available bikes at this station
last_update	The last update time in milliseconds since Epoch

4.2 Data Collection

Static data:

The static data was downloaded as a csv file.

Live Data:

To access the live data, an account was made on the JCDecaux website which provided an API key which gave unlimited access to the data. The data is public data and the disclaimer states that it can be compiled, adapted and used for any purpose.

Both Python and R were considered as platforms for gathering the data. Python was chosen over R because it has a wider range of packages for running repeated tasks and is faster for processing large amounts of data.

The data was collected as follows:

1. A function was built to get the data from the API using the *urllib*() package.
2. A second function was built to continuously run the program every minute using the *scheduler*() package.
3. The data was compiled on a weekly basis as a csv file.

4.3 Data Exploration

After one week of data was collected, a sample dataset was compiled for initial exploration of the data. A time series was made for each stand to show available bikes and free stands over the week. Two types of patterns were observed:

- Pattern 1: Stands with a large number of free stands in the morning and a large number of available bikes in the evening.
- Pattern 2: Stands with a large number of available bikes in the morning and a large number of free stands in the evening.

Weekends did not have any noticeable patterns.

Figure 4.3.1 shows a time series graph for bike stands 13 (Fitzwilliam Square) and 43 (Portobello).

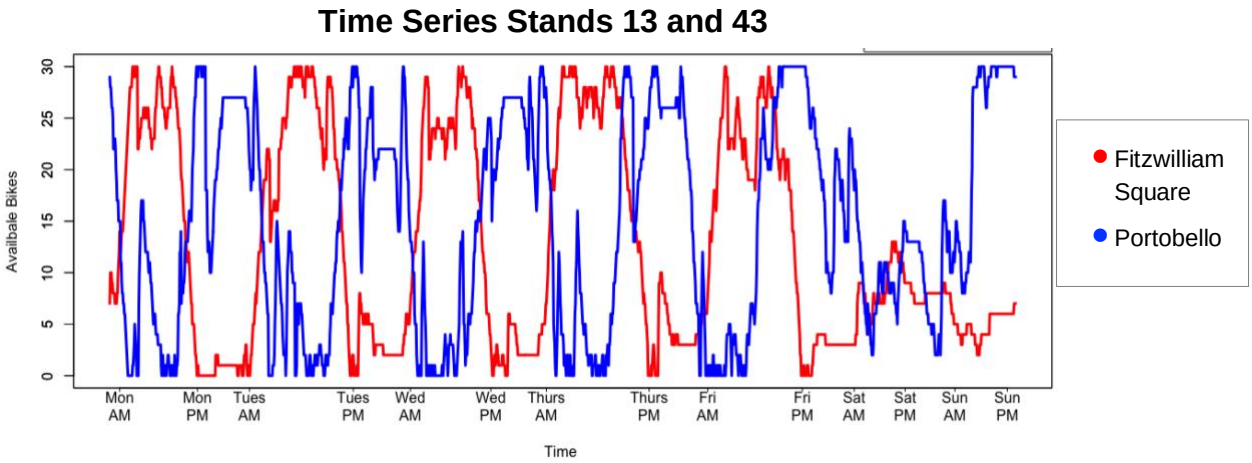


Figure 4.3.1 - Time Series of Contrasting Stands

Figure 4.3.1 shows a seasonal daily pattern (Pattern 1) for Fitzwilliam Square. As Fitzwilliam Square is a highly concentrated office area this pattern is expected. The opposite pattern (Pattern 2) was observed in residential areas such as Portobello. Figure 4.3.1 shows there is no obvious pattern on weekends.

The initial exploration also highlighted the high correlation between stands which are located close to each other. Stands that are closely located tend to follow a similar pattern. This can be seen in Figure 4.3.2 which shows the time series graph for stands 56 and 58, which are located 152m apart on Mount St and Sir Patricks Duns (Grand Canal Street) respectively.

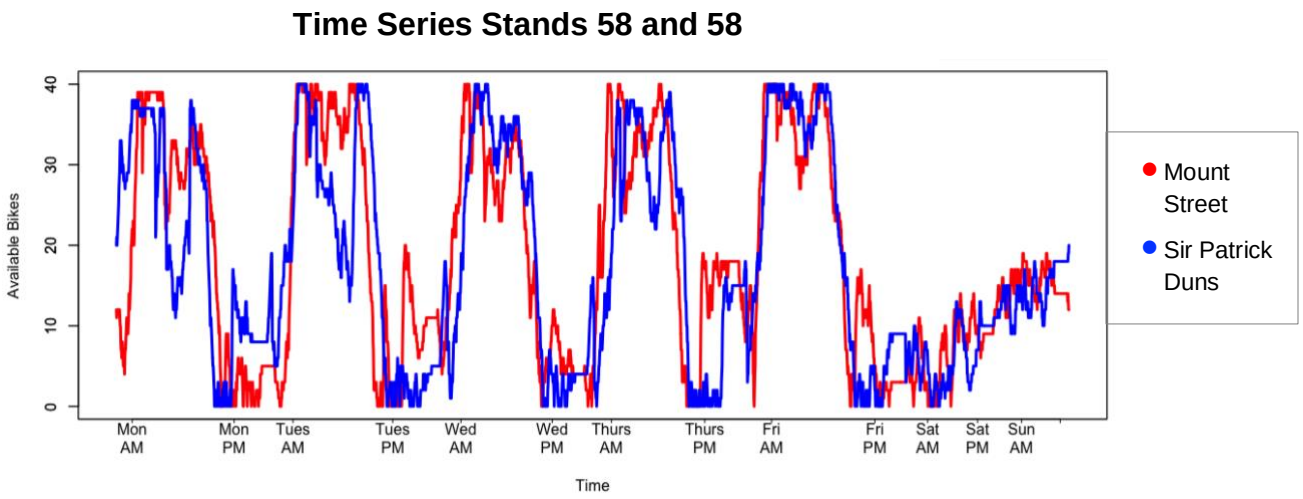


Figure 4.3.2 - Time Series of Similar Stands

4.4 Data Preparation

The scraping program ran for a total of 6 weeks from 17th January 2018 to the 28th February 2018. This time period included no public or bank holidays. This gave approximately 6,000,000 (60 minutes * 24 hours * 7 days * 6 weeks *100 stands) observations. Each weeks dataset was read into R and all 6 were combined using the function *rbind*().

Although the data was collected every minute, the data was only updated on the dublinbikes API every 6 minutes on average. This lead to duplication of rows. After removing these rows, there were approximately 1,000,000 rows remaining.

The data was grouped into hourly time periods. Although this gives a less specific solution, a broader solution is more practical for users. A solution given in minutes introduces a lot of noise into the data. For example, a solution saying the demand for bikes at Grand Canal Dock at 9:17am is 0.85 bikes, is not user-friendly, whereas a solution saying the demand for bikes at Grand Canal Dock between 8am and 9am is 30 bikes is more informative.

When collecting the predictor variables, it is impossible to collect some of them on a minute-by-minute basis so grouping the time periods into hours also makes more sense from this aspect.

5. TARGET VARIABLE

This chapter describes how the variables *available_bikes* and *free_stands* from the live data were transformed into the target variable; net demand.

5.1 Variable Description

A net demand variable was derived from the *available_bikes* variable and the *free_stands* variable. Net demand represents the number of bikes coming in or out of the stand per hour. Both demand for bikes and demand for free stands can be represented by one variable rather than two separate variables because there can only be a demand for one of them i.e. there is only ever a net demand for bikes or for free stands. This is demonstrated in figure 5.1.1 which shows the number of available and bikes in stand 13 (Fitzwilliam Square) for one day.

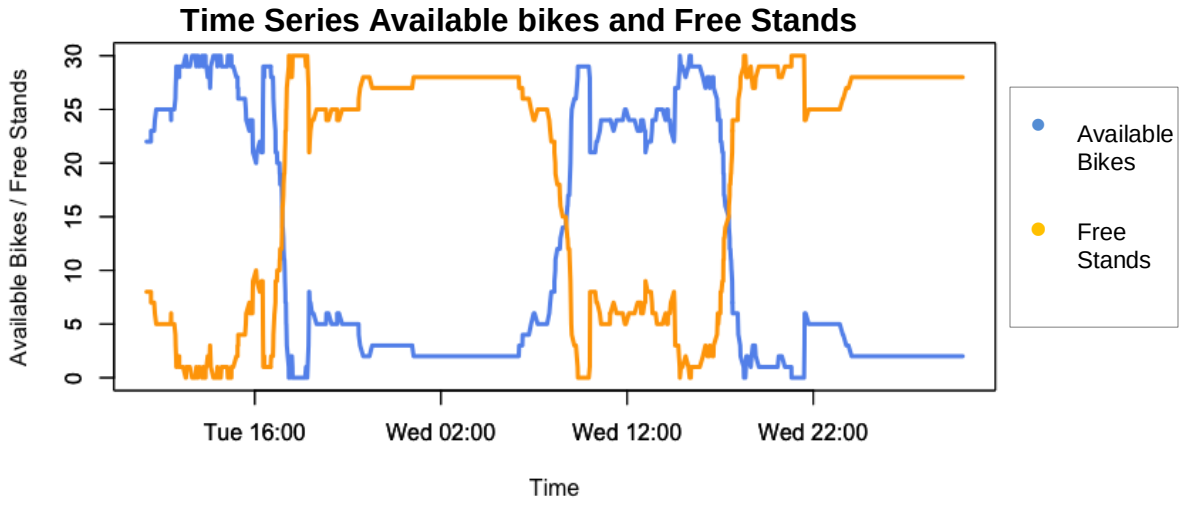


Figure 5.1.1 - Time series of Available Bikes and Free Stands

If net demand is positive it represents a demand for free stands, if it is negative, it represents a demand for bikes, i.e. If 12 bikes leave a stand and 2 arrive in an hour, there is a demand for 10 bikes so, the net demand is -10. For example, in figure 5.1.2 the slope of the line is positive between 7am and 8am which represents a demand for free stands. The slope of the line is negative from 4pm to 6pm which represents a demand for bikes.

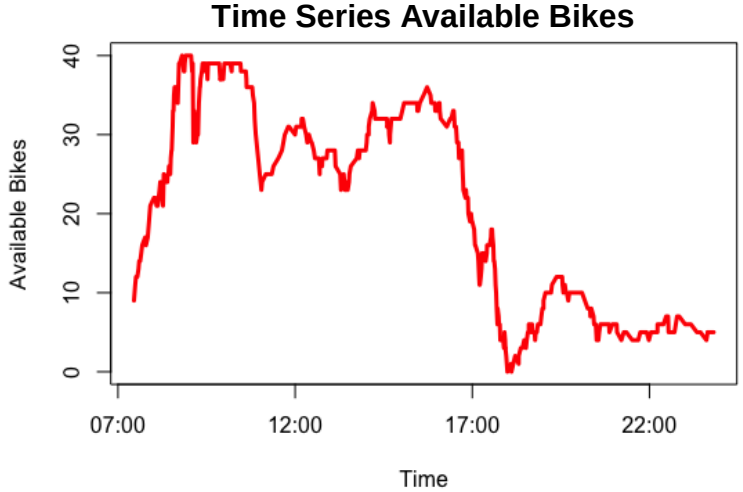


Figure 5.1.2 – Time Series Available Bikes

The number of available bikes at a stand for one hour depends on how many available bikes there were in the previous hour, hence, the observations are not independent. Predictive modelling does not work well when observations are dependent. Therefore, net demand was used as the target variable. Net demand each hour is independent of demand for all other hours i.e. For a given stand, if demand from 8am-9am is 20 bikes per hour, and demand from 9am-10am is 5 bikes per hour. These figures are independent of each other as they do not consider how many bikes were at the stand during the previous hour.

