Technical University of Munich
Chair of Transportation Systems Engineering

Master's Thesis

Analysis and Prediction of Bike Sharing Traffic Flow

Case of CitiBike, New York

Author:
Salma Yosri Yousif Hamad

Matriculation No:
03693576

Supervised by:
Dr. Tao Ma

Chair leader:
Univ.-Prof. Dr. Constantinos Antoniou

September 16th, 2020

Declaration of Authorship

I hereby confirm that the presented thesis work has been done independently and using only the sources and resources as are listed. This thesis has not previously been submitted elsewhere for purposes of assessment.

16th September 2019 Salma Hamad

Abstract

Bike Sharing Systems have witnessed unprecedented growth and significant scholarly attention in recent years. Technological advancement, environmental awareness, and the demand for socially equitable transport modes were the major contributors to this development. However, with the ongoing expansion of these systems, companies are faced with the constant need to rebalance them and meet the growing demand. Thus, operating companies are always exploring the right tools for flow prediction.

This thesis examines three machine learning algorithms and addresses the neglected aspect of multiple seasonality in time-series models. The study set out to explore the relationship between bike sharing and weather as well as its users. Then the four different techniques are developed and evaluated to determine the best performing algorithm and suggest further aspects of research in traditional time series models.

The research presented here confirms that neural networks deliver the best performance. The findings also provide a solid evidence base accounting for complex seasonality with traditional time series models

Acknowledgement

I would like to take this opportunity to thank my supervisor, Dr Tao Ma, for the constant encouragement, valuable advice, and patient guidance. He has provided throughout my time as his student. I have been truly fortunate to have had a supervisor who cared so much about my work, and who promptly responded to my questions and enquiries. My sincere gratitude and appreciation are never ending for all the support he has shown and given to me throughout my study journey. I would not have made it without his guidance and expertise.

I would also like to express my love and great appreciation to my family, my father Yousri Hamad and my mother Eman Ahmed whom have individually and unitedly supported me in many ways through good and bad times, especially whilst being away from them to study here. I am indebted to them for all their help. An incredibly special thank you goes out to my Sisters Deena and Sara, and my brother Moustafa.

Finally, I would like to thank the Technical University of Munich for giving me the opportunity to excel and making use of their valuable resources, campus experience and professional expertise. I would have not reached my goal without your outstanding assistance and would have not met some extremely interesting knowledgeable people.

إلى أبي وأمي..

Table of Contents

List of Figures

List of Abbreviation

A

Adam: Adaptive Moment Estimation 42
ADH: Augmented Dickey-Fuller test 39
AIC: Akaike Information Criterion. 39
ANN: Artificial Neural Networks 36
ARIMA: Autoregressive Integrated Moving Average 8
ARMA: Auto-Regressive Moving Average 8

B

BS: Bikesharing 6

C

CART: Regression Tree 9
CARTs: Classification and Regression Trees 35
CDF: Cumulative Distribution Function 10
CNN: Convolutional Neural Network 10
CNS: Courier network services 6
CV: Cross-validation 44

D

DBSS: Dockless Bikesharing systems 7

E

EDA: Exploratory Data Analysis 23

F

FZ: Functional Zones 9

G

GBRT: Gradient Boosting Regression Tree 9
GC: Grid Clustering 9
GCP: Google Cloud Platform 14
GRU: Gated Recurrent Unit 10

H

HA: Historical Average 9

I

IDE: Integrated Development Environment 14

List of Tables

Chapter 1

Introduction

1.1 Background

Climate change, air quality, fluctuating fuel prices have recently become of primary concern and heightened the need for a more sustainable and green mean of transport. In the last two decades, there has been a surge of interest in Bikesharing Systems, and they have witnessed an expansive growth around major cities. There is concrete evidence that Bikesharing Systems play a crucial role economically, environmentally, and provide notable health benefits. Bikesharing systems are inexpensive, efficient, flexible and accessible. According to *Shaheen et al.* [1], bikes have the potential of cutting off the production of 37,000 kg of carbon dioxide per day compared to a car travelling the same distance. The given flexibility of the system, increase the accessibility of the city or the area where it is provided, adding the benefit of reaching and contributing to local businesses [2].

The majority of existing bike sharing systems are docked systems. They have docking stations distributed around the city, where users can self-checkout providing the choice of picking up a bike from any station and dropping it in any of the available stations with free docks. In addition, the reasonable pricing of the system makes it appealing for commuter and students alike. It is also a solid solution for the 'last mile' problem [1], which makes it even more appealing for commuters who use public transport. However, the limited docks and bikes available in each station constrain the flexibility of the system due to the overwhelming demand contributed by commuters in residential and business areas during the morning and evening rush hours.

1.2 Research Objective and Questions

This prospective study was designed to investigate the performance of traditional time series models and the common machine learning used in prediction the station-based hourly flow. It set out to assess three machine learning algorithms; random forest, gradient boosting regression tree, and multilayer perceptrons, and ARIMA time series model exploring the different seasonality in the system. Surprisingly, there is little to no research on modelling the complex seasonality of a bike-sharing system using time series models.

The motivation for this research is addressing the rebalancing issue of the bike sharing system. The high demand during the morning and evening peaks results in an uneven distribution of bikes. Additionally, forecasting the demand is of tremendous commercial value. The goal will be achieved by performing the algorithms as mentioned above and examine the seasonality closely to incorporate in the time series model. This study aimed to address the following research questions:

- Which machine learning algorithm is most suitable to predict the flow, and how is it compared to the time series model?
- How to address the multiple seasonality of bike-sharing system and include them in a time series model?

To address these questions, the primary research goals are:

- Analyse and visualise the general characteristic of the bike sharing system, and the general behaviours of the users under different weather circumstances.
- Predict the check-in and check-out in the next hour
- Train the three predictive models; Random Forest, Gradient Boosting Regression Tree, and Multilayer Perceptrons
- Implement the Autoregressive Integrated Moving Average (ARIMA) model with multiple seasonality
- Evaluate the performance as well as the computational costs to train the model and tune its hyperparameters
- Identify the best performing model using the defined performance metrics

1.3 Research Framework

Data for this study were collected from Citi Bike Bikesharing system in New York City. The top three stations with the highest count of hourly check-in and check-out were selected; some station were repeated (Figure 3.11 and Figure 3.10). They are located in Manhattan, in either highly attractive touristic areas or business areas, as seen in the following figures.

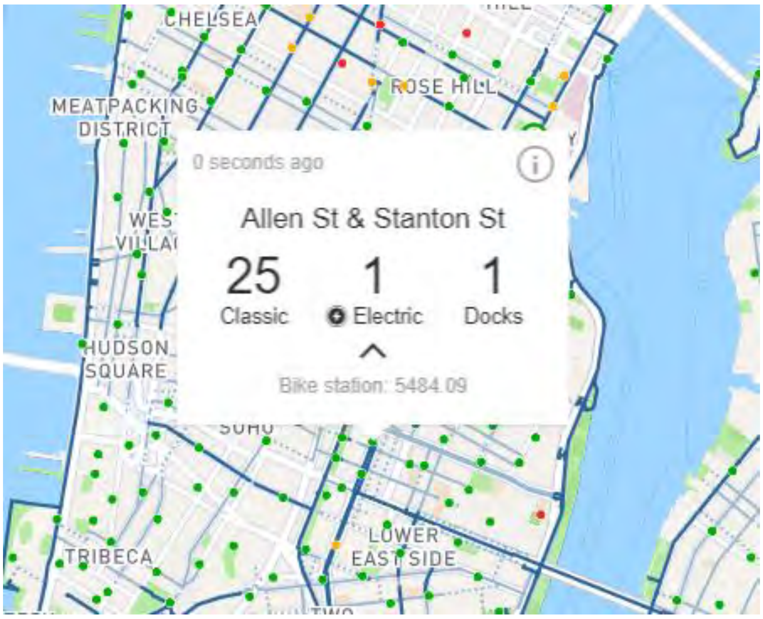
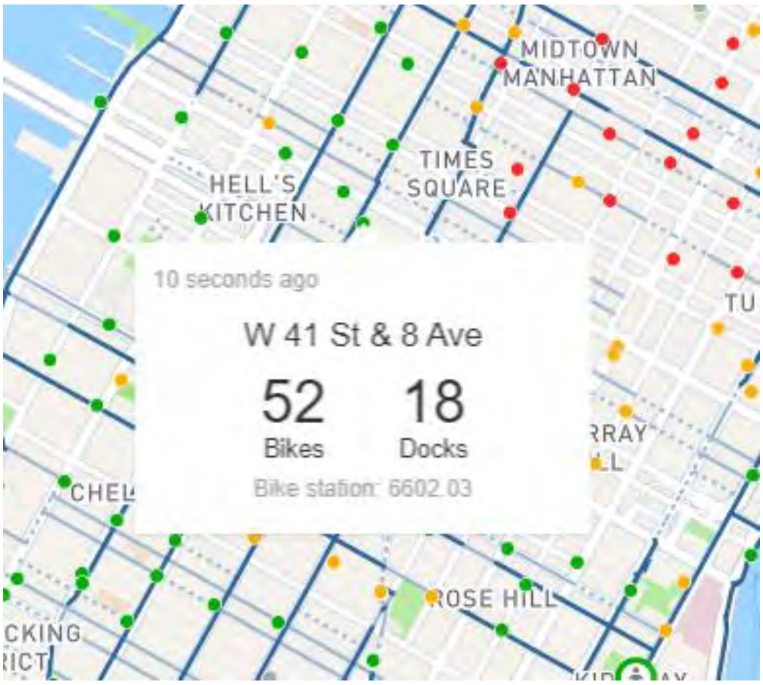


Figure 1.1 Allen St & Stanton St Station [3]



10 seconds ago

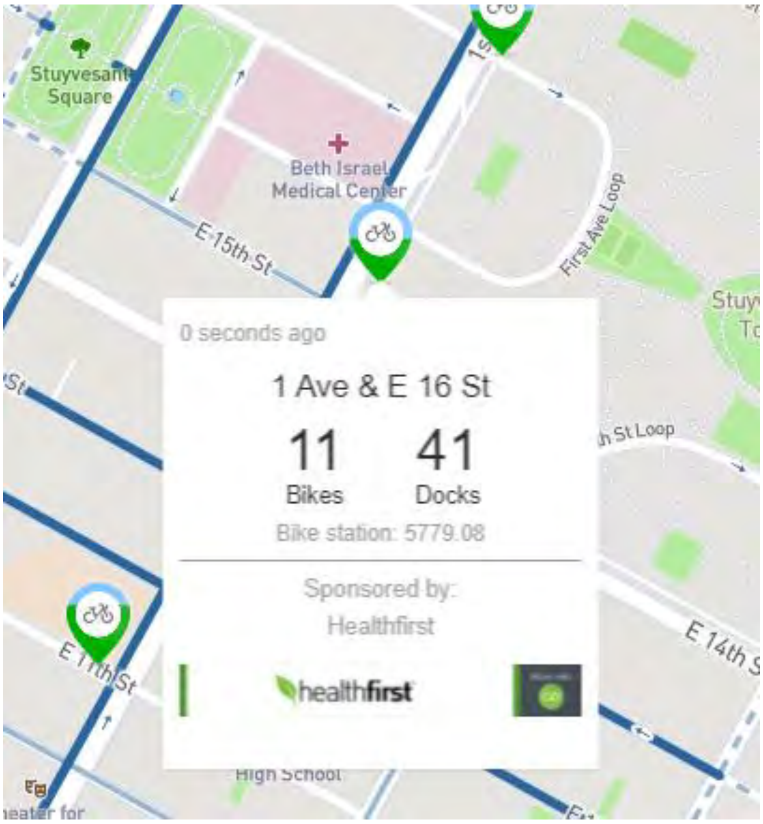
W 41 St & 8 Ave

52 Bikes

18 Docks

Bike station: 6602.03

Figure 1.2 W 41 St & 8 Ave Station [3]



0 seconds ago

1 Ave & E 16 St

11 Bikes

41 Docks

Bike station: 5779.08

Sponsored by: Healthfirst

Figure 1.3 1 Ave & Ee 16 St Station [3]

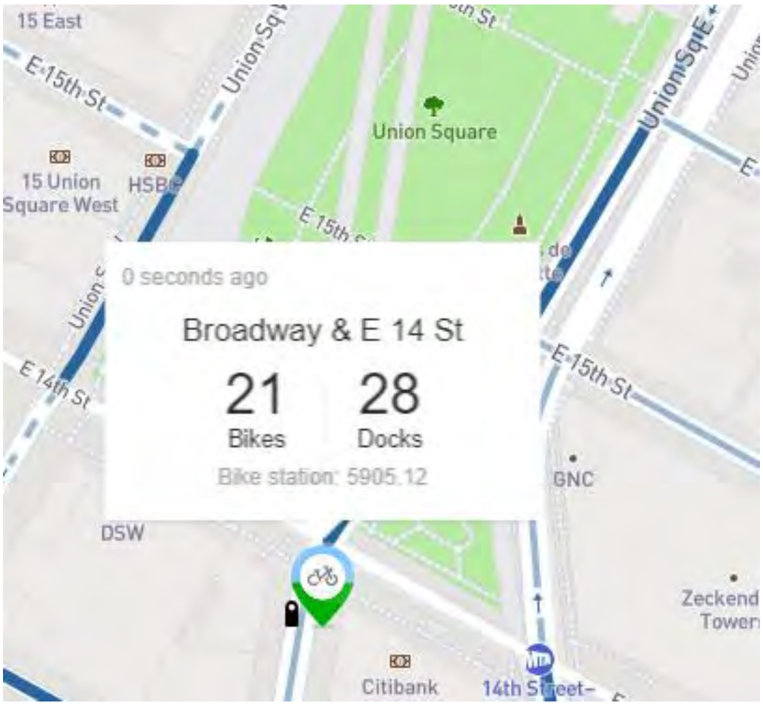


Figure 1.4 Broadway & E 14 St Station [3]

My thesis is composed of four themed chapters. Chapter 1 contextualises the background and the objective and research questions. Chapter 2 discussed the previous studies and state-of-the-art literature in bike-sharing demand prediction. The third chapter (Chapter 3) is concerned with the methodology employed for this study. The final chapter (Chapter 4) summarises the principal findings of these experiments and draw the conclusion and suggestion for further research.