The following process was used to arrive at the net demand figure:

- Step 1. Cluster nearby bike stands.
- Step 2. Calculate the net demand figures for each stand at each point in time.
- Step 3. Forecast the net demand for times when the stand is full or empty.

5.2 Step 1: Clustering

From the initial exploration (Section 4.3, p.8), it is evident that the activity between closely located stands is highly correlated. Highly correlated observations lead to the inclusion of unnecessary data as both variables give the same information. In order to avoid this, the stands were clustered by geographical location. Each cluster was treated as a general area.

The first step in clustering is calculating a dissimilarity matrix with the longitude and latitude of each stand. Hierarchical clustering was used. This is an agglomerative clustering approach which starts from the bottom and clusters each data point (bike stand) in its own cluster. The two stands with shortest proximity (distance) are then combined. Clusters are continued to be combined until there are no possible combinations under a set cut-off point.

A cut-off point of 500 meters was decided. This means that stands within 500 meters of each other can be clustered together but no stands greater than 500m will be part of the same cluster. A recent study conducted on the distances people walk for transport shows that the median distance people walk to and from transport stations is between 470 meters and 600 meters, thus, 500 meters was chosen as the cut-off point (Burke and Brown, 2007). Figures 5.2.1 and 5.2.2 show the results of the clustering, grouped by colours. The available bikes and total stands within each cluster were added together and each cluster was treated as one unit.

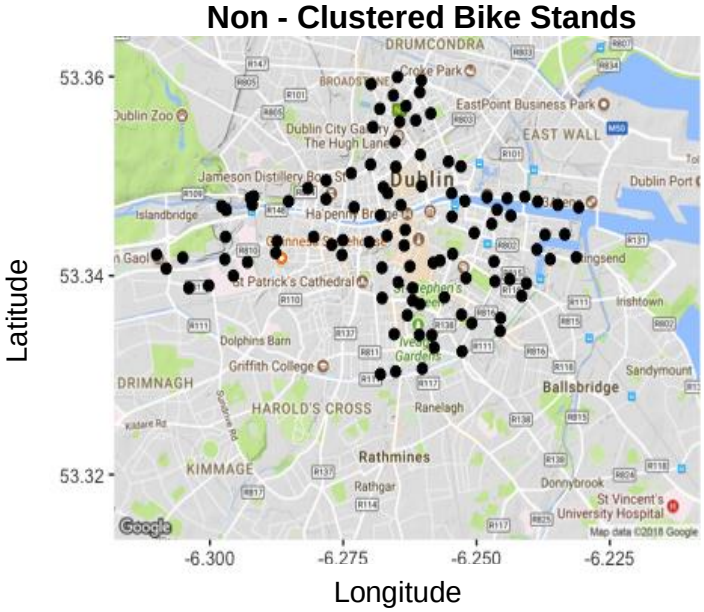


Figure 5.2.2 - Map of Non-Clustered Bike Stand Locations

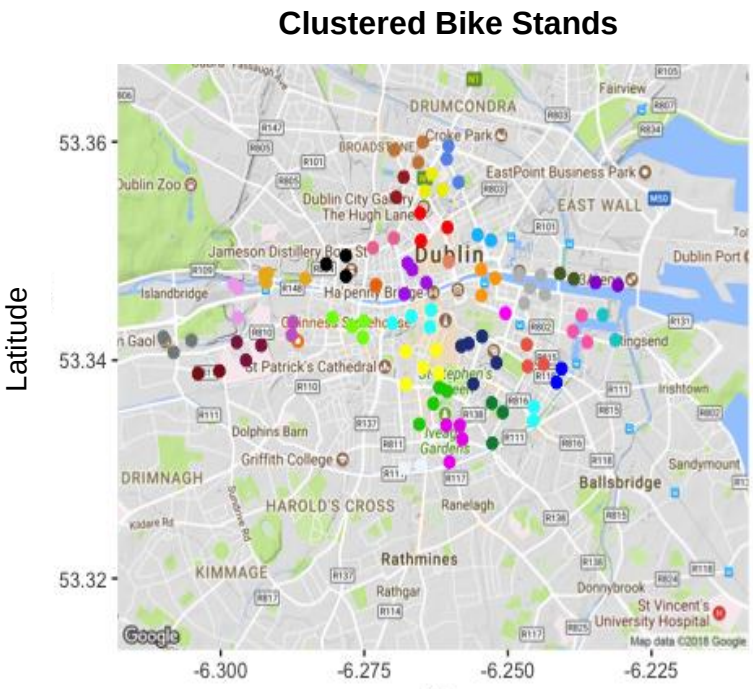


Figure 5.2.1 - Map of Clustered Bike Stand Locations

5.3 Step 2: Calculating Net Demand Figures

The net demand was calculated as the rate of change of available bikes/free stands per hour. A positive net demand equates to demand for free spaces and a negative net demand equates to demand for bikes.

$$Net\ Demand = \frac{Bikes\ in - Bikes\ out}{Time}$$

To calculate the net demand for each data point, a time series of the available bikes for each cluster was made. The time series was smoothed using Lowess (Locally Weighted Scatter-plot Smoother) regression to get rid of irregular roughness and to create a curve as the basis of the slope calculation (Appendix D, p.D.1). Lowess is a non-parametric least square smoothing algorithm. A non-parametric algorithm was used because each different stand follows a different pattern so one distribution cannot be assumed on all of the data

A small smoothing parameter was used to ensure the smoothing was not removing the seasonality of the data. Figure 5.3.1 shows a time series plot and its smoothed curve.

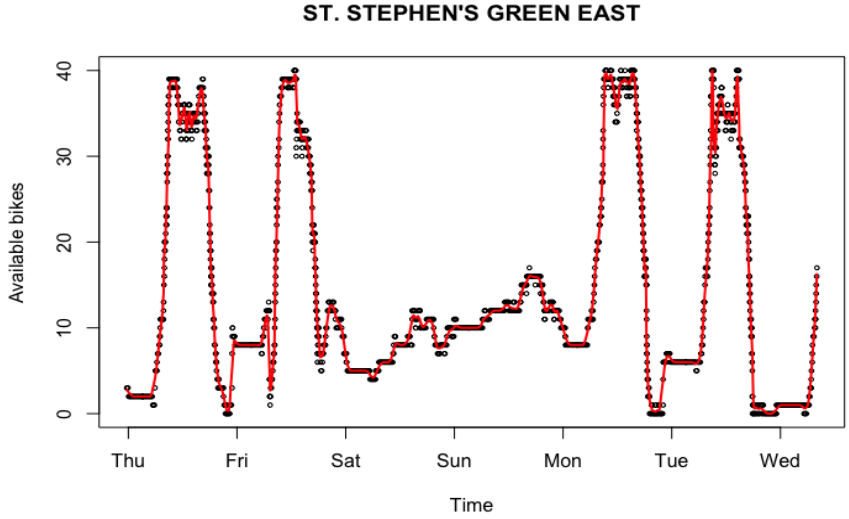


Figure 5.3.1 - Smoothed Time Series Example

To calculate the slope for each observation, a moving average slope was used. This is an average slope of the 5 observations before and after the observation in question. This was calculated for each observation in the dataset to get the final net demand figure.

5.4 Step 3: Forecasting

When the stand is full or empty the net demand will appear to be 0. This is not an accurate reflection of net demand as there may be people wishing to take or return a bike but it is not possible. To overcome this problem time series forecasting was used. A forecast was made of the net demand for time periods when a stand is full/empty based on the previous data in the series. ARIMA(Appendix D, p.D.1) time series forecasting was used.

The function *auto.arima*() in R generates the best ARIMA model to use given the input time series. It does this my minimising the information criterion; AIC and BIC (Appendix D, p.D.1) The information criterion measures the accuracy of the model while penalising it for being over complicated. By minimising the information criterion, we find the correct trade-off between accuracy and complexity of the model.

The demand forecasts were calculated and these replaced the values of net demand when the stands are full or empty.

5.5 Data Cleaning

After calculating the net demand, the data must be adjusted to factor in the redistribution vehicles moving the bikes. When the vehicles collect bikes or fill the stands, there appears to be periods of high demand when, in fact, there is no demand. It is difficult to distinguish this apparent demand from actual periods of high demand; thus, a rough estimate was made.

From the field research (Section 3.2, p.5), it was observed that it takes between 5 and 10 minutes to fill or empty a bike stand. The vehicles can hold 18 bikes. This equates to a net demand greater than 108 or less than -108. It is likely any demand greater than this is caused by the vehicles.

$$Net\ Demand: \frac{\pm 18\ bikes}{10\ minutes} = \frac{\pm 18\ bikes}{0.16667\ hours} \approx \pm 108$$

For simplicity, 108 was rounded to 100 and any net demand greater than 100 or less than -100 was removed. Figure 5.4.1 shows a plot of all the net demand values. All of the values on the outside of the red lines (>100/<-100) were removed.

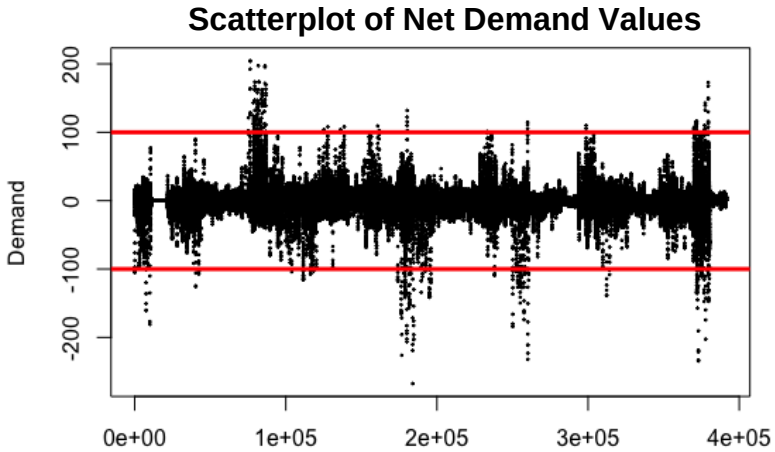


Figure 5.4.1 - Scatterplot of Net Demand Values

6. PREDICTOR VARIABLES

This chapter describes the predictor variables and how they were collected.

6.1 Approach

In order to build a predictive model, a set of predictor variables are needed. These are any variables which could affect the target variable (net demand). A list compiled of factors which may affect dublinbikes usage. These variables were chosen based on their relevance (do they actually affect the net demand for bikes) and if they were possible to gather. With approval from the client 10 predictor variables were selected. Creating these variables involved downloading and cleaning datasets, gathering and collating data, and using online resources and tools. Each variable and how it was created is detailed below.

6.2 Description of Variables

When calculating the predictor variables, it was necessary to define a geographical constant on the area surrounding the bike stand which influences bike usage (area of influence). This distance of influence is 500 meters (see Section 5.2, p.11).

Day

- Description: A categorical variable with 3 levels; Weekday, Saturday or Sunday.
- How to gather: The data was collected using the live data form the dublinbikes API. Time was in epoch time format. This was converted into normal date and time and then categorized into either Weekday, Saturday or Sunday.
- Assumptions/Limitations: Categorising all weekdays together assumes bike patterns are constant across all weekdays. However, for the purposes of simplicity this is an efficient way to categorize the dates.

Time Category

- Description: A categorical variable with 14 levels; before 6am, 7am-8am, 8am-9am, 9am-10am, 11am-12pm, 12pm-1pm, 1pm-2pm, 2pm-3pm, 3pm-4pm, 4pm-5pm, 5pm-6pm, 6pm-7pm, 8pm-9pm, after 9pm.
- How to gather: The data was collected using the live data from the dublinbikes API. This was in epoch time format. This was converted into normal date and time and then categorized into its various levels.
- Assumptions/Limitations: Grouping the time by hour assumes the activity within each hour is constant. This allows for variance from day to day.

Rain

- Description: A continuous variable stating the amount of rain in mm during that hour.
- How to gather: Historical data was downloaded from Met Eireann (Met.ie, 2018). Each observation in the main dataset was matched which the corresponding date and time in the weather dataset to give the amount of rainfall during that time period.
- Assumptions/Limitations: This assumes the same weather over the whole region. It also assumes the level of precipitation was constant over the whole hour.

Temperature

- Description: A continuous variable stating the average temperature in Degrees Celsius for that hour.

- How to gather: Historical data was downloaded from Met Eireann (Met.ie, 2018). Each observation in the main dataset was matched which the corresponding date and time in the weather dataset to give the temperature for that time period.
- Assumptions/Limitations: This assumes the same temperature over the whole region. A constant temperature is assumed over the whole hour.

Luas

- Description: A continuous variable of the average number of luas' that stopped within a 500m radius of each cluster per hour.
- How to gather: Google maps was used to get the longitude and latitude of each luas station in the region served by Dublinbikes. Online timetables were used to find out how many luas' stopped at each station every hour (Luas.ie, 2018). The distance between each luas stand and each bike station were compared and if they were within the area of influence, the number of luas' at that stop per hour was recorded.
- Assumptions/Limitations: 500m was assumed as the area of influence.

Dart

- Description: A continuous variable of the average number of darts that stopped within a 500m radius of each cluster per hour.
- How to gather: Google maps was used to get the longitude and latitude of each DART station in the region served by Dublinbikes. Online timetables were used to find out how many darts stopped at each station every hour (Rail, 2018). The distance between each dart station and each bike stand were compared and if they were within the area of influence, the number of darts at that stop per hour was recorded.
- Assumptions/Limitations: 500m was assumed as the area of influence.

School

- Description: A numeric variable of the number of pupils attending school within a 500m radius of the cluster.
- How to gather: Google maps was used to get the longitude and latitude of every school in the region served by dublinbikes. The department of education website was used to find out how many pupils attend each school (Department of Education and Skills, 2018). The distance between each bike stand and each school were compared and if they were within 500m the area of influence, the number of pupils in that school was recorded.
- Assumptions/Limitations: Only secondary schools were considered as it is assumed that primary school students are too small to use dublinbikes. The figure 500m was assumed as the area of influence. University students were also considered but accurate data was not available.

Affluence

- Description: A continuous variable of the average rent price in each area.
- How to gather: Google maps was used to find which area of Dublin each bike stand is in. A Daft.ie report lists the average rent price for each Dublin area (Daft.ie, 2018).
- Assumptions/Limitations: The regions used by Daft.ie are quite large and therefore generalisations had to be made.

Residential Units

- Description: A numeric variable of the number of residential units within a 500m radius of the centre of each cluster.
- How to gather: The centroid of each cluster was calculated. The online geodictionary tool was used to get the number of residential units within a 500m of this point (Geodictionary, 2018).
- Assumptions/Limitations: 500m was assumed as the distance of influence.

Commercial Units

- Description: A numeric variable of the number of commercial units within a 500m radius of the centre of each cluster.
- How to gather: The centroid of each cluster was calculated. The online geodictionary tool was used to get the number of commercial units within a 500m of this point (Geodictionary, 2018).
- Assumptions/Limitations: 500m was assumed as the distance of influence.

6.3 Variables Considered but not Included

- Bus: The number of buses that stop the area of influence of each bike stand. This was not possible to obtain due to the large concentration of bus stops in Dublin city.
- Event: Whether there is a large event on within the area of influence. This was not included because it was difficult to determine what size event cause an impact. Within the 6 weeks of data used, there were not many large events, however, if the study was being done over a longer period this should be included.
- Population: The population density within each cluster. This could not be determined for the small cluster areas.
- Gradient: Whether the area surrounding the bike stand is flat, on top of a hill or bottom of a hill. This is not relevant to Dublin as the dublinbikes region is flat,1` but if being applied to another city this would be relevant.

6.4 Compiling the Dataset

When the data for each variable was collected, it was read into R, cleaned and formatted. Each observation initially contained the cluster number, time and net demand. For each observation, each predictor variable was added until all the information had been filled in. Net demand for one hour is independent of net demand from the previous hour, therefore, each observation was treated as an independent observation. This gave a dataset with 1,000,000 rows of 11 variables. Table 6.4.1 shows a sample of the dataset.

Table 6.4.1 – Sample dataset

demand	rain	temp	day	luas	dart	school	affluence	residential units	commercial units	time cat
1.0129	0	4.6	Midweek	24	0	246	1873.6	4703	643	7am - 8am
2.2591	0	10.6	Midweek	65	0	372	2332.4	3258	2212	11am - 12pm
-0.5236	0	7.5	Midweek	74	0	618	2332.4	3959	637	2pm - 3pm
-1.8032	2.8	5.2	Sat	80	0	1818	2332.4	3793	1062	2pm - 3pm
-0.6426	0	2.5	Midweek	16	0	1904	2466.2	1169	1753	after 9pm

Target Variable

Predictor Variables

7. PREDICTIVE MODELLING

This chapter outlines the implementation of Single Tree, Boosting and Random Forest models.

7.1 Partitioning the Data

The data was partitioned into two sets; training data and testing data. The training data was used to create the model and the testing data was used to evaluate the performance of the model. This ensures that reliable accuracy metrics can be produced with data the model has never seen before. The ratio of the training/testing split should be considered. With less training data, the parameter estimates have greater variance. With less testing data, the performance statistics have a greater variance. It was decided that a 70:30 split between training and test respectively was the best balance between these competing objectives.

7.2 Data Preparation

In the exploratory analysis, it was discovered that bike usage patterns are different on the weekends and weekdays. Therefore, it was decided to run two separate models for weekends and weekdays. This gives a greater accuracy when predicting net demand.

7.3 Measuring Model Performance

To measure the performance of the predictive models, the following methods were used:

- RMSE: The RMSE (Root Mean Squared Error) is the square root of the average sum of squared errors of the distance between the predicted and actual values. The model with lowest RMSE has the best performance.
- Mean absolute error: MAE is the average error of the absolute differences between predicted and actual values. The model with lowest MAE has the best performance.
- R-squared: R-squared measures how close the data would be to a fitted regression line. It is calculated as: Explained variation / Total variation. It is measured as a percentage between 0nd 100 0 percent indicates that the model explains none of the variability of the test data around its mean. 100 percent indicates that the model explains all the variability of the test data around its mean. The higher the R-squared, the better the model fits your data.
- Correlation: Using the test data, a set of predicted values are calculated. The correlation between the predicted values and the actual values of the target variable is calculated. The higher the correlation, the better the performance of the model. A plot of predicted versus actual values gives a visual representation of correlation. If the values appear to be randomly scattered, there is a correlation of 0, if they form a straight line of slope 1, they have perfect correlation.
- Variable Importance: Random Forest models output variable importance showing which variables have the most influence when predicting the outcome variable. It gives two different measures of variable importance. The percentage increase in mean-squared error (MSE) tests to see how much the MSE increases with each variable. A high increase in MSE is expected for very predictive variables. The increase in node purity measures how pure the nodes are at the end of each tree. It tests to see the result if each variable is taken out. A high score means the variable is important.

7.4 Single Tree

Decision trees are supervised learning algorithms that recursively split the data on a single variable at a time in order to best model the target variable. This leads to a simple model which consists of a series of binary splits until a leaf node with the final prediction is reached.

Weekday Model

The tree was built with a complexity parameter of 0.0001 (Appendix D, p.D.1). Figure and table 7.4.1 illustrate the model performance by plotting actual against predicted values for the test set.

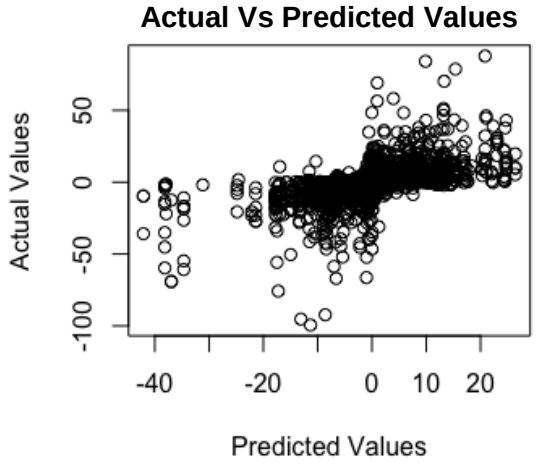


Figure 7.4.1 - Actual Vs Predicted Values for Weekday Single Tree Model

Table 7.4.1 – Performance of Weekday Single Tree Model

RMSE	Correlation
3.579	0.552

A correlation of 0.552 indicates that this model is slightly better than random prediction (correlation of 0.5). It struggles to predict the extreme values.

Weekend Model

The tree was built with a complexity parameter of 0.0001. Figure and table 7.4.2 illustrate the model performance by plotting actual against predicted values for the test set.

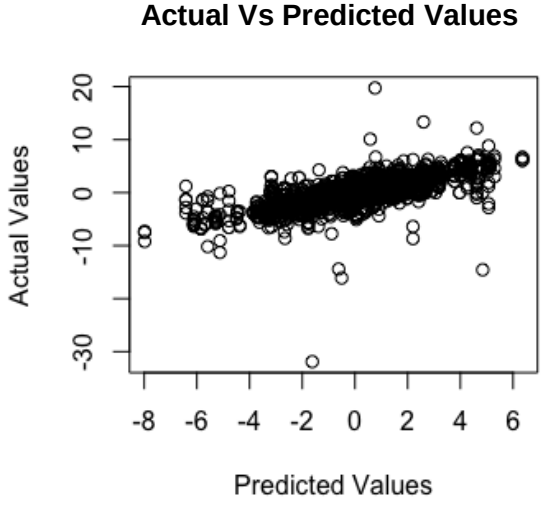


Figure 7.4.2 - Actual Vs Predicted Values for Weekend Single Tree Model

Table 7.4.2 – Performance of Weekday Single Tree Model

RMSE	Correlation
1.153	0.718

This model performs notably better than the weekday model. A correlation of 0.718 denotes the model performs a lot better than random prediction.