Chapter 2

Literature Review

This chapter is divided into two main sections. The first section gives a brief overview of the foundation of shared mobility, its concepts, and classifications. It also examines bike sharing in detail. Lastly, the different methodologies frequently implemented to analyse bike sharing demand and build a predictive model are summarised.

2.1 Shared Mobility

This section begins by investigating the concept of a shared economy and how relevant it is to form a general understanding of shared mobility. Then, shared mobility is defined, and its different types are introduced. Finally, the Bike Sharing System is investigated lengthily.

2.1.1 Shared Economy

The past two decades have seen rapid growth in shared economy driven by the prominence of resources scarcities and the attempt to merge both offline and online worlds [4]. However, only in the last decade, the term "shared economy" has propelled to the forefront of research. Shared economy is defined as "a socioeconomic system enabling an intermediated set of exchanges of goods and services between individuals and organisations which aim to increase efficiency and optimisation of underutilised resources in society" [5]. Accommodation and transport are amongst the major underutilised sectors that commonly implement shared economy concepts [4], [6]. For example, Airbnb and Uber are some of the successful business models that early adopted this scheme, that was built upon underutilisation as their driving force [4].

To date, previous studies highlighted three main components that are associated with shared economy, namely economic, social, and environmental [4]–[6]. These are three key aspects of sustainability, which is another product of shared economy. The economic component supports creating new business ventures and the overall growth of the economy. The social component identifies inequality among individuals and is employed as a tool for raising awareness about overconsumption and tie the society together in a collaborative manner. Both economically and socially, shared economy benefits individuals in controlling their spending habits and creating a new source of revenue. Social economy indirectly impacts the environment; through the reuse of existing services and goods, and thus less raw materials are used, and individuals become more aware of their consumption behaviour. Additionally, the booming of technology in recent years and the rapid development of social networks and electronic devices served as an enabler for the outspread and promotion of shared economy [7].

Lastly, *Cohen et al*. concluded seven unique features of sharing business models. 1. Platforms for collaboration, 2. Underutilised resources, 3. Peer-to-peer interactions, 4. Collaborative governance, 5. Mission-driven, 6. Alternative funding, and 7. Technology reliance [5, p. 3].

2.1.2 Shared Mobility Definition and Different Forms

Shared mobility is a component and a forerunner of shared economy and technological advancement. It is defined as "the shared use of a vehicle [..] that enables users to have a short-term access to transportation modes on an 'as-needed' basis" [8, p. 4]. Shared mobility is classified into the following categories:

1. Carsharing
2. Micro-mobility: Bike Sharing (BS), scooter sharing
3. Personal vehicle sharing: Peer-to-peer (P2P) carsharing and fractional ownership
4. Ridesharing: Carpooling
5. Shuttle service
6. Vanpooling
7. Ride-sourcing and transport network companies (TNCs): Ride-hailing
8. Courier network services (CNS)

In the first two forms of shared mobility, users pay a membership fee or a time-based fee to get access to the vehicles. They are considered on-demand services that do not require advanced booking. The operating companies, which in this case also own the fleet vehicles, are responsible for insuring, maintaining, and storing those shared vehicles as well as paying any fees required for the usage of public spaces or parking to the authorities. In peer-to-peer carsharing, operating companies utilises privately owned vehicles, but the owners are responsible for the maintenance of their vehicles.

Ridesharing and ride-sourcing are on-demand services that also utilise the use of privately owned vehicles. The driver of the vehicle; would it be the owner or another person hired by the owner, subcontracted by the operating company. Carpooling rides requires passengers to have the same origin and destination or en route between the origin and destination. Vanpooling deploys the same concept as carpooling but on a larger scale.

Shuttle services are mostly seen in transport terminals and have a fixed origin and destination. Courier network services connect users to companies or individuals to facilitate package delivery.

2.1.3 Bike Sharing

Bike Sharing Systems were developed primarily as a result of environmental concerns by non-profit organisations. Bike Sharing went through a series of four generations. The first Bike Sharing System emerged in July 1965 in Amsterdam. A fleet of fifty bikes was painted white, hence the name White Bikes *(or Free Bike Systems).* The bicycles were distributed around the urban city centre unlocked. Similar systems were later developed in La Rochelle, France, in 1974 and Cambridge, the United Kingdom, in 1993. These early implementations of free bike systems are categorised as the first-generation. This generation failed due to theft and vandalism. Shortly after, the second-generation of Bike Sharing System emerged as a coined-deposit system in 1993 in Copenhagen, Denmark, which marks the first appearance of docking stations. Users deposit a small fee that was around 3 Euros to unlock the bicycle. [1], [9], [10]

Nevertheless, this system was also susceptible to theft because of the user's anonymity. The third generation IT-based systems integrated advanced technology to overcome this issue, allowing the operating company to track the bicycle's usage and location as well as accessing user information. The system included a kiosk and a user interface technology for reservations, pick-up, and drop-off. The first third-generation system appeared in 1998 in Rennes, France, where users used a smartcard to unlock the bicycle. Lastly, the fourth-generation systems are characterised as demand-responsive and multimodal systems. A fourth-generation system may or may not incorporate docking stations, or it could be a hybrid of both systems. It is integrated with public transportation and other mobility sharing systems. It also incorporates an innovative bicycle redistribution and advanced IT-infrastructure, such as GPS tracking. [1], [9], [10]

2.1.3.1 Service Models

Bike Sharing systems follow three service models:

1. Station-based Bikesharing systems (SBBS)
2. Dockless Bikesharing systems (DBSS)
3. Hybrid Bikesharing systems.

In the station-based system, each station has multiple docks where users can unlock a bike and drop it off in any of the designated stations. The trip could be either a one-way or a round trip. Users can quickly locate available bikes through the system's mobile application that uses GPS tracking. Dockless Bikesharing systems allow users to pick up and drop off the bike within a specific geographical radius. Hybrid systems are a combination of both systems, where the user can opt to either pick up a dockless bike or a bike from a station and drop off the bike in a station or a non-station location. [11]

Both station-based and dockless systems carry a level of convenience to users; the former, with its distinctive locations, guarantees availability at the same place daily and a definitive start and end location. While the latter removes this obligation of planning the starting and ending points, and additional costs for stations' maintenance and balancing. However, SBBS users may encounter some issues locating an available dock to drop off the bike. Although DBSS mitigate this issue, yet users have to abide by parking the bike in an accessible and legal spot. Otherwise, this will increase the level of uncertainty of bikes' availability and accessibility. Moreover, incurring additional costs to locate and balance bikes' distribution throughout the operating area. On the other hand, hybridisation of the bike sharing systems could help to alleviate these issues and attain the benefits of both systems [12], [13].

Regardless of the service model applied, the trip purpose is highly dependent on the type of users, long-term (Subscribers) or short-term (Customers). However, the majority of trips made on weekdays are commuting trips from/to work or education, while trips on the weekends are for leisure and sightseeing. [14]

2.2 Demand Modelling and Forecasting

This section gives a brief overview and findings of the recent research in BS demand modelling and forecasting methods. The first subsection focuses on the prevailing methodologies and algorithms for bike usage prediction, and the second subsection is concerned with the concluded prediction methods applied in this research.

2.2.1 Modelling and Forecasting

A large and growing body of literature focuses particularly on forecasting bikes' or docks' availability in Station-Based Bikesharing systems. Operating companies are continually trying to optimise their redistribution system to "minimise the financial and environmental costs" [13] and, in return, ensure users' satisfaction. Real-time monitoring provides an intelligent management system but usually not sufficient to provide a robust one. Therefore, predicting the behaviour of users and bikes movement between stations is essential.

Previous research has examined both time series forecasting methods and machine learning methods in a univariate and multivariate setting, respectively. A univariate method involves only one variable typically indexed by time, which is also the predicted value. A multivariate method involves multiple variables that compose the independent variable(s), known as the predictor variable, and the dependent variable(s), known as the predicted variable.

Researchers developed several approaches to tackle the rebalancing issue and predict the availability of bikes. *Kaltenbrunner et al.* [15] based their approach on time series forecasting methods for predicting hourly station-level bike availability; they used naïve methods as their baseline approach and conducted an Auto-Regressive Moving Average (ARMA) model. Naïve approaches are considered cost-effective computationally, but they are rigid methods. Therefore, they are mainly set as baseline methods for comparison. ARMA model is a univariate time series forecasting method that is more sophisticated than the naïve methods, as it considers the variation of the predicted variable over time. In their case study of the Bicing Bikesharing system in Barcelona, Spain, *Kaltenbrunner et al.* [15] concluded that their ARMA model outperformed the naïve approaches. They also indicated that by incorporating the information of spatially close stations into the ARMA model, improved their prediction. However, there are significant drawbacks associated with the use of their approach. ARMA models cannot identify stations' non-stationarity. Their approach assumes that the mean and variance of available bikes in each station is stationary over time.

Yoon et al. [16] developed an application, CityRide, using data from Dublin, Ireland BSS to predict bike availability in the next 5 and 60 minutes. The created application is a "spatio-temporal prediction system" based on the Auto-Regressive Integrated Moving Average (ARIMA) model. ARIMA model is another univariate time-series forecasting model that can capture the seasonal trend. *Yoon et al.* [16] extended their ARIMA model by using different clustering methods to obtain the spatial correlation between stations; Voronoi region, K-Nearest Neighbour (KNN), and Linear Regression based method. Their attempt to use three different clustering methods were cumbersome as they delivered significantly similar error terms.

Overall, the approaches mentioned above were unable to incorporate meteorological factors that influence the demand significantly. Gallop et al. explored the relationship between weather and bike users' behaviour. It has conclusively been shown that temperature is mainly a significant factor [17]–[19]. Additionally, precipitation, high humidity, high wind speed, and snow are negatively correlated with Bikesharing demand [17], [19], [20] Machine learning methods are multivariate methods, and one of their advantages is that they allow the addition of meteorological data as predictor variables, thus capturing the complexity of the demand problem. Ensemble learning and Neural Network (NN) are currently the most prevalent methods for investigating bikes availability and flow. Gradient Boosting Regression Tree (GBRT) and Random Forest (RF) are the most adopted ensemble learning methods in predicting clustering-based or station-based short-term and long-term demand. In station-based prediction, machine learning algorithms have shown to outperform linear models [21], [22].

Li et al. [23] conducted a cluster-based prediction and integrated the weather information into their data. First, they utilised a bipartite clustering algorithm as a geographical clustering method to account for the correlation between stations; then, they used GBRT to predict the global lending amount of bikes in the next hour. Finally, a multi-similarity-based inference model predicted the rent proportion across clusters and the inter-cluster transition. Their baseline method included historical average (HA), ARMA model, a hierarchical prediction based on K-Nearest Neighbour (HP-KNN), and a uniform geographical grid clustering (GC), which GBRT showed outperformance over them with plausible results for anomalous periods.

Some researchers adopted the hierarchical demand prediction method, where they either predicted the demand from a global level, i.e., the system demand, to a cluster-level, or from a cluster-level to station-level [24], [25]. *Liu et al.* [25] predicted the demand before and after the expansion of CitiBike, New York Bikesharing system. They utilised a Bi-Clustering method using the point-of-interests (POIs) and Voronoi Regions extracted from Google Map API and taxi trips records. POIs were employed to distribute the demand and the Voronoi region to divide the system into service areas, i.e., functional zones (FZ). Their data included trip distance preference, zone-to-zone preference, and zone characteristics. As their baseline, they used RF, KNN regression, NN, and GBRT for station-level prediction, and FZ+GBRT for their hierarchical demand prediction. They firstly predicted the functional zones demand and distributed it considering the Voronoi region POI structure to each station. Their proposed FZ+RF model and FZ+GBRT baseline model both have shown smaller error terms, with FZ+RF hierarchical model outperforming them.

Yang et al. [26] predicted the total lending amount of station clusters in the next hour and compared the performance of linear and non-linear prediction models, namely Ordinary Least Squared (OLS), Poisson Regression, Regression Tree (CART), and Gradient Lifting Tree (GBRT). They introduced a new clustering method; the bottom-up hierarchical clustering method, which considers spatial constraints, and examined their models on three selected clusters regarding their accuracy and time consumption. The input data featured time and weather predictors as well as the lending amount in the previous hour, day, and week. The accuracies of their prediction models were relatively the same, with GBRT being the worst in terms of time consumption, mainly due to extensive parameter tuning. They concluded that the accuracy of their models improved by clustering compared to station-based models, and linear models are the optimal prediction models.

Collectively, these studies relied on utilising and optimising different clustering methods while using similar baseline methods and machine learning algorithms for prediction with and without integrating weather information. Similarly, *Hulot et al.* [27] utilised different reduction methods but to optimise and reduce the complexity of station-level demand prediction models. They were able to achieve better accuracy and reduction in computational time.

Yang et al. [28] proposed a spatio-temporal mobility model to predict the hourly check-ins and check-outs on a station-level. Their proposed mobility model was implemented for check-in estimation, while RF was adopted for check-out estimation. Consequently, each model carried out different parameters. Their proposed model included the number of bikes checked in and out, the probability of bikes transiting from a particular station to another, and the cumulative distribution function (CDF) of trip duration between station pairs. In contrast, RF included time factors, meteorology, and real-time bike availability. Both models outperformed their baseline model, which included HA and ARMA. Moreover, their proposed mobility model established a foundation for further research in rebalancing methods for Bikesharing.

Ruffieux et al. [29] studied real-time short-term and long-term predictions of bikes and slots availability focusing on two machine learning algorithms; RF and Convolutional Neural Network (CNN). They proposed a system to collect open-source data and examined the algorithms' accuracy using past and pseudo-live data. The data included the bike network, weather, and holiday information. RF showed promising results in short-term forecasting, i.e., not more than 6 hours. In comparison, CNN was better suited for long-term forecasting.

Wang et al. [30] carried out a station-based short-term prediction of the number of available bikes using a month-long data without the inclusion of meteorology or time factors. They developed two Recurrent Neural Network (RNN) models, Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU), and used RF as their baseline method. Although LSTM and GRU exhibited relatively similar results, GRU was more accurate and faster to train. Additionally, their RF model did not include any complex hyperparameter tuning and showed favourable result with short time intervals.

2.2.2 Conclusion

Thus far, previous studies tended to focus on clustering optimisation in terms of clustering-based forecasting, in which some stations lose their individuality. Nevertheless, some of these studies neglected the weather data despite their significant effect on Bikesharing demand. On the other hand, the majority of station-based forecasting were centred around machine learning algorithms with no regard to traditional time series forecasting except ARMA, which was mainly utilised as a baseline method and has shown many drawbacks. Furthermore, studies that employed the ARIMA model neglected the seasonality effect on demand.

This study examines and compares the ARIMA model considering the complex seasonality of Bikesharing demand and the three prominent machine learning algorithms: Random Forest, Gradient Boosting Regression Tree, and Neural Network, specifically Multilayer Perception (MLP).

Chapter 3

Methodology

This chapter is concerned with the methodological approach taken to achieve the objective of this study. A quantitative case-study approach was adopted to assess and determine the model that has the ability to deliver the most accurate prediction results of station-based check-ins and check-outs.

An overview of the targeted Bikesharing system is given, followed by an outline of the data sets collected. Afterwards, data analysis and visualisation of the whole system and the chosen stations for this study is performed. Lastly, a detailed explanation of the models performed.

3.1 Citi Bike

The data collected for this study is the open-source trip data taken from Citi Bike Bikesharing system in New York. It is a privately owned Bikesharing system launched in May 2013 and named after its first sponsor Citi Bank and currently operated by Lyft. The system is operational in New York City boroughs as well as in Jersey City, New Jersey. Citi Bike is a docked system with more than 750 stations; they are continually adding more stations and expanding to other boroughs in New York City. Moreover, they recently added e-bikes to their operational fleet in New York City.

The Citi Bike interactive stations map is colour-coded in proportion with the number of docks available in each; where red indicates that there are no bikes available, yellow indicates relatively short supply, green indicates an abundance of bikes, and grey indicates inactive stations. Additionally, when a user selects a station, the map shows the number of docks and the number of bikes (classic or electric) available. [3]

Users firstly need to register, and they can either opt to become annual members (subscribers) or one-time users (customers) with a single-trip pass or a day pass. Annual members can enjoy unlimited rides for the first 45 minutes with the additional cost of $0.15 per minute beyond the 45-minutes limit. Day passes include unlimited rides for the first 30 minutes with the additional cost of $4 per 15 minutes beyond the 30-minutes margin. [31]

Users have multiple options to start their rides, as following:

1. Citi Bike mobile application
2. Station kiosk
3. Lyft mobile application
4. Bike Key

For the first three options, the users enter the single-use 5-digit code provided into the dock's keypad to unlock a bike. In contrast, the bike key is simply inserted into the dock's slot to unlock a bike.

The scope of this study focuses on trips made within New York City, specifically the Bronx, Brooklyn, Manhattan, and Queens.

3.2 Data Collection

The data sets collected for this study are the trip data sets from Citi Bike and New York City hourly weather data for the period between November 2018 and November 2019. The following subsections present the details of both data sets.

3.2.1 Trip Data

The trip data sets collected for this study are available through the Citi Bike website as downloadable zipped files of monthly trip data. Trip data between November 2018 and November 2019 of the city of New York were downloaded and unzipped. Preliminary inspection of the data showed that each row represents a trip and includes the following information:

1. Trip duration (in seconds)
2. Start time and date
3. Stop time and date
4. Start station name
5. End station name
6. Station ID
7. Station latitude and longitude
8. Bike ID
9. User Type (Customer or Subscriber)
10. Gender (0: Unknown; 1: Male; 2: Female)
11. Year of birth

Trips made by staff for inspection, from/to test stations, and trips less than 60 seconds were removed by Citi Bike themselves. [32], [33]

3.2.2 Weather Data

The hourly weather data were obtained from a secondary source that provides historical weather data with different time aggregation; daily, hourly, or sub-hourly [34]. The same period used for the trip data was used to download the historical weather data for New York City, which are available as CSV (Comma-separated values) files.

The hourly weather data were downloaded in the US imperial system, and included the following information:

1. Location
2. Address
3. Resolved Address
4. Latitude
5. Longitude
6. Date time
7. Info
8. Id
9. Name
10. Cloud cover (percentage)

11. Minimum temperature (in Fahrenheit)
12. Maximum temperature (in Fahrenheit)
13. Temperature (in Fahrenheit)
14. Heat index (in Fahrenheit)
15. Wind chill (in Fahrenheit)
16. Precipitation (in inches)
17. Precipitation cover (percentage)
18. Dew point (in Fahrenheit)
19. Relative humidity (mean percentage)
20. Visibility (in miles)
21. Wind speed (miles per hour)
22. Wind direction
23. Weather type
24. Snow depth (in inches)
25. Sea level pressure (in millibars)
26. Wind gust (miles per hour)
27. Conditions

Further descriptions and indication of each column are further discussed in section 3.3.2 and 3.5 below.

3.3 Data Analysis

This subsection begins by laying out the tools and software used to perform data pre-processing and analysis. The architecture of data pre-processing is shown in Figure 3.1. Data pre-processing includes exploring, combining, and cleaning the data for further analysis and modelling.

The first step in this process was to pre-process the weather data and extract the useful features needed for this study. After that, the trip data were pre-processed and split into check-in/out trips. Finally, the trip data set was aggregated on an hourly basis and merged with the weather data to extract the top 3 stations used for modelling.

3.3.1 Software and Tools

Python 3.0 was the programming language used to analyse the data. Python is a versatile and efficient language that is faster than the most common programming languages used in data science and analytics. It has a wide variety of well-developed libraries and frameworks and is fully supported for big data and machine learning. The libraries and modules used for this study are as follow:

- Data analysis and manipulation: Pandas
- Numerical computation: NumPy
- Visualisation and plotting: Matplotlib, Seaborn, Missingno
- Date and time manipulation: Datetime
- Missing data methods: Fancyimpute
- Geospatial data: GeoPandas, Shapely
- Holidays: Holidays
- Machine learning: Scikit-Learn, TensorFlow, Keras
- Statistical models: Statsmodels, Pmdarima
- Optimisation: Hyperopt

Google Cloud Platform (GCP) was used throughout this study, specifically Cloud Storage and AI Platform. A bucket was created in Cloud Storage to upload the data files and later save the processed data for further analysis and modelling. A notebook instance was created in AI Platform with the following specifications:

- Environment: Debian 10 operating system
- Machine configuration: 8 vCPUs, 64 RAM

Jupyter notebook was the integrated development environment (IDE) used to perform the analysis. In addition, Tableau was used for visualisation to extract insights into the trip data post-cleaning using a Windows Laptop with an i5 processor and 8 GB RAM.

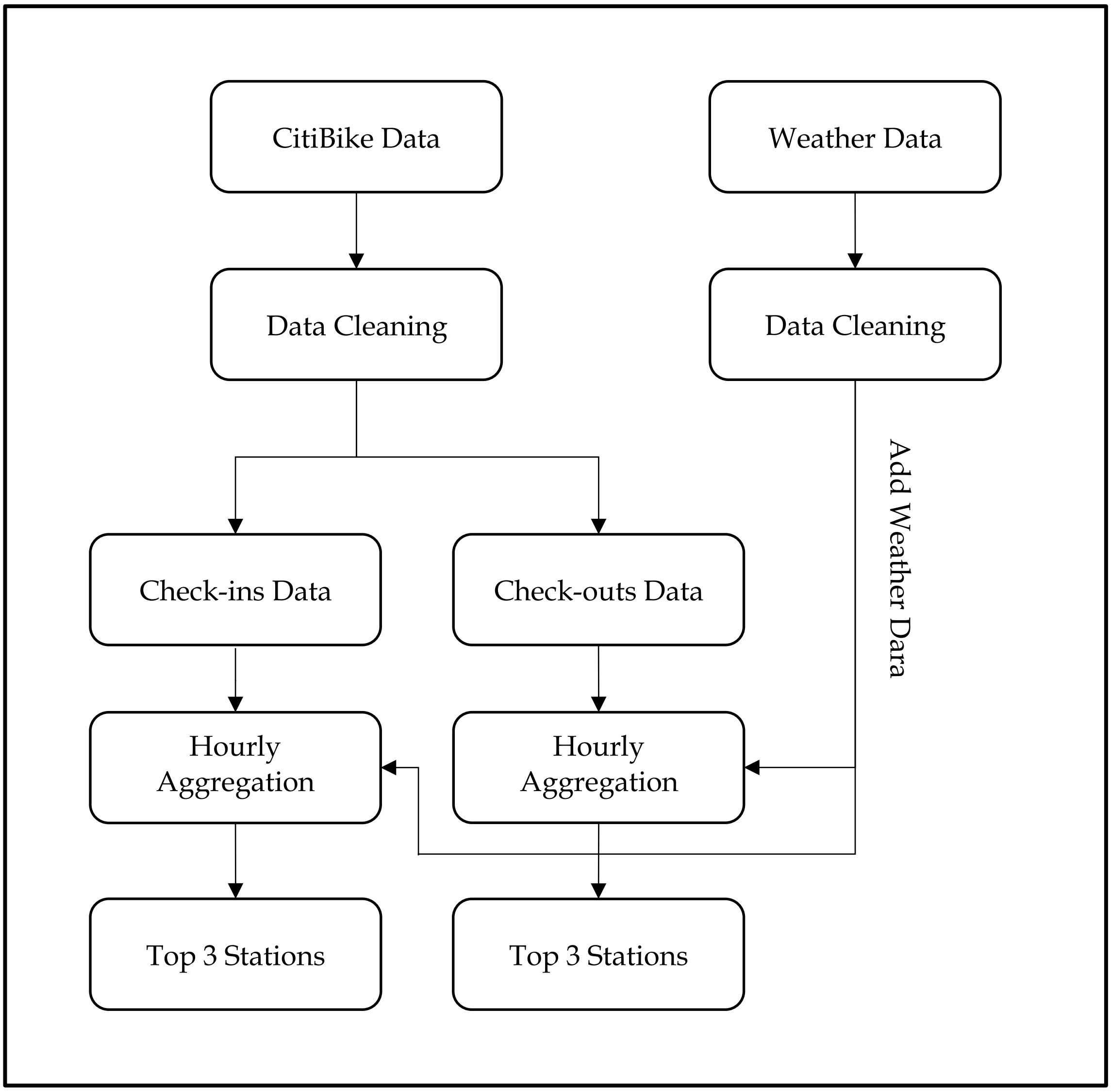


Figure 3.1 Data Pre-processing Framework

3.3.2 Data Pre-processing

The data sets obtained were broken into multiple files; 13 files for trip data and two files for weather data. Therefore, the first step was to concatenate the files before data cleaning. Some of the common data problems are missing data, duplicate rows, and column types. Both data sets were checked for these problems, with a slight variation between the two data sets given in their respective subsection.

3.3.2.1 Weather Data

The architecture of cleaning weather data is given in Figure 3.2.