The Single Tree models are not examined in great detail as they are used as a benchmark model. The following ensemble methods combine many Single Trees to create a model with greater performance.

7.5 Boosting

Boosting is a sequential process which builds many trees in an additive manner. Each tree is built on weighted versions of the data. More weight is given to observations which were misclassified in earlier rounds. The final prediction is a weighted sum of the predictions of each tree.

An extreme Gradient Boosting model was built using the *xgboost*() function in the *xgboost* package. In order to find the optimal model, it was run with different parameter values for *nrounds* (the number of trees used in the model) and *eta* (the shrinkage parameter - controls the learning rate) and *max_depth* (the maximum depth of each tree in the model).

Weekday Model

To find the optimal number of iterations the initial model was run with 10000 iterations. Appendix shows the RMSE for each iteration. The RMSE drops dramatically until approximately 1000 Boosting iterations where it levels off.

Values for shrinkage and max tree depth were tuned and the RMSE for each value was calculated using 5 - fold cross-validation. Figure 7.5.1 shows the parameter values for each model and the corresponding RMSE (Appendix E, p.E.4). 1000 Boosting iterations were used.

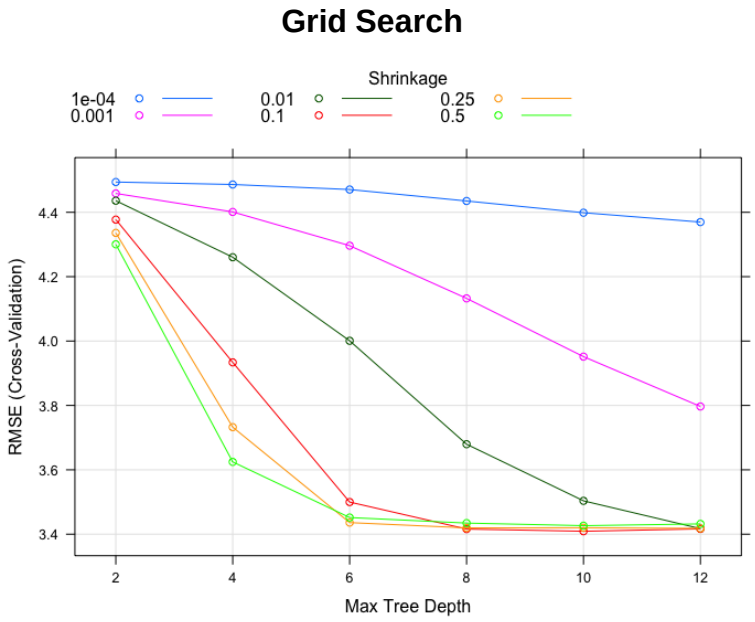


Figure 7.5.1 - Grid Search Weekday Boosting Model

The model with lowest RMSE has parameter values of:

- *Nrounds* = 1000
- *eta* = 0.1
- *max_depth = 10*

The performance of the model was evaluated using the test set. Figure 7.5.2 and table 7.5.1 evaluate the performance of the model.

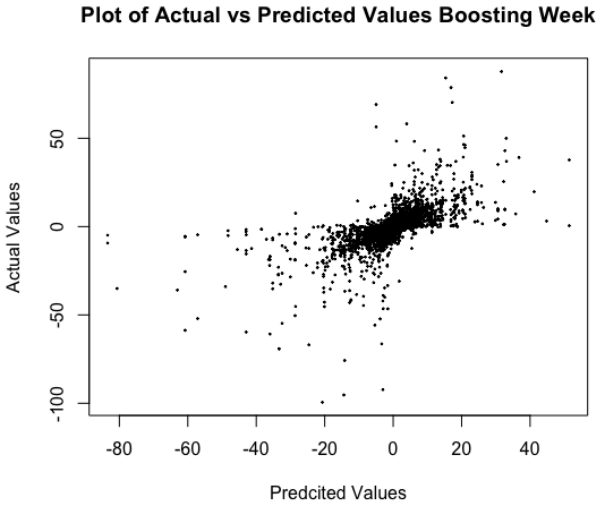


Figure 7.5.2 - Actual Vs Predicted Values Boosting Weekday Model

Table 7.5.1 – Performance of Weekday Boosting Model

RMSE	R-squared	MAE	Correlation
3.237	0.469	0.937	0.663

The model struggles to predict the extreme demands (very high or very low) but it performs well when predicting smaller net demand values. It performs significantly better than the Single Tree benchmark model with correlation improving from 0.552 to 0.663. This shows how multiple trees combined outperforms one Single Tree.

Weekend Model

In order to find the optimal number of iterations the model was initially run with 10000 iterations. Appendix shows the RMSE for each iteration. The RMSE falls with each iteration until approximately 3000 Boosting iterations where it levels off.

Values for *eta* and *max_depth* were tuned and the RMSE for each value was calculated using 5-fold cross-validation. Figure 7.5.3 shows the parameter values for each model and the corresponding MSE (Appendix E, p.E.5). 3000 Boosting iterations were used.

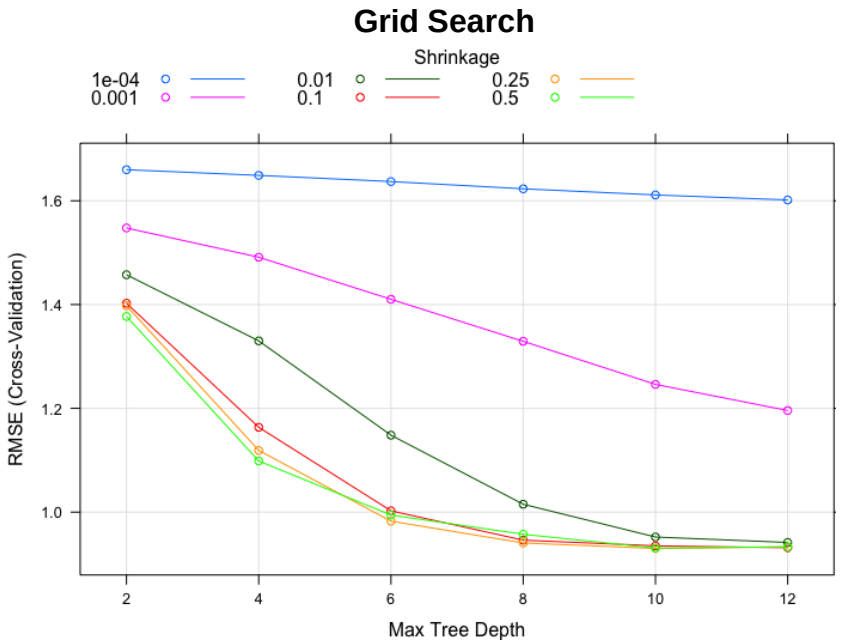


Figure 7.5.3 - Grid Search Boosting Weekend Boosting Model

The model with lowest RMSE has parameter values of:

- *nrounds* = 3000
- *eta* = 0.25
- *max_depth* = 10

The performance of the model was evaluated using the test set. Figure 7.5.4 and table 7.5.2 show the actual test values plotted against the predicted test values.

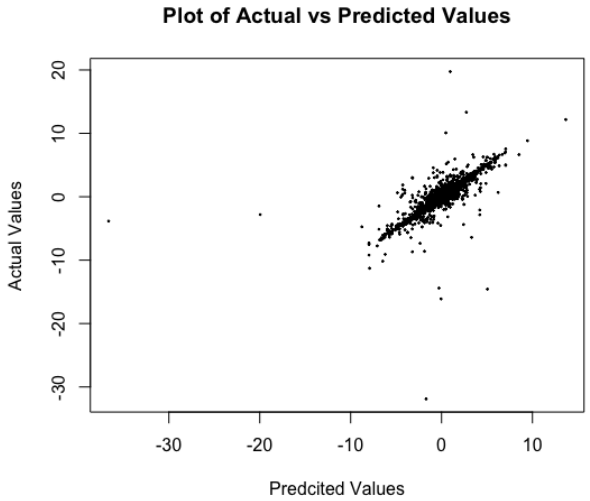


Figure 7.5.4 - Plot of Actual Vs Predicted Values Boosting Weekday Model

Table 7.5.2 – Performance of Weekday Boosting Model

RMSE	R-squared	MAE	Correlation
0.983	0.641	0.274	0.815

This model performs significantly better than the weekday model which can be seen by comparing figure 7.5.6 and 7.5.3. There is a lot less variance experienced on the weekend which is likely the main contributing factor to the improved performance. It also improves on the Single Tree benchmark model.

7.6 Random Forest

Random Forest is a supervised learning algorithm which fits many decision trees to the data. For regression problems (target variable is continuous), the mean of the individual tree predictions is used to give an overall prediction. Each tree is differentiated from the rest by being trained on a random subset of the predictor variables and a random subset of the rows in the data. This works well because a single decision tree may be prone to a noise, but aggregate of many decision trees reduces the effect of noise giving more accurate results.

Random Forest was run using the function *randomForest*() in the randomForest package. In order to find the optimal model, it was run with different parameter values for *ntrees* (the number of trees used in the model) and *mtry* (the number of variables randomly sampled as candidates at each split).

Weekday Model

The percentage error for different number of trees was explored (Appendix E, p.E.2). The percentage error decreases rapidly until approximately 100 trees where it levels off at an error of approximately 10.5. This means there is no need for more than 100 trees as the added complexity of the additional trees outweigh the marginal decrease in the error.

Values for mtry were tuned and the RMSE for each value was calculated using 5-fold cross-validation. Figure 7.6.1 shows the parameter values for each model and the corresponding RMSE. 100 trees were used.

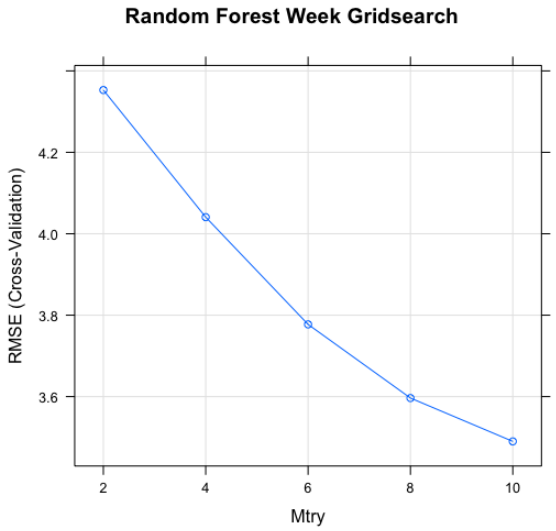


Figure 7.6.1 - Grid Search Random Forest Weekday Model

The model with lowest RMSE has parameter values of:

- *ntrees* = 100
- *mtry* = 10

The performance of the model was evaluated using the test set. Figure 7.6.2 and Table 7.6.1 show the actual test values plotted against the predicted test values.