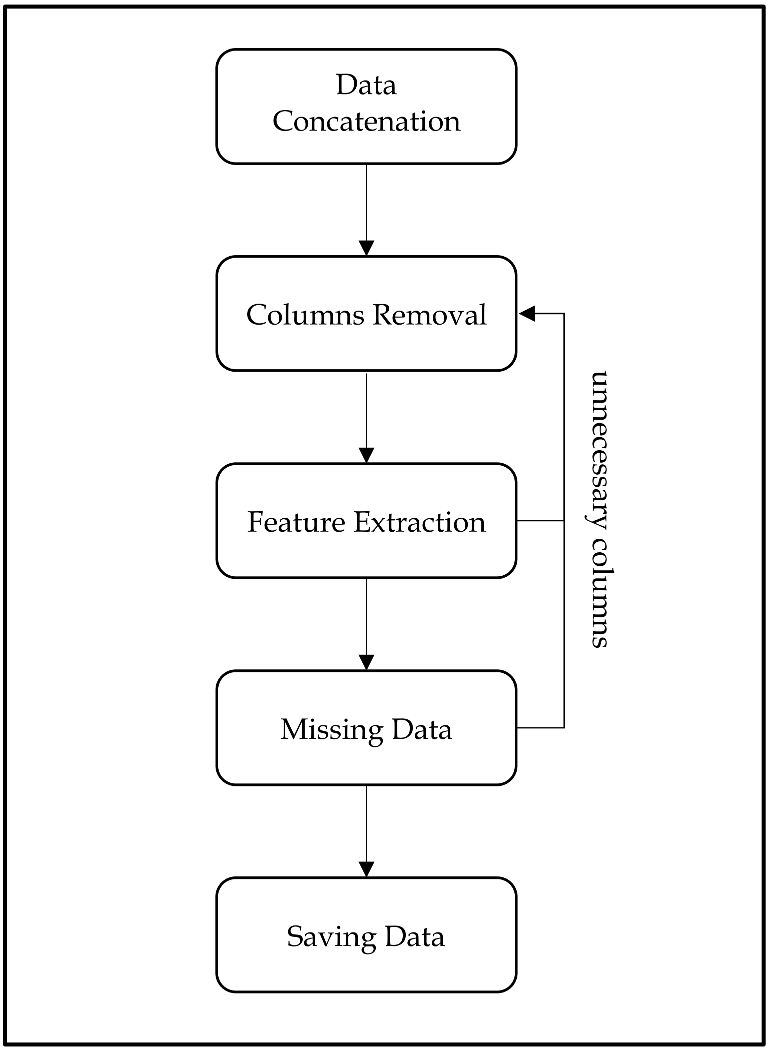


Figure 3.2 Weather Data Cleaning Architecture

Data Concatenation
The weather data consisted of two files, which were easily loaded onto two data frames and concatenated onto one data frame. The data frame consisted of 10945 rows and 27 columns.

Columns Removal
Unnecessary columns were removed upon preliminary inspection of the columns, namely, location, resolved address, wind direction, latitude, longitude, name, sea level pressure, id, info, minimum and maximum temperature, wind gust, cloud cover, and lastly, snow depth.

The remaining columns are defined as following [35]:

1. Address: New York, NY, United States

2. Date-time: refers to the date and hour
3. Temperature: the air temperature in that time period in Fahrenheit
4. Heat Index: the apparent temperature in Fahrenheit, which is a combination of air temperature and relative humidity. Generally, It is calculated when the air temperature reaches 80°F or above, and relative humidity reaches 40% or above [36]
5. Wind chill: the apparent temperature in Fahrenheit, which is a combination of air temperature and wind speed. Generally, It is calculated when the air temperature reaches 40°F or below, and wind speed is five mph or higher [37].
6. Precipitation: the amount of any form of precipitation (drizzle, rain, sleet, snow, ice pellets, graupel, or hail) fell or predicted to fall in the respective time period in inches
7. Precipitation cover: the probability of precipitation in percentage
8. Dew point: the air temperature in Fahrenheit at which a relative humidity of 100% is achieved [38]
9. Relative humidity: the ratio in
10. Visibility: the percentage of present water vapour to maximum water vapour given at the same temperature.
11. Weather type & Conditions: the state of the weather in the respective time period, weather type is more detailed and elaborate than conditions.

The date-time was set as an index to ensure a continuous and ordered time series when extracting additional features and checking for missing data.

Feature Extraction

Upon preliminary inspection of missing data, heat index and wind chill showed the highest missing values combined. Heat index and wind chill both refer to the apparent temperature with respect to either relative humidity or wind speed. Therefore, a new feature, apparent temperature, was created combining the two features using more precise equations for different temperatures, relative humidity, or wind speed conditions.

If the temperature is less than or equal to 50°F and wind speed is less than or equal to 3 mph, Equation 3.1 is used, indicating wind chill. Calculating the heat index is more complicated than wind chill; two equations are used with some adjustment made accordingly with the temperature and relative humidity measurements. Equation 3.2 is used when the temperature is higher than 80°F. If the calculated heat index is higher than 80°F, then Equation 3.3 is used instead. Equation 3.4 is subtracted from Equation 3.3 when Relative humidity is less than 13%, and the temperature is between 80°F and 120°F. On the other hand, if Relative humidity is more than 85%, and the temperature is between 80°F and 87°F, Equation 3.5 is added to Equation 3.3. [39], [40]

If the temperature, relative humidity, and wind speed did not meet any of the criteria collectively, the apparent temperature is equal to the air temperature.

$$Wind\ Chill = 35.74 + (0.6215 \times T) - (35.75 \times Wind^{0.16}) + (0.4275 \times T \times Wind^{0.16})$$

Equation 3.1 Wind Chill [39]

$$Heat\ Index = 0.5 \times (T + 61.0 + \big((T - 68.0) \times 1.2\big) + (RH + 0.094))$$

Equation 3.2 Heat Index 1 [40]

$$\begin{aligned} Heat\ Index = {} & -42.379 + 2.0401523 \times T + 10.14333127 \times RH - 0.22475541 \times T \times RH \\ & - 0.00683783 \times T^2 - 0.05481717 \times RH^2 + 0.00122874 \times T^2 \times RH \\ & + 0.00085282 \times T \times RH^2 - 0.00000199 \times T^2 \times RH^2 \end{aligned}$$

Equation 3.3 Heat Index 2 [40]

$$Adjustment = \frac{13 - RH}{4} \times \sqrt{\frac{17 - |T - 95|}{17}}$$

Equation 3.4 Adjustment 1 [40]

$$Adjustment = \frac{RH - 85}{10} \times \frac{87 - T}{5}$$

Equation 3.5 Adjustment 2 [40]

After estimating the apparent temperature, wind chill and heat index columns were comparable to the estimated values, and thus removed from the dataset.

Missing Data

Missing weather data can be due to faulty weather sensors during data acquisition, data loss, or human error, causing deletion by mistake. Moreover, missing values can be categorised into three different categories; missing completely at random (MCAR), missing at random (MAR), or missing not at random (MNAR), which implies the missingness of the variable in relation to the variable itself or other variables [41]. All the weather information is equally essential to forecast the demand; therefore, missing data techniques were applied to handle missing data.

The weather type column values were missing when the condition column value indicated 'clear' weather, which implied that there is a systematic relationship between the missingness of weather type values and the observed condition values, but not the weather type values itself satisfying the condition of MAR. Consequently, all missing data in the weather type column were set to 'clear.'

The remaining columns with missing data are of a numerical type. Analysis of the amount and randomness of the missing data were firstly carried out, then finding the best missing data technique to acquire accurate measurements were employed.

Figure 3.3 is a graphical analysis of how complete the data is. There were 10945 rows, of which 242 (2.21%) missing values of wind speed, 28 (0.26%) missing values of visibility, and 1 (0.009%) missing values of dew point and relative humidity each. To further understand the missingness of the data, the location of the missing data was visualised over time since the weather is a time-series data, as seen in Figure 3.4. Most missing values were during October 2018 and October 2019. The sparkline on the side summarised the nullity in the data. The number at the bottom is the total number of columns (11), and the number in the middle (8) is the total number of valid values in a row. Hence the maximum number of missing values in a row was 3. The graph also indicated that visibility and wind speed were MCAR; missingness had no relationship between any values. At the same time, the other missing values might correlate in terms of missingness. Figure 3.5 described the correlation between the missing values, and Figure 3.6 displayed the pattern of missingness. They both confirmed that there is a strong correlation between dew point and relative humidity; thus, dew point and relative humidity were MAR.