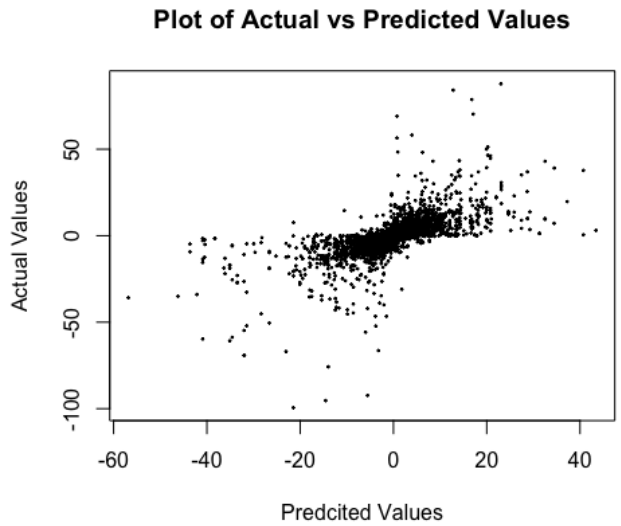


Figure 7.6.2 - Plot of Actual Vs Predicted Values Random Forest Weekday Model

Table 7.6.1 – Performance of Weekday Random Forest Model

RMSE	R-squared	MAE	Correlation
3.15	0.481	0.966	0.681

Similarly, to the Boosting model, this model struggles to predict the extreme demands (very high or very low). It has a lower RMSE and R-squared value than the Boosting model but has a higher MAE. This suggests it captures more of the variance of the data.

Variable Importance Analysis

The variable importance output by the Random Forest was analysed (Appendix F, p.F.1) The variables with the most influence on predicting net demand are temperature and time category. It is not surprising that time category has a strong predictive power because the initial analysis (Section 4.3, p5) highlighted the seasonal nature of the data. It is useful to note the significant impact of the temperature has on bike usage.

Weekend model

The percentage error for different number of trees was explored (Appendix E, p.E.1). The percentage error decreases rapidly until approximately 150 trees where it levels off at an error of approximately 10.5. This means there is no need for more than 150 trees as the added complexity of the additional trees outweighs the marginal decrease in the error.

Values for mtry were tuned and the RMSE for each value was calculated using 5-fold cross-validation. Figure 7.6.3 shows the parameter values for each model and the corresponding RMSE. 150 trees were used.

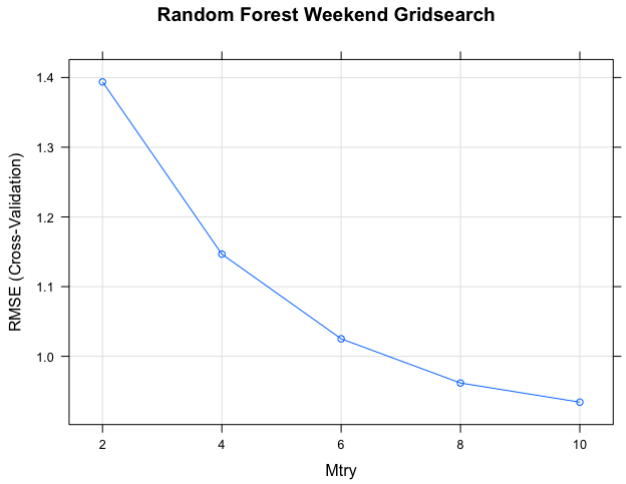


Figure 7.6.3 - Grid Search Weekend Random Forest

The model with lowest RMSE has parameter values of:

- *Mtry* = 10
- *ntrees* = 150

The performance of the model was evaluated using the test set. Figure 7.6.4 shows the actual test values plotted against the predicted test values.

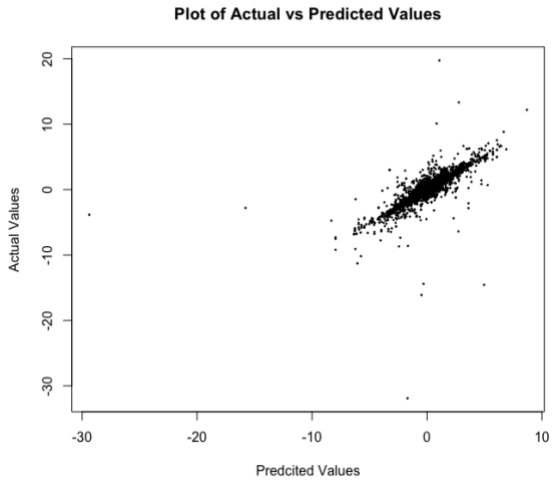


Figure 7.6.4 - Actual Vs Predicted Values Random Forest Weekend

Table 7.6.2 – Performance of Weekend Random Forest Model

RMSE	R-squared	MAE	Correlation
0.965	0.655	0.36	0.81

As with the Boosting model, this model performs significantly better than the weekday model. It has a high correlation of 0.81 and as shown from figure 7.6.6, there are few outliers observed. This also shows the small variance experienced on weekends.

Variable Importance Analysis

The variable importance output by the Random Forest was analysed (Appendix F, p.F.2). The most important variables were affluence, rain and temperature. Unlike the weekday model, time category does not have a high variable importance. This is as expected as the initial analysis (Section 4.3, p5) highlighted the lack of seasonal trend on the weekends. Weather always has a significant impact on bike usage. This information is valuable for dublinbikes, as they could plan for low demand periods if poor weather is forecast. For example, if poor weather is forecast, they could use this time to service the bikes.

8. MODEL SELECTION AND ANALYSIS

This Chapter compares the models, then chooses and analyses the optimal model.

8.1 Comparison of Models

Weekday Model

Table 8.1.1 – Weekday Model Comparison

Model	RMSE	R-Squared value	MAE	Correlation actual and predicted
Single Tree	3.579	0.447	1.517	0.552
Extreme Gradient Boosting	3.340	0.378	0.945	0.662
Random Forest	3.15	0.654	0.966	0.681

All of the weekday models struggle to predict the extreme demands. This may be caused by the redistribution vehicles filling up or removing bikes from the stand. Although the data points with high demands were removed, it is impossible to tell exactly when the observed net demand is caused by a redistribution vehicle or occurs naturally. It is likely that there are still instances included where the high net demands are caused by the vehicles. This would make it difficult for the model to predict high demand instances.

Weekend Model

Table 8.1.2 – Weekend Model Comparison

Model	RMSE	R-Squared value	MAE	Correlation actual and predicted
Single Tree	1.153	0.285	0.641	0.717
Extreme Gradient Boosting	0.982	0.641	0.274	0.814
Random Forest	0.965	0.655	0.359	0.810

A lower variance in net demand observed at the weekend which explains why the weekend model outperforms the weekday model. The reasons for this variance should be examined further, however, some explanations may include:
(i) It is possible that the redistribution vehicles don't work on weekends or there are less vehicles working which also may explain why the weekend model is more accurate.
(ii) There are less extreme demands recorded at the weekend. This is likely a result of offices and schools not being open on the weekends so the high demand periods at the start and end of the working day don't exist.

8.2 Choosing the Optimal Model

Using RMSE as the decision metric, the optimal model for predicting new demand on weekdays and weekends is Random Forest. Both Random Forest models also have the best R-squared values. The weekday Random Forest model has the highest correlation and the weekend model has marginally smaller correlation than the Boosting model. Overall, the Random Forest model performs the best and should be used as the final model.

If the movements of the redistribution vehicles were known, the affected observations affected could be removed from the data, making the model more accurate.

APPENDICES

A. ORIGINAL PROJECT OUTLINE

Client: Accenture
Project: Usage of AI Techniques to Resolve Dublin Bikes Distribution Issue
Location: 1 Grand Canal Square, Dublin 2.
Client Contact: Darren O'Donoghue, darren.odonoghue@accenture.com
Department Contact: Aideen Keaney

Client background:
Accenture is a multinational consultancy firm, offering management consultancy, technology, outsourcing and analytics services to clients across the globe. Accenture's global headquarters are based in Dublin along with Accenture's Analytics Innovation Centre; designed to help accelerate the development and delivery of innovative analytics solutions. Accenture is a Fortune Global 500 company employing more than 323,000 people in 200 cities across 56 countries. Accenture analytics offers clients solutions to data, technology, and business challenges. Our comprehensive approach to analytics is fuelled by our deep industry knowledge, broad functional experience, and technology expertise.

Project background
Machine learning and advanced analytics are hot topics in today's business environment, as is the rise of dynamic data visualisation. The value of advanced analytics gives the business insights that would have been previously unreachable. Data Science can provide solutions to many of the world's problems such as transport which is what we will attempt to do here

Client requirement
The objective of this project is to build a model that will help to identify Dublin bikes' issue with traffic at the bike station and where the bikes should be distributed. This affects many peoples daily routine alongside the health and safety of its users and should also help to reduce costs for the government in distributing the bikes. The key requirements for this are as follows:

- Understand the business problem
- Decide on an Advanced Analytics/ Machine Learning technique (decision trees, regression models, neural networks, network analysis etc.)
- Fully explain in a simple, clear visual manner the complexities of the problem
- The completed project must be groundbreaking and insightful with proven efficiency increase through cost-reduction, queue reduction etc.
- It must clearly explain the complexities of the technique, while also highlighting the value of undertaking an advanced analytics project

What is involved for the student?
The main technical emphasis in this project will be on AI and advanced analytics. The student will need to become thoroughly familiar with standard data science methods such as network analysis which is a key technique for Accenture's future, interpret the client and project requirements and become familiar with the use of languages such as Python or R or alternative languages that they choose. Some knowledge of data analytics techniques is assumed, but a creative approach to the analysis and an ability to relate to the client's perspective on the project will be essential.

B. INTERIM REPORT

Management Science and Information Systems Studies

Project: Usage of AI Techniques to Resolve Dublin Bikes Distribution Issue
Client: Accenture
Student: Rhona Digan
Supervisor: Fergal Shevlin

Review of Background

The Dublin Bikes scheme suffers from two availability problems: (i) having bikes when and where people want to take them; (ii) having parking bays available when and where people want to leave them. Consequences of these problems include wasted time waiting and inefficient use of the bikes ultimately leading to reduced mobility around the city. There may be scheduling and resource allocation techniques already developed for other problems, perhaps already used in the domains of data analytics and machine learning, that could be applied here.

Work to date

- Investigated suitable programming languages and decided to use Python to proceed with the project.
- Researched the dublin bikes api which provides the relevant data.
- Researched similar projects and discovered appropriate techniques such as volcanoes and black holes.
- Used python to download the data via the api.
- Conducted general field research which included talking to dublin bikes employees, talking to customers and general observation.