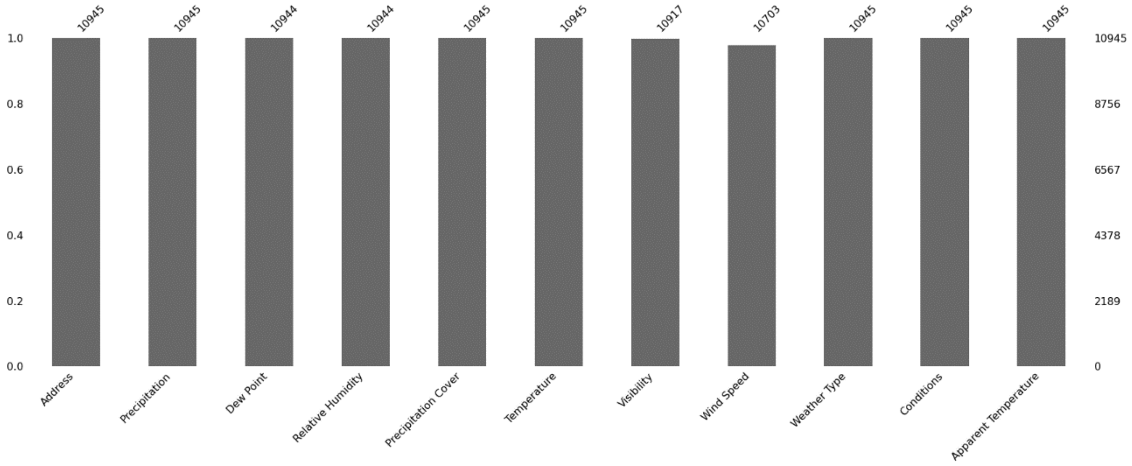


Figure 3.3 Completeness of Weather Data

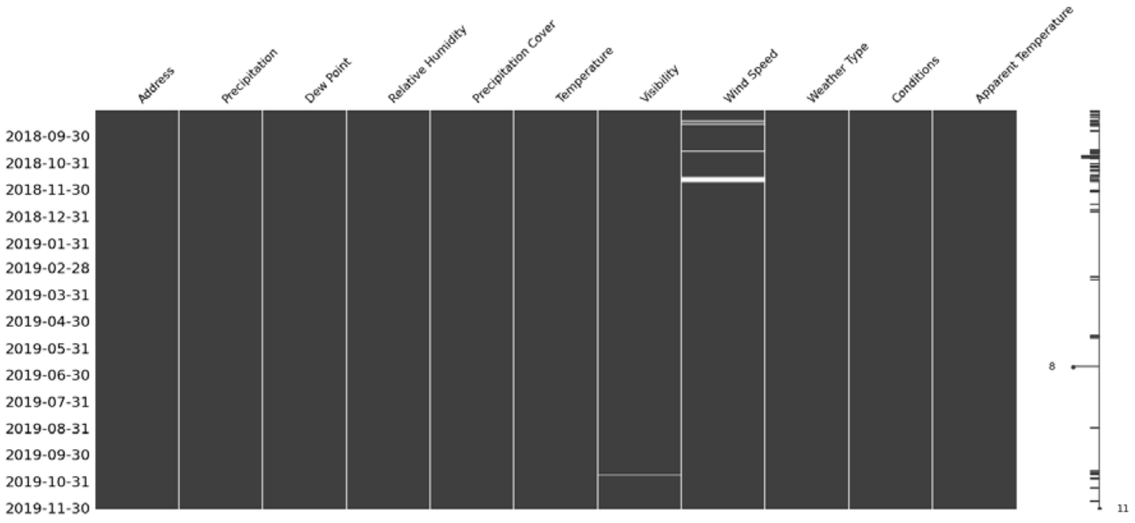


Figure 3.4 Missingness of Weather Data

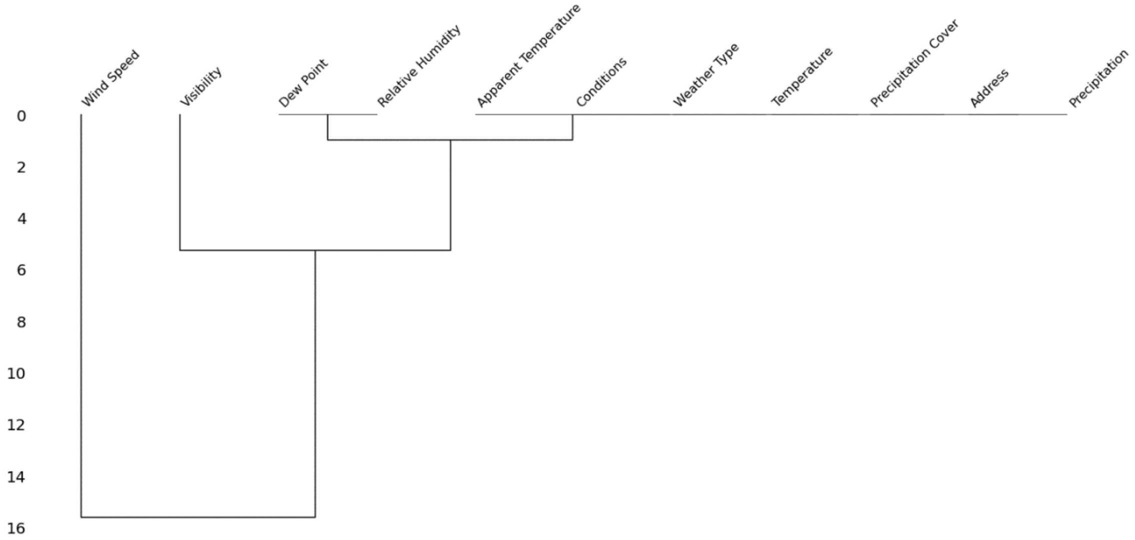


Figure 3.5 Weather Data Dendrogram

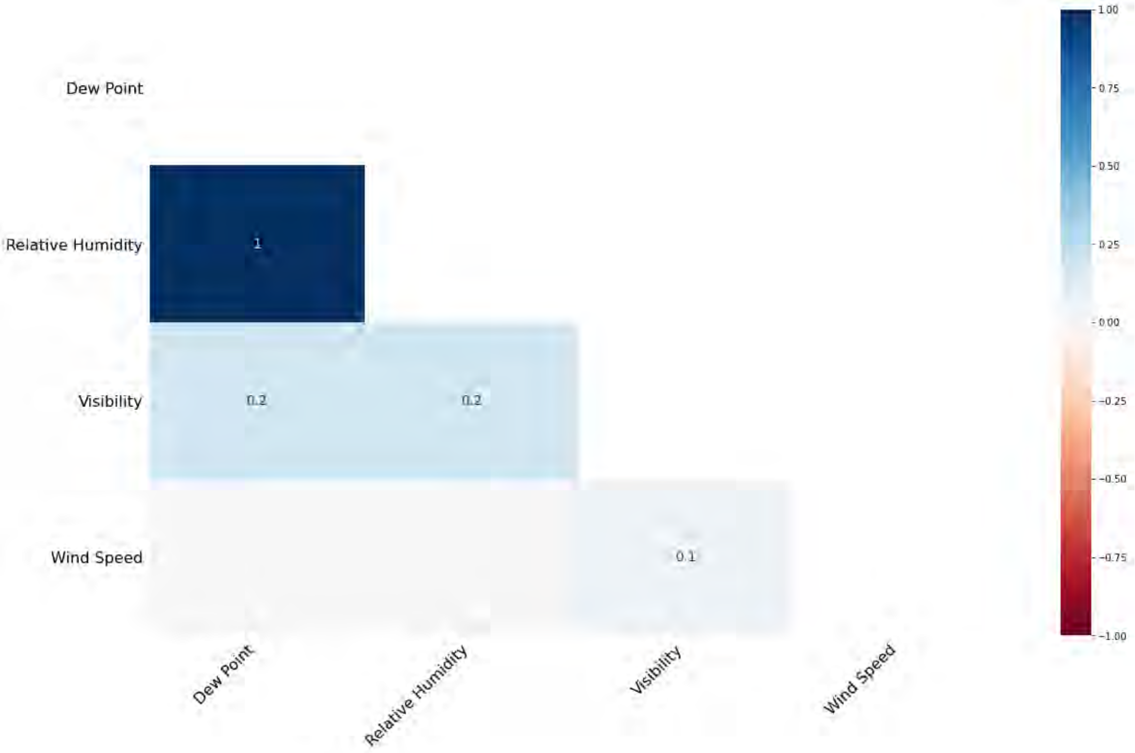


Figure 3.6 Missing Data Heat Map

Notable examples of missing data techniques are imputation, e.g., mean imputation or regression imputation, and model-based methods [41]. Three methods were used: mean imputation, KNN, and multivariate imputations by chained equations (MICE). Mean imputation substitute the missing values with the mean of the non-null values. KNN selects the K nearest non-null data points and substitutes the missing values with their average. MICE perform multiple regression over a random sample of data and substitute the missing value with their average. Overall, MICE exhibits robustness over the other imputation techniques, which is most suitable for MAR data [42].

An OLS model of the non-null values was set as a baseline to evaluate the outcome of the three methods for all four columns, respectively. Through the different variables, the three methods have the same r-squared adjusted, but by graphically examining the density plot of each (see Appendix), KNN has shown the closet resembles the shape of the original data. As a result, KNN was used to impute all the missing data.

Saving Data
The weather data was updated with the imputed data using the KNN method and rechecked for any missing data to ensure correct imputation. Then, the index was reverted into a column and reset. Lastly, the data was saved into a CSV file.

3.3.2.2 Trip Data

The architecture of cleaning and pre-processing trip data is given in Figure 3.7.

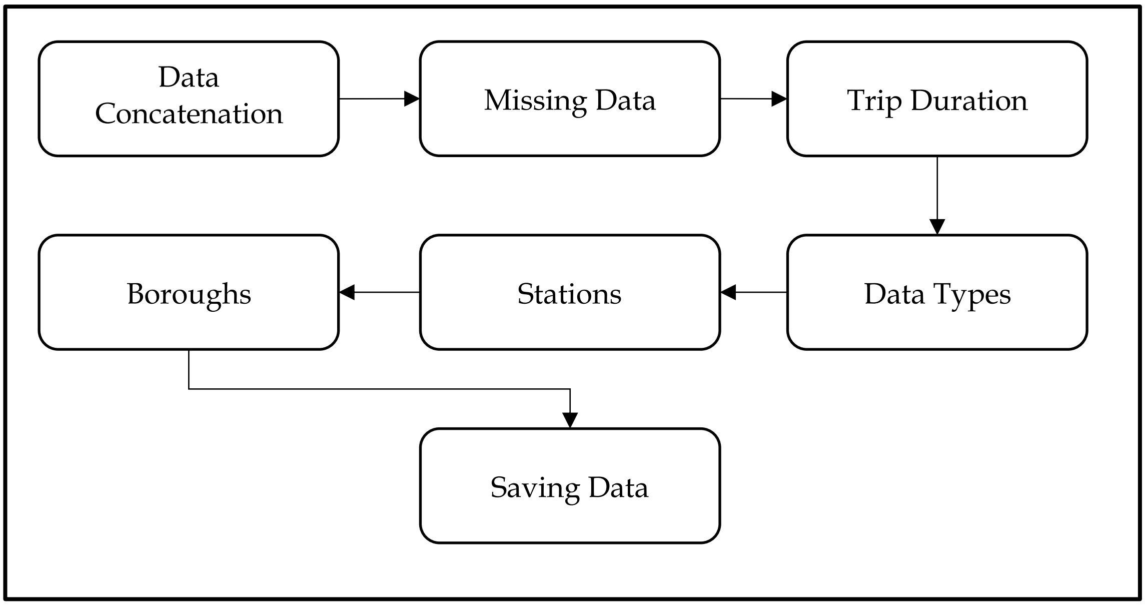
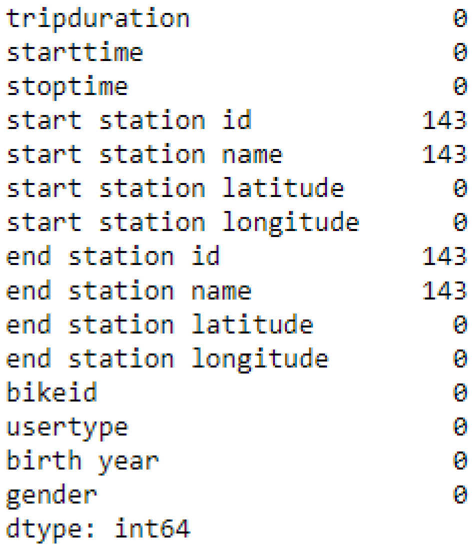


Figure 3.7 Trip Data Cleaning and Pre-processing Architecture

Data Concatenation
The trip data consisted of multiple files; loading each file separately into a data frame is tedious. Globbing is the process of pattern matching for file names using a wild card, '*' or '?'. Hence, the trip data were matched with *.csv using globbing, returning a list of the trip files names. The list was then concatenated to a data frame and saved as a new CSV file.

Missing Data
The trip data consisted of 21873141 rows, of which 143 contained null values, as seen in Figure 3.8. The longitudes/latitudes of the null values were inconsistent and did not match with any of the current stations' longitudes/latitudes, which indicated faulty entries. In conclusion, the rows with missing values only accounted for 0.0006%, which is insignificant and were removed.



```
tripduration                 0
starttime                    0
stoptime                     0
start station id           143
start station name         143
start station latitude       0
start station longitude      0
end station id             143
end station name           143
end station latitude         0
end station longitude        0
bikeid                       0
usertype                     0
birth year                   0
gender                       0
dtype: int64
```

Figure 3.8 Trips Missing Data

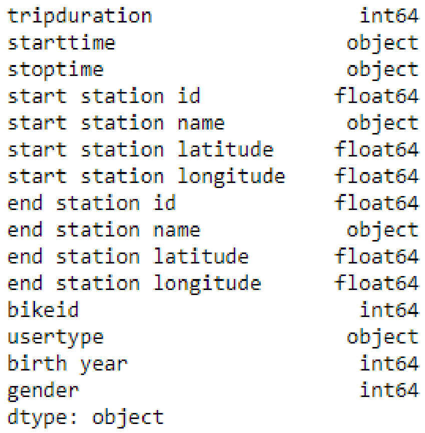
Trip Duration
As per mentioned in Section 3.1 above, the first 45 minutes and 30 minutes of each ride are included for annual members and one-time users, respectively. Users are charged an additional cost for exceeding these limits. Therefore, an assumption has been made that users would not use the service for more than 2 hours as the cost increases exponentially.

Trips lasting longer than 2 hours could indicate a stolen or faulty bike or a bike that was not appropriately docked. Additionally, stations are distributed within a 2 minutes bike ride [43]. Thus, trips that are less than 2 minutes could indicate an incomplete trip or the time it took the user to undock and dock the bike back.

The data were checked for trips that lasted more than 2 hours (7200 seconds) and less than 2 minutes (120 seconds). There were 59572 trips longer than 2 hours and 375661 trips less than 2 minutes, approximately 2% of the data. These trips were considered anomalous and were removed.

Data Types
Data type is a set of values that define the classification of the data [44]. The data type should indicate the correct format of the values in each column. Accordingly, ensuring that columns are set to the correct data type is a crucial part of data cleaning to create an accurate model. For example, numeric columns can be strings or vice versa. Additionally, categorical data should be converted to 'category' data type, which in return, make the dataset smaller in memory and more uncomplicated to utilise in further analysis.

Figure 3.9 shows the data types of columns before cleaning. The first step in this process was to convert the start/end into date-time. Prior to converting gender and user type into 'category,' the gender numerical values were mapped to indicate their actual values. Finally, the start/end station name was converted to 'category' to save memory.



```
tripduration                 int64
starttime                   object
stoptime                    object
start station id           float64
start station name          object
start station latitude     float64
start station longitude    float64
end station id             float64
end station name            object
end station latitude       float64
end station longitude      float64
bikeid                       int64
usertype                    object
birth year                   int64
gender                       int64
dtype: object
```

Figure 3.9 Trips Data Types

Stations
This step was adopted to assess the stations' names and ids and their unique values. It also complemented the Trip Duration step. First, station information was gathered from multiple resources from 2017 till 2020 [45]–[47] and loaded as a separate data frame. Once the names and ids were extracted from the station data frame, they were compared with the names and ids in the trip data. The results of the comparison have shown that the naming of the stations may slightly differ over time in the station data but updated in the recent data and the trip data. Also, some stations have been removed or relocated, thus having a different longitude and latitude. Overall, these modifications did not affect the results of the analysis and modelling.

Lastly, trips with a duration of 2 minutes (120 seconds) were checked if they started and ended at the same station, which resulted in none.

Boroughs
This step was used to extract additional features from the data (borough) to ensure that the trip data obtained were within New York City only. Firstly, A shapefile of the boroughs' boundaries was acquired [48]. Secondly, the unique values of the stations with their latitude/longitude were extracted from the trip data. A dictionary was created combining the information from the shapefile and the station data, which included each station id (key) and its distinctive borough (value).

New features were added to the trip data; start borough and end borough, mapping the information from start/end station columns. Start/end borough columns were checked for nullity. The end borough column contained 107 null values, which upon further investigation, revealed that those trips ended in New Jersey. Given that the sum was trivial, they were dropped.

Saving Data
The cleaned trip data with their newly added attributes were saved into a CSV file and further analysed in section 0

3.3.2.3 Data Aggregation

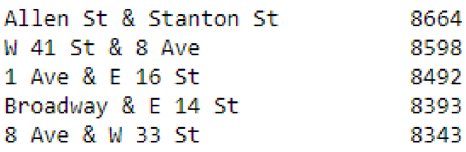
This process was repeated twice for hourly check-ins and check-outs trips separately. Prior to commencing the data aggregation, a 'merge' column was derived from the start/stop time column comprising the date and the hour only. An 'hour' column was then derived from the 'merge' column. Both columns were converted to date-time types.

The trip data were then grouped by start/end station id, merge, and user type columns and aggregated based on the 'hour' column. The resulting data frame was pivoted to get separate columns for customers and subscribers. Then, the null values, which indicated no users, were replaced by zero.