Terms of Reference

- Investigate the current situation of how to bikes are moved around the city.
- Create a database from the live data.
- Observe the distribution/slopes of the bikes coming into/ out of the bike racks and try and spot inconsistencies in the distribution as the racks become full/empty.
- Use these distributions/slopes to predict the length of the queue and length of time people are queuing for at each bike stand.
- Build multiple models to map the movements of bikes around the city.
- Compare the models and pick the best one.

Further Work

- Continue to investigate data analytics and machine learning techniques which could be used.
- Build the database by developing a program which runs every 30 mins and compiles the data.
- Observe the distribution/slopes of the bikes coming into/ out of the bike racks and try and spot inconsistencies in the distribution as the racks become full/empty.
- Use these distributions/slopes to predict the length of the queue and length of time people are queuing for at each bike stand.
- Build multiple models to map the movements of bikes around the city.
- Compare the models and pick the best one.

Conclusion

There is no model or algorithm currently in place to distribute the bikes. There is a major traffic issue with dublin bikes which leads to queues and people waiting to get or take a bike. This problem could be reduced with the implementation of an algorithm to dictate the movement of bikes around the city. The most appropriate model is to be determined.

C. LITERATURE REVIEW – REBALANCING BIKE-SHARING SYSTEMS

Introduction

Bike sharing schemes are fast emerging in many cities around the world. This is a relatively recent concept with Velib (Paris) being one the pioneer systems of bike sharing. This system was implemented in 2007. Velib now has over 100,00 riders per day (En.wikipedia.org, 2018). Many cities have since successfully adopted this system. As of December 2016, approximately 1000 cities worldwide have a bike-sharing program (En.wikipedia.org, 2018)

The most significant challenge for the schemes is the rebalancing of bikes within the network to meet the demand needs of users. An inefficient system can result in increased costs for users and system owners. Users can incur a money-cost if there is no stand available to return their bike and a time-cost when looking to take or return bikes if they are not available. System owners incur the cost of extra vehicle trips to re-balance the bikes.

Although this is a relatively new system, there has been a lot of literature written on the topic. These can be divided into the static case and the dynamic case. The static case involves rebalancing the bikes when the system is closed i.e. users cannot interact with the system during rebalancing. This is usually done at night when there is little demand for the bikes. The dynamic case involves rebalancing the bikes when the system is open. This is more complex because the system is constantly changing and the demands must be constantly met as users move bikes around the network.

There is extensive literature available surrounding static rebalancing. However, as the dublinbikes problem is a dynamic rebalancing problem, this will be the primary focus of this review.

Static Case

The objective of a static rebalancing problem is to visit all the necessary stations in the network in the minimum time in order to redistribute the bikes to maximise the efficiency of the system. The majority of the literature reviewed used a two-prong approach to solve the problem. The first element is a bike distribution problem which predicts the state of the system when it closes and the optimum state of the system when it reopens. The second element is a vehicle routing problem which aims to optimise the efficiency of the vehicles distributing the bikes. Most literature focuses on the second element as their main study.

The bike distribution problem is often tackled through forecasting (Alvarez-Valdes et al., 2016, Faghih-Imani et al., 2015). Much of the literature surrounding the static does not include the bike distribution problem as part of their study and it is assumed that these targets are previously available. (Rainer-Harbach et al., 2013, Benchimol et al., 2011, Cruz et al., 2017). Various approaches are taken for the vehicle routing problem. In all articles examined, the first step taken is to create a network graph with the bike stands as vertices. In all static cases examined, the edges of the network represent the movements of the redistribution vehicle.

The most common approach taken is a Travelling Salesman Heuristic. Benchimol et al. (2011) and Cruz et al. (2017) both begin by using the one commodity pick-up and delivery travelling salesman problem. This assumes there is a single redistribution vehicle. More complex algorithms went on to examine multi commodity pick-up and delivery Travelling Salesman method, which, more realistically models, a system with multiple redistribution vehicles (Benchimol et al., 2011). Another common method used was integer programming, which defines the system through a set of constraints which are to be minimised (Rainer-Harbach et al., 2013, Alvarez-Valdes et al., 2016).

Alvarez-Valdes et al. (2016) examine the problem in great detail. They combine a bike distribution problem using a forecasting model and vehicle routing problem using linear programming. Throughout the study, key issues are highlighted which had not been acknowledged in any other static studies. The first key point highlighted is the issue of damaged bikes within the system. The proposed model incorporates the dropping off of damaged bikes and collection of repaired bikes at a depot.

In addition, Alvarez-Valdes et al. (2016) conduct an in-depth analysis of patterns in bike data from the bike-sharing system in Palma de Mallorca (Spain). Some conclusions include that some stands are self-regulating as bike inflow generally matches bike outflow, thus they require no interaction. The significant difference between bike usage patterns on weekends and weekdays is highlighted: "The behaviour of the users at each station is very similar on weekdays, while it is clearly different on Saturdays and on Sundays". From this, 3 different patterns were distinguished for the purposes of their model:

(i) Weekdays
(ii) Saturdays
(iii) Sundays

The model is then tested using the Palma bike data which shows that the system becomes significantly more efficient with the use of this model.

Although the static redistribution problem can improve the efficiency of a system, a dynamic approach is more applicable to real life situations and gives better results.

Dynamic Case

A dynamic approach to the bike rebalancing problem suggests models which are constantly applied to system with the ultimate aim of avoiding stands ever being full or empty. Unfortunately, there is limited literature on the topic. The literature which is available all took varying approaches to the problem; thus, a broad range of solutions were examined. Similarly, to the static case the majority of the literature had two parts. Firstly, an analysis of the bike distribution problem. Secondly, a vehicle routing problem to optimise the distribution of the bikes.

Both Schuijbroek et al. (2017) and Chiariotti et al. (2018) use similar approaches to both parts. They both use a queuing system to optimise the bike distribution problem. As with Alvarez-Valdes et al. (2016) real data is used to create and test the models. Schuijbroek et al. (2017) use real data from Capital Bikeshare (Washington, DC) and Hubway (Boston, MA). Chiariotti et al. (2018) use Citibike (New York) to build and test the models. Both data had similar structures including bike trip data. Bike trip data is composed of the start and end time of the trip and the start and end station of each trip. Schuijbroek et al. (2017) have data from over 2 months, whereas Chiariotti et al. (2018) use data from over 4 years.

In order to overcome the problem of the variance between weekday and weekend data, Schuijbroek et al. (2017) limited the study to weekday data points alone. Chiariotti et al. (2018) acknowledge the difference in patterns, however, do not remove these data points. This model is built over a larger time period; thus, it should be able to detect the pattern difference. Schuijbroek et al. (2017) utilise a Markov Queuing model to predict bike demand at each stand. Demand is predicted with a finite capacity (the minimum and maximum number of stands). Similarly, Chiariotti et al. (2018) use a Markov Queuing model to optimise the bike distribution problem but do not establish an upper and lower bound for the demand level i.e. the queue doesn't have a finite capacity. Chiariotti et al. (2018) take a "survival time" approach, where instead of predicting the demand for bikes it predicts the time remaining before the stand becomes full or empty.

In terms of the vehicle routing problem, Chiariotti et al. (2018), like Alvarez-Valdes et al. (2016), use integer programming to optimise the redistribution of the bikes. Whereas Schuijbroek et al. (2017), like Cruz et al. (2017) explore a range of Nearest Neighbour heuristics, including the Travelling Salesman heuristic.

Both Schuijbroek et al. (2017) and Chiariotti et al. (2018) use clustering to maximise rebalancing optimization. However, they take a different approach when applying the clustering. Chiariotti et al. (2018) clusters stations of similar demand and treats each cluster as one unit. Schuijbroek et al. (2017) cluster the stands on a larger scale and only carries out vehicle routing within each stand. This breaks the solution down into multiple single vehicle routing problems. It was found that certain clusters were self-sufficient so this was an efficient method to reduce the level of rebalancing. Clustering was a beneficial technique for both models to increase efficiency. Chiariotti et al. (2018) highlight the importance of having GPS tracking data which gives the bike trip information. It states how this has previously hindered the ability to build accurate models, thus hinders the efficiency of bike sharing systems. Both of these models outperformed the benchmark solutions of not having any rebalancing solution and a static rebalancing solution.

A contrasting dynamic approach was examined by Hong et al. (2015) and Benarbia et al. (2013). This involved creating a Spatio Temporal graph of live bike data. Each station is treated as a vertex in the network. The edges in the model presented by Hong et al., 2015 represent the movements of the bikes, whereas, the movements in the model presented by Benarbia et al. (2013) represent the movements of the redistribution vehicles. Hong et al. (2015) use approximately one million bike trips observations from Citibike (New York) as its case study and unlike the other dynamic studies Benarbia et al. (2013) does not test its algorithms on real data. Data equivalent to that used by Schuijbroek et al. (2017) and Chiariotti et al. (2018) was used.

Benarbia et al. (2013) applies Dantzig-Wolfe decomposition and Benders decomposition to the network in order to predict bike demands. Consistent with the other studies examined, these demand predictions are used as input to a vehicle routing problem. As seen the methods proposed by Rainer-Harbach et al. (2013) and Alvarez-Valdes et al. (2016), Integer Programming was used as the vehicle routing solution. Hong et al. (2015) applies Volcanoes and Black Holes Detection algorithm to the network. This identifies periods of high inflow of bikes into an area (black hole) and periods of high outflow of bikes from an area (volcano). This reveals the movements of the bikes around the city which can be used to inform vehicle drivers of potential high demand areas. It also can also optimise the network by observing areas of constant high demand where a potential new stand could be installed.

Conclusion

The wide range of literature available demonstrates the various types of methods that have been used. When comparing these studies to the dublinbikes problem, the available data is a cause for concern. The dublinbikes live data provided the number of bikes and free spaces at each bike stand, rather than bike trip data. As stressed by Chiariotti et al. (2018), the GPS data is essential to building an accurate model, especially for a dynamic rebalancing

D. Glossary

ARIMA

Autoregressive integrated moving average(ARIMA) is a general time series model used to forecast future values. It combines a weighted sum of past values and a weighted sum of values of the errors. ARIMA is composed of the three parameters.

ARIMA(p,d,q):

- AR(p)- This is an autoregressive model of order p which states that the Yi is a linear function of the previous p values of the series plus and error term.
- I(d) - This is the integrated part of the time series which is used to remove trends by differencing the time series.
- MA(q) - This is a moving average model adjusts for the error from one term to the next in the next forecast.

Information Criterion

Information criterion estimate the quality of statistical models relative to each of the other models. AIC (Akaike Information Criterion) AND BIC (Bayesian Information Criterion) are used to find the best ARIMA model. AIC and BIC are calculated as follows:

- AIC = -2log(L)+2m
- BIC = -2log(L)+2mlog(n)
- Where: L = likelihood of the data with a given model
  n = number of observation
  m = number of parameters used in the model

Complexity Parameter

Complexity parameter(cp) is a penalty based on the size of the tree. It is used to choose the optimal size tree. This helps us manage the trade-off between overfitting and under fitting the data. Multiple complexity parameters were tried and the best one chosen for the final Single Tree model.

Cross Validation

Cross-validation is a technique to evaluate predictive models. The data is randomly split into 5 equal size subsamples. Of the 5 subsamples, a single subsample is used as a test set, and the other 4 subsamples are used as training data. The cross-validation process is then repeated 5 times with each of the subsamples used once as the test data. The 5 results are then averaged to produce a single prediction.

Lowess

Lowess uses a weighted average of the surrounding points to calculate the smoothed value. The number weightings and number of points to be considered are controlled by the smoothing parameter, f. The smoothing parameter used was 1/100 which is 1/100th of the data.

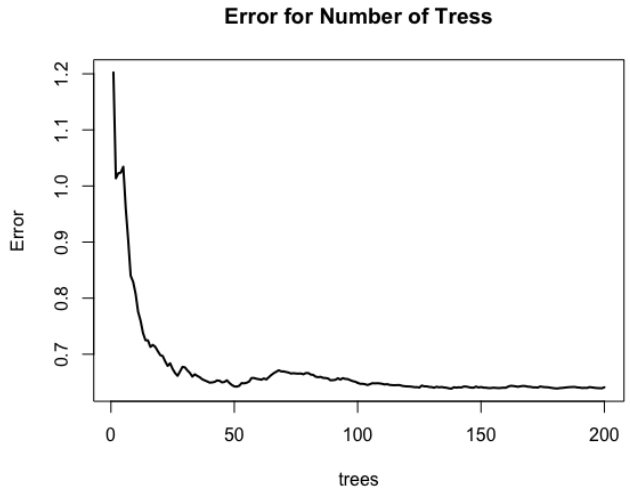
E. MODEL PERFORMANCE

E.1 Random Forest Weekend Model

Grid search results

mtry	RMSE	Rsquared	MAE
2	4.352992853	0.102127367	1.663458826
4	4.040774625	0.240245607	1.575774832
6	3.777374328	0.325086812	1.453370929
8	3.596480658	0.37155111	1.345128685
10	3.489976864	0.397913523	1.261075456

RMSE for Number of Trees

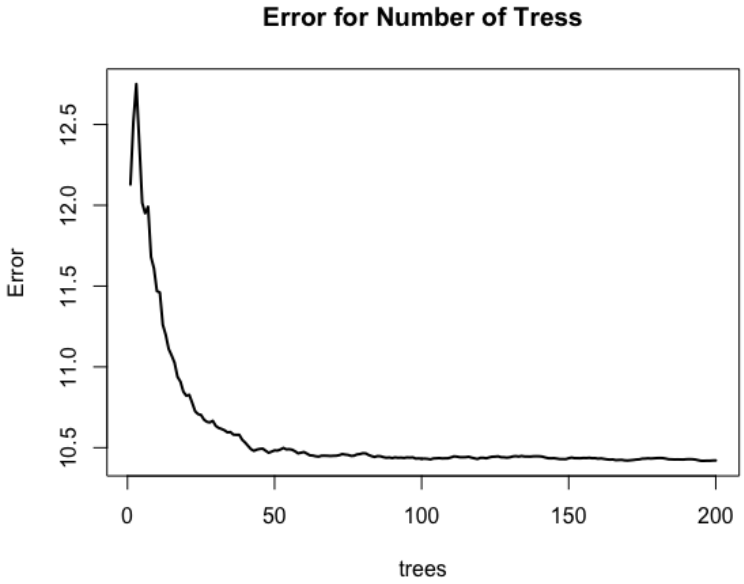


E.2 Random Forest Weekend Model

Grid search results

mtry	RMSE	Rsquared	MAE
2	1.39368398	0.356604021	0.834667138
4	1.146602792	0.547245079	0.676465073
6	1.025100671	0.624802643	0.578942635
8	0.961681195	0.66487592	0.523341732
10	0.934206155	0.679624169	0.489119513

RMSE for Number of Trees

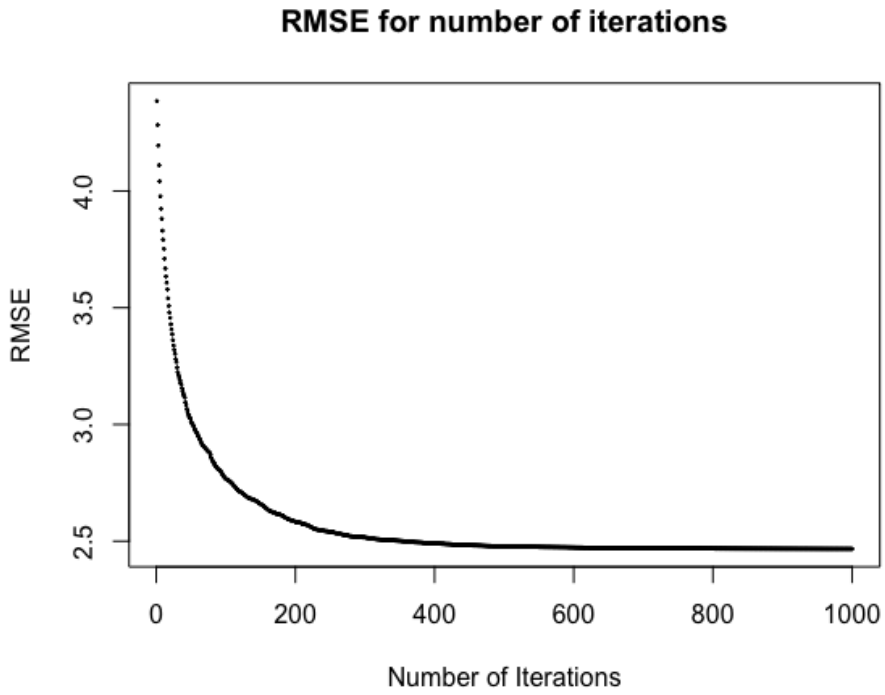


E.3 Extreme Gradient Boosting Weekday Model

Grid search results

eta	max_depth	RMSE	Rsquared	MAE
0.0001	2	4.493944654	0.01150967	1.812769007
0.001	2	4.458522084	0.01460834	1.722703996
0.01	2	4.436018017	0.020913038	1.735356243
0.1	2	4.377282878	0.045668492	1.793185531
0.25	2	4.335988836	0.063082979	1.824336639
0.5	2	4.300531896	0.078027371	1.84364515
0.0001	4	4.486303854	0.032546457	1.808817723
0.001	4	4.401038054	0.054765122	1.716105741
0.01	4	4.260390242	0.111519851	1.721638198
0.1	4	3.933682116	0.237030961	1.663577333
0.25	4	3.732518835	0.308469947	1.593248122
0.5	4	3.624526517	0.348447509	1.55328691
0.0001	6	4.470685461	0.069245662	1.801923274
0.001	6	4.296242362	0.119474834	1.689044976
0.01	6	4.000746244	0.224500801	1.630377305
0.1	6	3.499828816	0.393452016	1.348444763
0.25	6	3.435967432	0.425120402	1.224677917
0.5	6	3.451700665	0.426092404	1.213037194
0.0001	8	4.43510182	0.152562371	1.789037559
0.001	8	4.132517213	0.207827847	1.620975084
0.01	8	3.679520722	0.339264352	1.458334786
0.1	8	3.415872695	0.434217521	1.139783358
0.25	8	3.419535604	0.437175736	1.116678758
0.5	8	3.434458047	0.438244217	1.107818841
0.0001	10	4.398645466	0.23493723	1.773237284
0.001	10	3.951592436	0.282188778	1.538443358
0.01	10	3.503597103	0.393221213	1.285995188
0.1	10	3.409134163	0.441161293	1.08708953
0.25	10	3.419910889	0.441756703	1.069256383
0.5	10	3.4264439	0.442420805	1.077342669
0.0001	12	4.369576682	0.304687179	1.759402696
0.001	12	3.796709882	0.339585813	1.452221566
0.01	12	3.417175048	0.427112877	1.144559215
0.1	12	3.416458676	0.442480819	1.052552085
0.25	12	3.418013341	0.443063146	1.055350759
0.5	12	3.43198888	0.441761204	1.060910849

RMSE for Number of Iterations

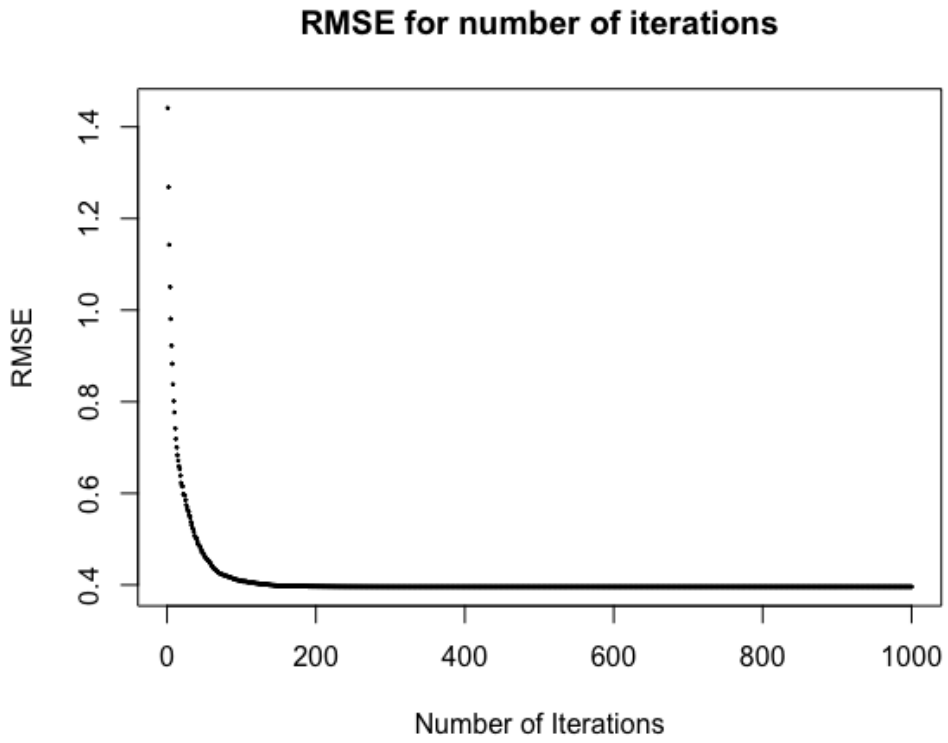


E.4 Extreme Gradient Boosting Weekend Model

Grid search results

eta	max_depth	RMSE	Rsquared	MAE
0.0001	2	1.659911526	0.103288177	1.046627272
0.001	2	1.547642445	0.141147722	0.937315001
0.01	2	1.457415436	0.20704648	0.904085932
0.1	2	1.402567536	0.262682148	0.882252888
0.25	2	1.397688748	0.267410169	0.878368289
0.5	2	1.377055843	0.287194431	0.865333324
0.0001	4	1.648963206	0.177356787	1.040921029
0.001	4	1.49121994	0.219603408	0.912373538
0.01	4	1.330050601	0.346398261	0.81889023
0.1	4	1.163486921	0.498419865	0.694983503
0.25	4	1.118936539	0.532654772	0.656843159
0.5	4	1.09848974	0.547218559	0.633992265
0.0001	6	1.637053172	0.260356014	1.032121029
0.001	6	1.410133335	0.338100243	0.860365103
0.01	6	1.148283154	0.521276987	0.685973223
0.1	6	1.002492958	0.625876026	0.548015893
0.25	6	0.982535429	0.637999964	0.524277228
0.5	6	0.994325771	0.629192435	0.521817801
0.0001	8	1.623088699	0.357054412	1.023680828
0.001	8	1.329168787	0.43401725	0.799921618
0.01	8	1.015188453	0.620391828	0.555981629
0.1	8	0.945971931	0.664863871	0.483562867
0.25	8	0.940532606	0.667903988	0.467673817
0.5	8	0.957342942	0.655813633	0.473819248
0.0001	10	1.61125518	0.434544152	1.016280666
0.001	10	1.246132923	0.523720856	0.735528558
0.01	10	0.951982059	0.660632077	0.474765993
0.1	10	0.934979524	0.671958641	0.455917152
0.25	10	0.929974061	0.674171976	0.441944656
0.5	10	0.930735274	0.674226079	0.438079379
0.0001	12	1.601588992	0.495825849	1.01009263
0.001	12	1.195853909	0.567150606	0.695545938
0.01	12	0.941539441	0.666415659	0.447658486
0.1	12	0.931391136	0.673032682	0.436054656
0.25	12	0.932949782	0.672385667	0.426490211
0.5	12	0.933225001	0.672962754	0.425676235