Time attributes were extracted from the resulting data frame. First, the month, day columns were created. The day column was mapped to show the name of the day instead of numbers. Once the day column was set, the weekend column was created, referring to whether the day was a weekend (Saturday and Sunday) or not. Then, a holiday column was added using the 'holidays' library to get New York City holidays. Afterwards, a working day column was created as well to indicate the day was neither a holiday nor a weekend. Lastly, a season column was added with respect to the official start and end dates of each season in the United States [49].

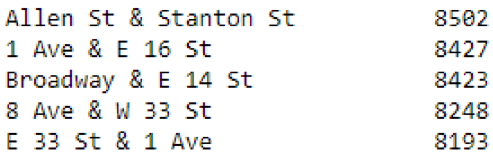
Once all the time attributes were extracted, the weather data was loaded and prepared for merging. The weather information was already hourly based; therefore, only a 'merge' column was added to set for merging with the trip data. After merging, a 'count' column was created, summing the values of customers and subscribers. Lastly, all the unnecessary columns were dropped.

Finally, the top 3 stations with the highest count of hourly demand in both datasets were extracted for modelling, as seen in Figure 3.11 and Figure 3.10.



```
Allen St & Stanton St          8664
W 41 St & 8 Ave                8598
1 Ave & E 16 St                8492
Broadway & E 14 St             8393
8 Ave & W 33 St                8343
```

Figure 3.11 Top Check-outs Stations



```
Allen St & Stanton St          8502
1 Ave & E 16 St                8427
Broadway & E 14 St             8423
8 Ave & W 33 St                8248
E 33 St & 1 Ave                8193
```

Figure 3.10 Top Check-ins Stations

3.4 Data Description & Visualisation

This section moves on to describe in greater detail the CitiBike Bikesharing system and the final data used for modelling. The following is considered under the umbrella term exploratory data analysis (EDA).

3.4.1 Users

Figure 3.12 presents the gender distribution across the system, and what can be clearly seen is the high percentage of male riders compared to females rides, almost three times more. It shows that men are more likely to use the service than women.

The majority were between the ages of 25 and 35, yet high rates can also be noticed between the age of 35 up to 50 (Figure 3.13). It is also noted that men between the age of 35 and 42 tended to use the service more than women of the same age range, as seen in Figure 3.14 and Figure 3.15. The figures only show ages up to 75, but the actual data contained ages above 75 up to 135, which are considered outliers or inaccurate data. Nevertheless, the trips were legit; it may indicate that users were not willing to share their age.

Approximately 90% of the users were subscribers; it can be deduced that most users opted to become members (Figure 3.16). What is striking in Figure 3.17 is the general pattern of usage of different user types during the week. Subscribers used the service more during the weekday rather than on the weekends, whereas customers were more likely to use it on the weekends, concluding that subscribers are more likely to be employees or students.

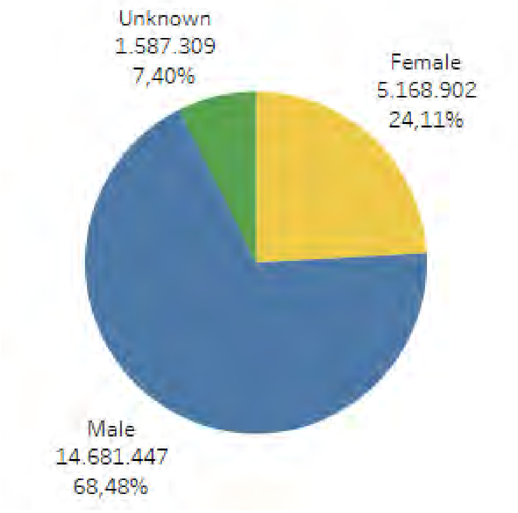


Figure 3.12 Gender Distribution

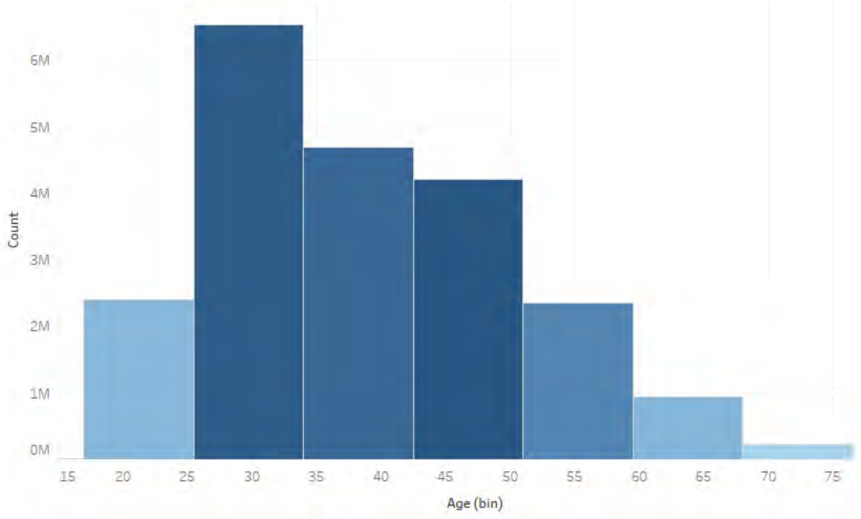


Figure 3.13 Age Distribution

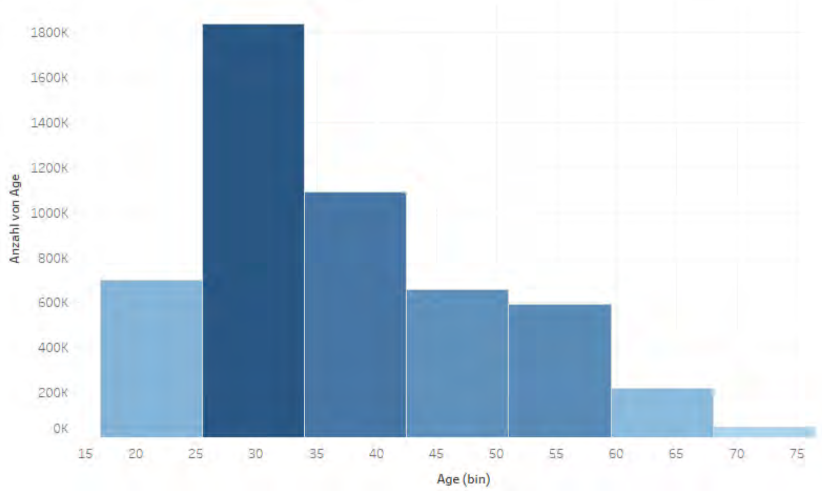


Figure 3.14 Age Distribution by Gender (Female)

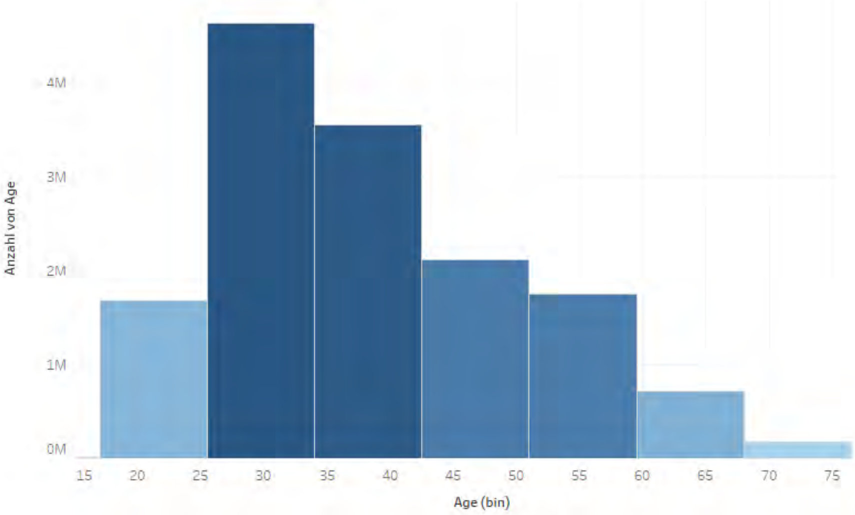


Figure 3.15 Age Distribution by Gender (Male)

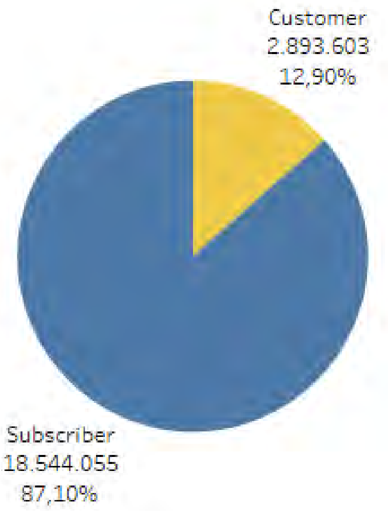


Figure 3.16 User Type Distribution

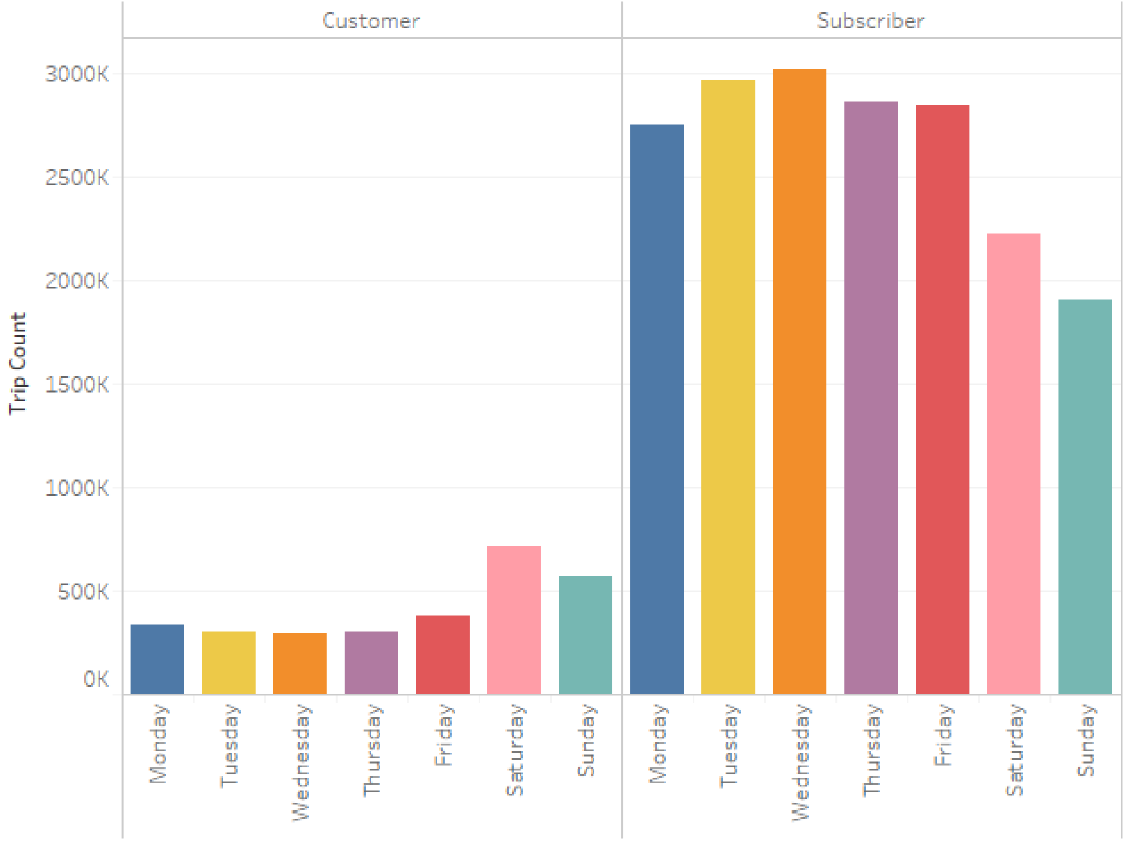


Figure 3.17 User Type Distribution (Weekday)

3.4.2 Stations

The location of the station is one of the main factors that influence the demand immensely [50]. Both the top 10 start and end stations are located in Manhattan, as seen in Figure 3.18 and Figure 3.19. Manhattan is the highest populated borough in New York City, which explains the high volume of demand and stations. They are located around a lot of touristic attractions, offices, and universities, thereby explaining the usage pattern of subscribers.

Pershing Square North station is a good illustration with the highest demand for starting and ending trips. It is located in front of Grand Central Terminal, which is excellent evidence of bike-rail intermodality and potentially provides a solution for the last mile problem. It is also a key access point to historical and touristic attractions, restaurants, and shopping areas.

E 17 & Broadway is the second-highest demand station for starting and ending trips as well. It is located next to the famous Union Square, a pedestrian plaza and park. It is also notable for its landmarks, green markets and businesses, and its attraction for street artists and protesters.

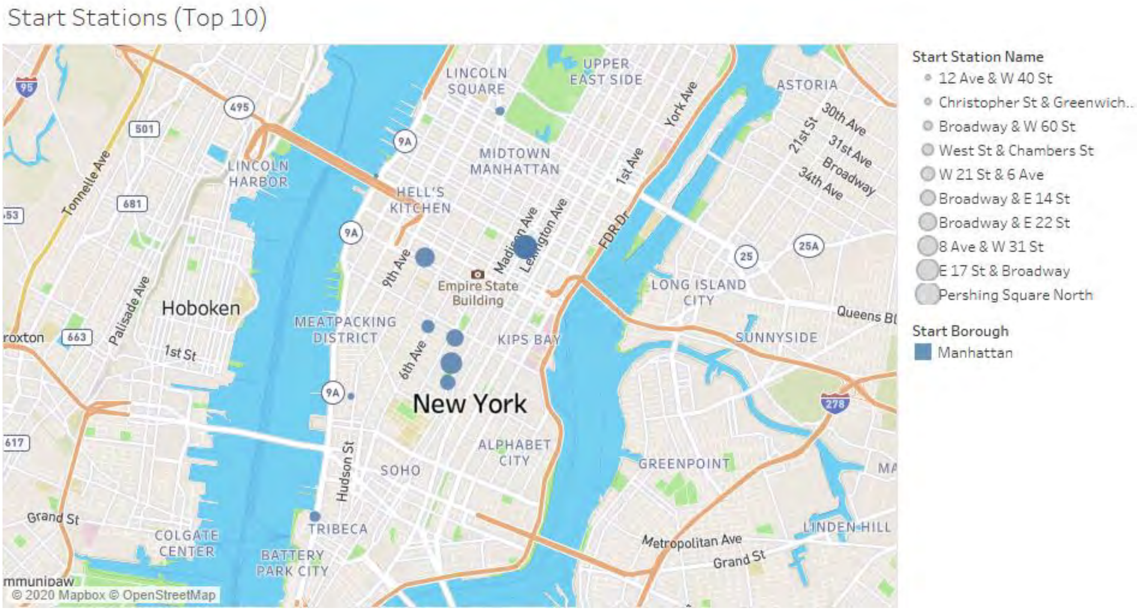

Start Stations (Top 10)

Figure 3.18 Top 10 Start Station Locations

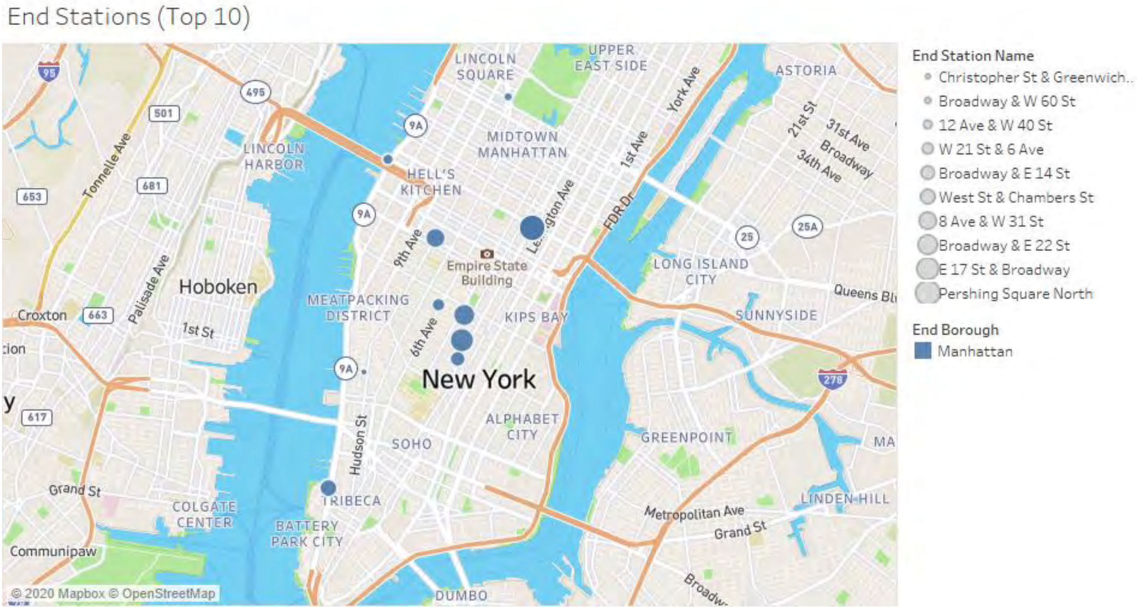

End Stations (Top 10)

Figure 3.19 Top 10 End Station Location

By contrast, the bottom stations are located away from the central New York City, mostly in Queens and Brooklyn. All bottom stations in Brooklyn and Queens are located in the periphery of the system. Bressler station is the only station that is now closed, but the others are still existing as Citi Bike studies its expansion. (Figure 3.20 and Figure 3.21)

Manhattan is highly packed with stations; therefore, it is understandable that some stations are more attractive than others, causing the closure of the least attractive. NYCBS Depot – Delancey, 58th St Depot, W 39 St & 9 Ave stations are currently closed.

On the other hand, University Pl & E 14 St station was relocated. (Figure 3.20 and Figure 3.21)

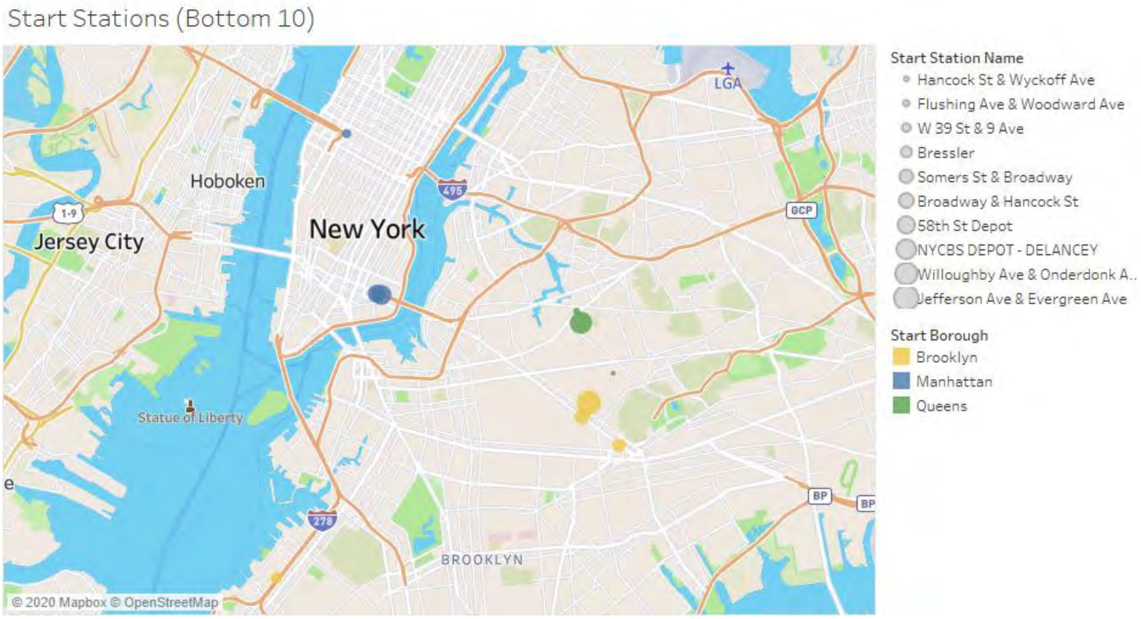


Figure 3.20 Bottom 10 Start Station Locations

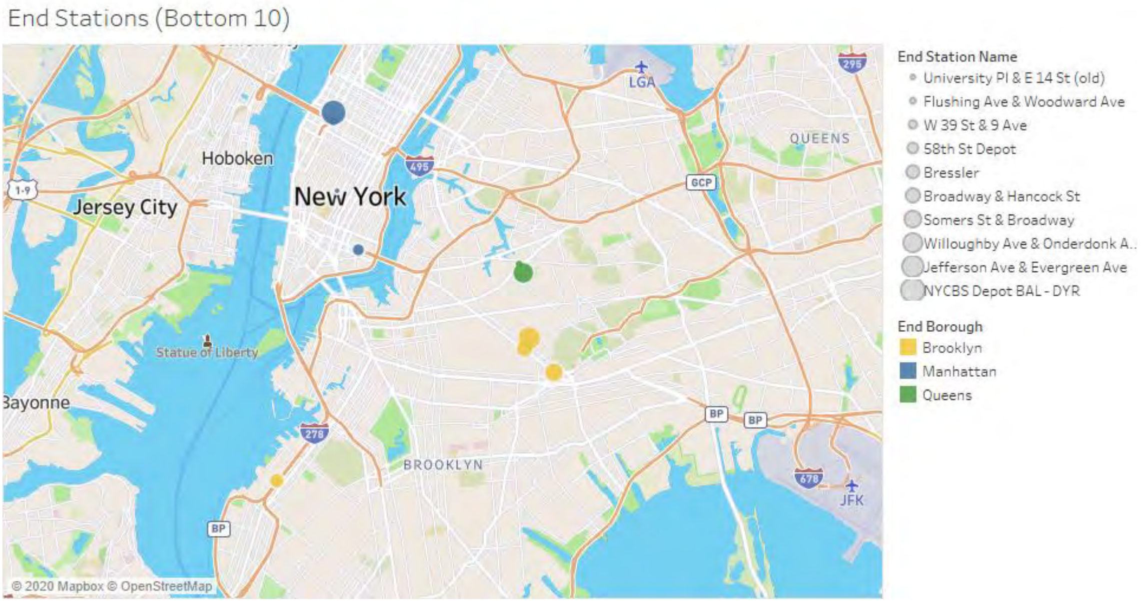


Figure 3.21 Bottom 10 End Station Location

3.4.3 Trips

As shown in Figure 3.22, the trip duration distribution is skewed to the right, with the mode being around 5 minutes. Trip durations spread from 2 minutes up to 120 minutes, which is explained by the data cleaning process. The highest density can be observed between 2 and 9 minutes.

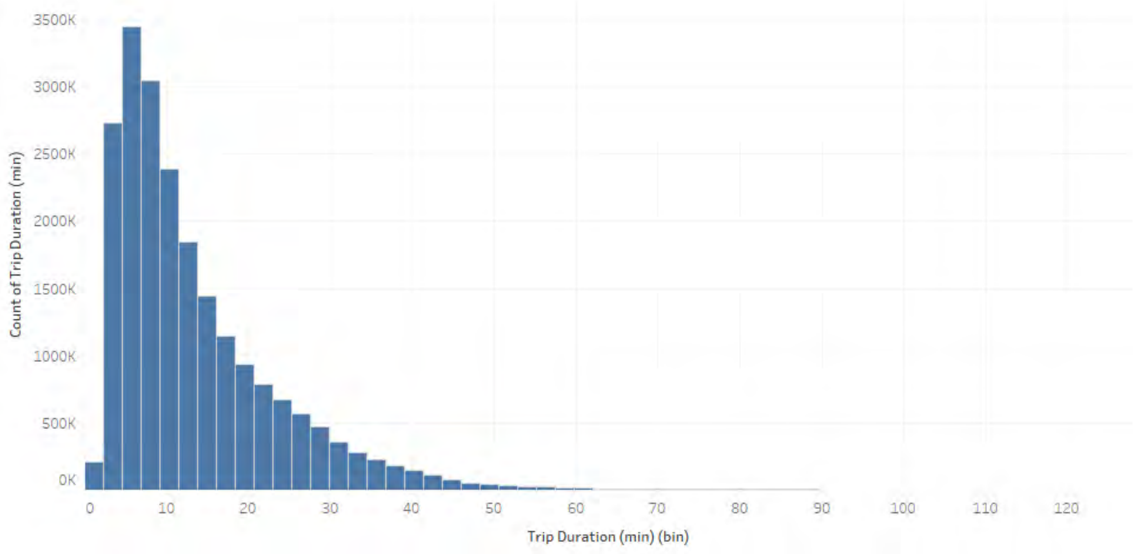


Figure 3.22 Trip Duration Histogram

Closer inspection of trip duration trend by months, Figure 3.23 shows that there has been a steady decline from November 2018 to January 2019, a gradual increase reaching its peak in June 2019, then it drops again from September 2019. The variability of trip duration is most likely attributed to weather conditions, as bike users tend to cycle more in warm conditions. Figure 3.24 confirms it; the same pattern can be seen, the gradual decline from January to February, then the continual growth reaching a peak in September.