RMSE for Number of Iterations



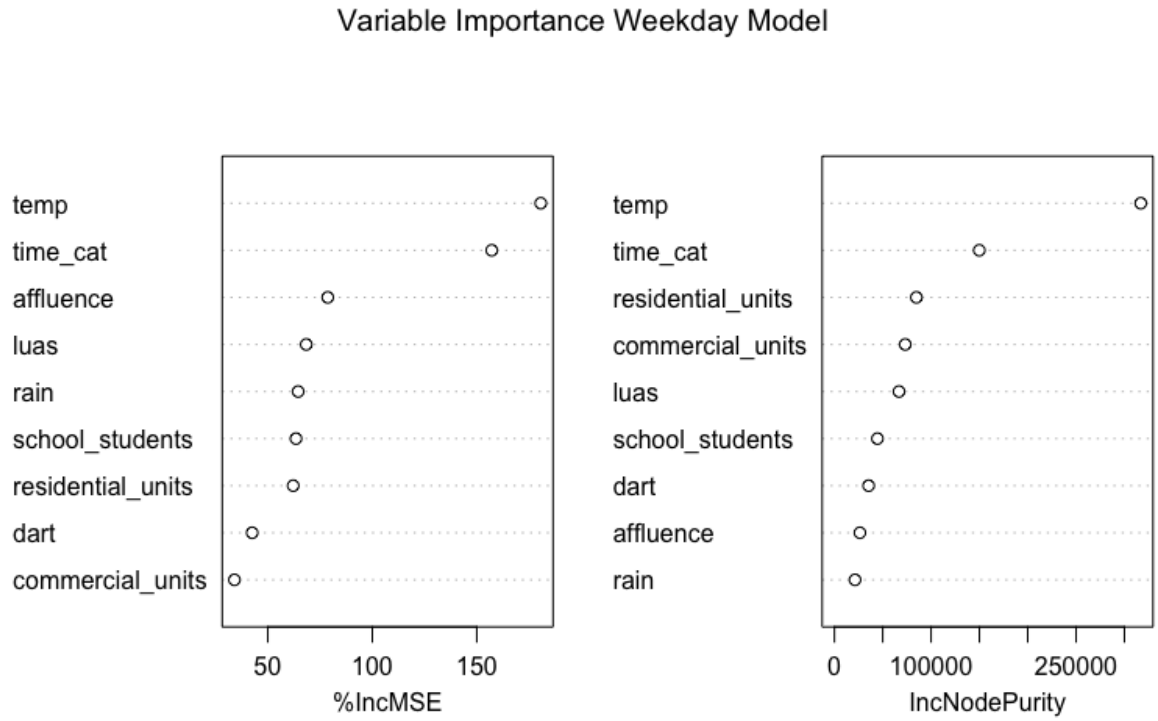
F. VARIABLE IMPORTANCE TABLES

F.1 Weekday Model

Random Forest Weekday Variable Importance Values

	%IncMSE	IncNodePurity
rain	3.129139	21309.43
temp	22.715034	316968.65
luas	9.884822	66811.89
dart	2.318146	35394.52
school_students	4.073760	44490.37
affluence	17.483765	26363.47
residential_units	5.455978	84830.41
commercial_units	5.216095	73254.15
time_cat	19.334107	149885.00

Random Forest Weekday Variable Importance Charts

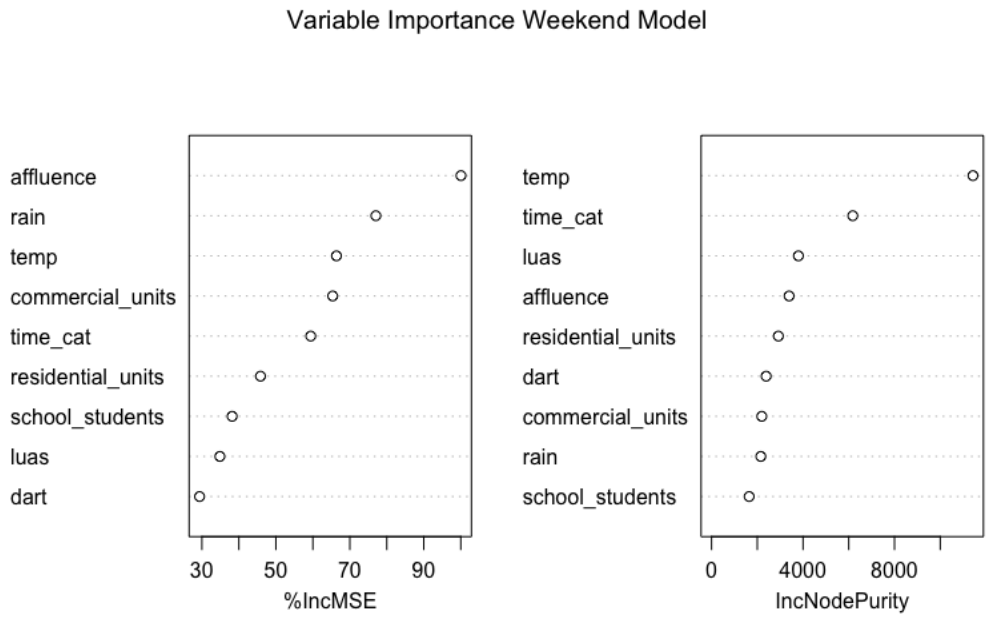


F.2 Weekend Model

Random Forest Weekend Variable Importance Values

	%IncMSE	IncNodePurity
rain	0.6271821	2153.857
temp	3.1389956	11431.530
luas	1.3129736	3797.930
dart	0.6889799	2387.575
school_students	1.1609361	1644.958
affluence	2.6009851	3390.963
residential_units	1.1965121	2921.504
commercial_units	0.7137088	2196.039
time_cat	1.9442388	6174.811

Random Forest Weekday Variable Importance Charts



REFERENCES

Alvarez-Valdes, R., Belenguer, J., Benavent, E., Bermudez, J., Muñoz, F., Vercher, E. and Verdejo, F. (2016). Optimizing the level of service quality of a bike-sharing system. *Omega*, 62, pp.163-175.

Benarbia, T., Labadi, K., Omari, A. and Barbot, J. (2013). Balancing dynamic bike-sharing systems. *Control, Decision and Information Technologies (CoDIT)*.

Benchimol, M., Benchimol, P., Chappert, B., de la Taille, A., Laroche, F., Meunier, F. and Robinet, L. (2011). Balancing the stations of a self service “bike hire” system. *RAIRO - Operations Research*, 45(1), pp.37-61.

Burke, M. and Brown, A. (2007). Distances people walk for transport. *Road & Transport Research*.

Chiariotti, F., Pielli, C., Zanella, A. and Zorzi, M. (2018). A Dynamic Approach to Rebalancing Bike-Sharing Systems. *Sensors*, 18(2), p.512.

Cruz, F., Subramanian, A., Bruck, B. and Iori, M. (2017). A heuristic algorithm for a single vehicle static bike sharing rebalancing problem. *Computers & Operations Research*, 79, pp.19-33.

Daft.ie. (2018). *Search Ireland's No. 1 Property Website | Daft.ie*. [online] Available at: http://www.daft.ie/ [Accessed 21 Mar. 2018].

Department of Education and Skills. (2018). *Data on Individual Schools*. [online] Available at: https://www.education.ie/en/Publications/Statistics/Data-on-Individual-Schools/ [Accessed 21 Mar. 2018].

Developer.jcdecaux.com. (2018). *JCDecaux Developer*. [online] Available at: https://developer.jcdecaux.com/#/opendata/vls?page=dynamic [Accessed 21 Mar. 2018].

Dublinbikes.ie. (2018). *Dublin - Dublinbikes*. [online] Available at: http://www.dublinbikes.ie/ [Accessed 21 Mar. 2018].

En.wikipedia.org. (2018). *Vélib'*. [online] Available at: https://en.wikipedia.org/wiki/Vélib%27 [Accessed 21 Mar. 2018].

Faghih-Imani, A., Hampshire, R., Marla, L. and Eluru, N. (2015). An Empirical Analysis of Bike Sharing Usage and Rebalancing: Evidence from Barcelona and Seville. *SSRN Electronic Journal*.

Geodictionary. (2018). *Geodictionary*. [online] Available at: https://www.geodirectory.ie/Maps/Find-Addresses-Near-You.aspx [Accessed 21 Mar. 2018].

Hong, L., Zheng, Y., Yung, D., Shang, J. and Zou, L. (2015). Detecting Urban Black Holes Based on Human Mobility Data.

Luas.ie. (2018). *Luas | Home*. [online] Available at: https://www.luas.ie/ [Accessed 21 Mar. 2018].

Met.ie. (2018). *Met Éireann - The Irish Meteorological Service Online*. [online] Available at: https://www.met.ie/ [Accessed 21 Mar. 2018].

Rail, I. (2018). *DART & Commuter*. [online] Irish Rail. Available at: http://www.irishrail.ie/about-us/dart-commuter [Accessed 21 Mar. 2018].

Rainer-Harbach, M., Papazek, P., Hu, B. and R. Raidl, G. (2013). Balancing Bicycle Sharing Systems: A Variable Neighborhood Search Approach. *LNCS*, 7832.

Schuijbroek, J., Hampshire, R. and van Hoeve, W. (2017). Inventory rebalancing and vehicle routing in bike sharing systems. *European Journal of Operational Research*, 257(3), pp.992-1004.