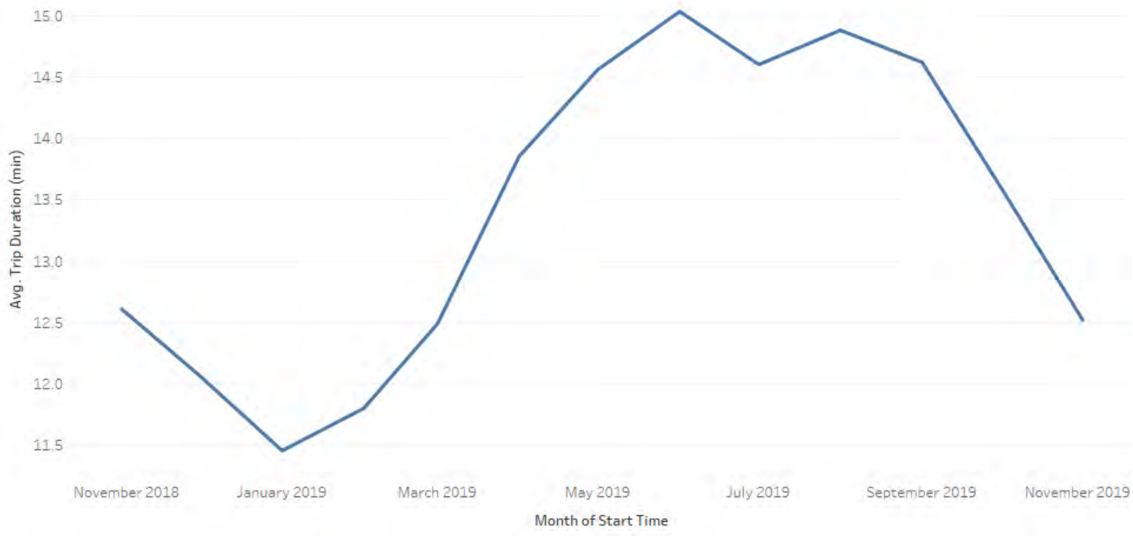


Figure 3.23 Trip Duration Trend (Monthly)

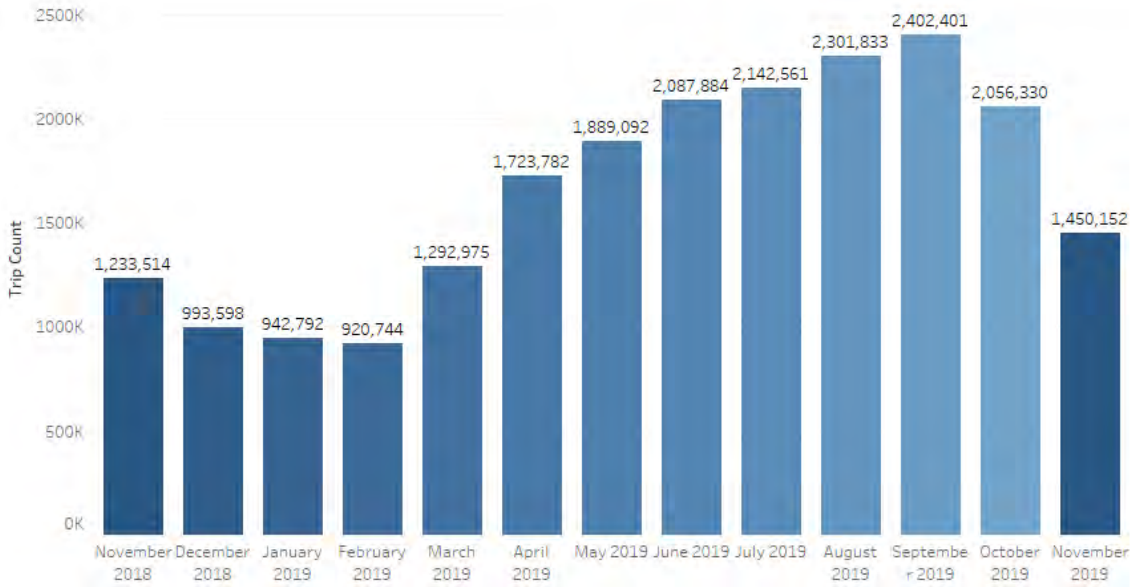


Figure 3.24 Monthly Trip Count

Figure 3.25 and Figure 3.26 shows that there are two peak periods during the weekdays: 8 am and 5-6 pm, which are most likely influenced by the subscribers' proportion and indicates work/education trips.

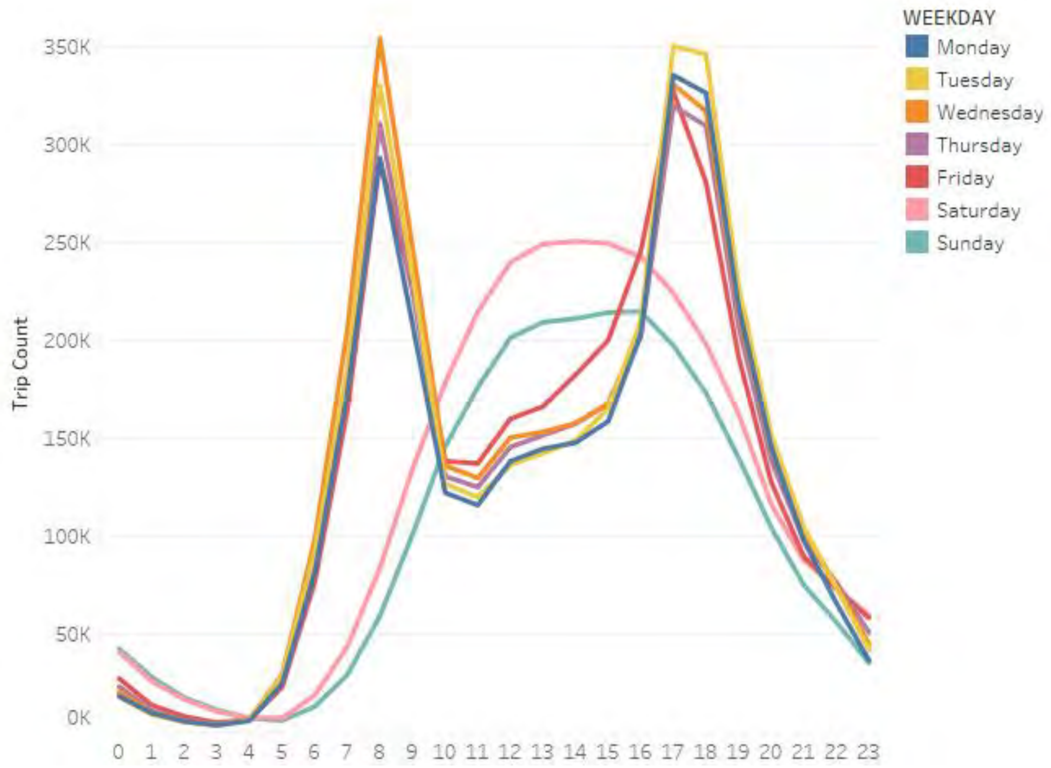


Figure 3.25 Daily Trip Count – Start

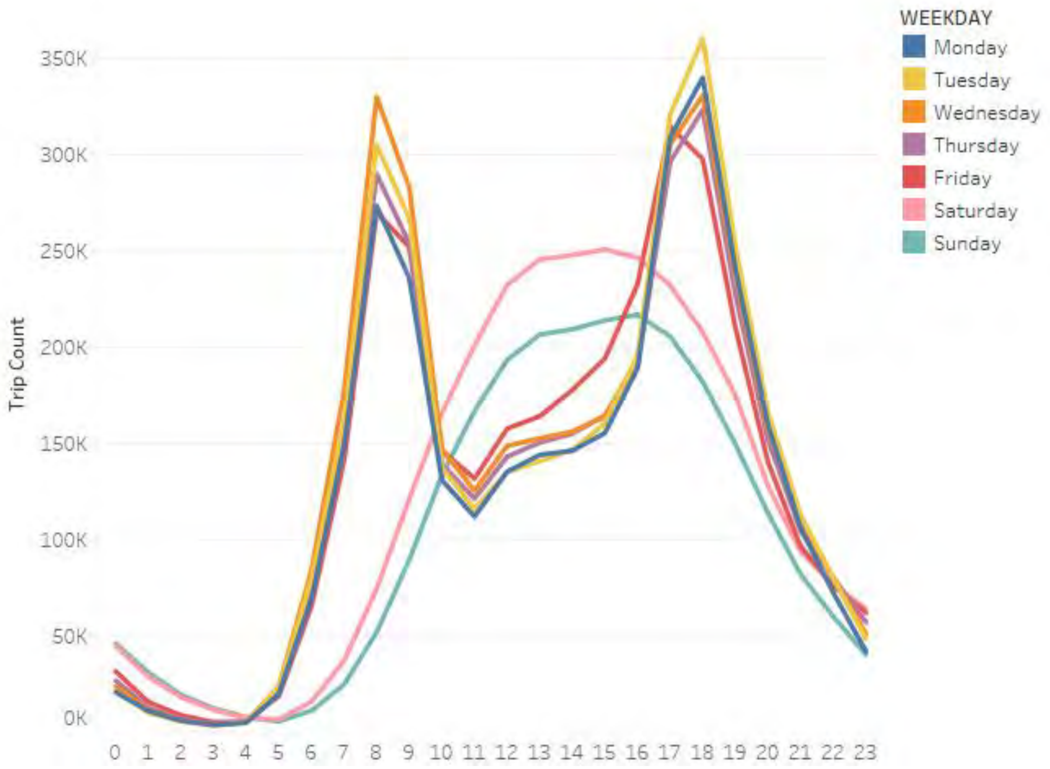
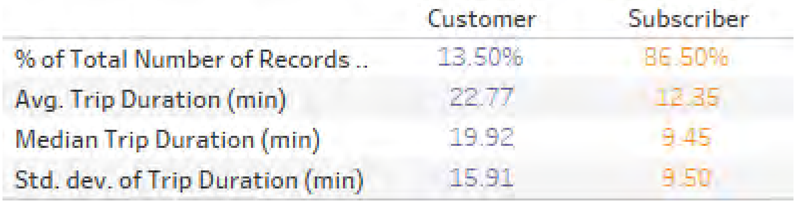


Figure 3.26 Daily Trip Count – End

Table 3.1 infers the trip duration statistical according to the user type. It shows that customers usually made longer trips than subscribers, and they have a larger standard deviation from the average. The percentage shows that they are not regular, and most likely used by tourists or on holiday.

Finally, plotting the trip duration across different ages, it is clearly seen that users at a younger age (16-25) made longer trips than older users. What stands out is the sharp peak at age 50; they made two times longer trips than regular users between the age of 35 and 70. (Figure 3.27)



	Customer	Subscriber
% of Total Number of Records ..	13.50%	86.50%
Avg. Trip Duration (min)	22.77	12.35
Median Trip Duration (min)	19.92	9.45
Std. dev. of Trip Duration (min)	15.91	9.50

Table 3.1 Trip Duration Summary

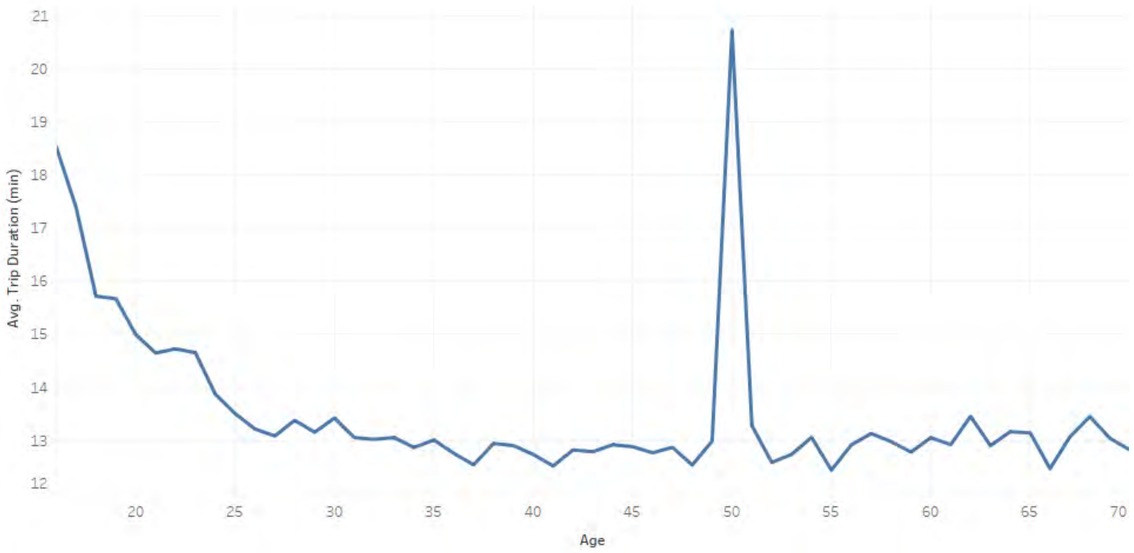
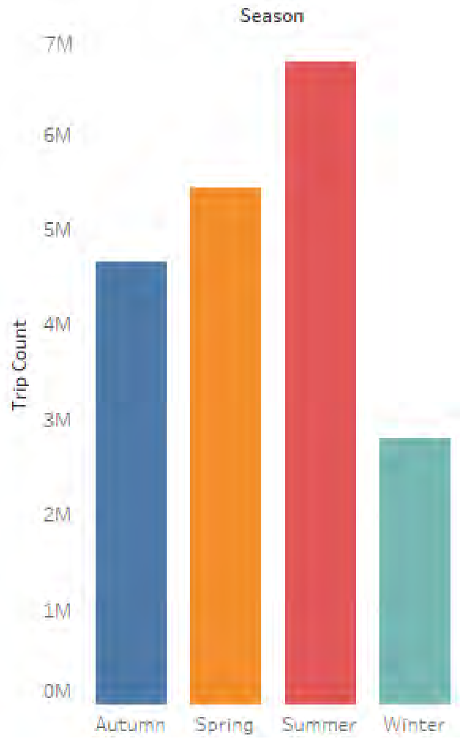


Figure 3.27 Average Trip Duration by Age

3.4.4 Weather-Related

Bike-sharing systems are highly affected by seasons. The summer has the highest trip count compared to other seasons, followed by spring then autumn (Figure 3.29). Customers were explicitly influenced by season when compared to subscribers, with a significantly smaller proportion in winter than subscribers (Figure 3.28).



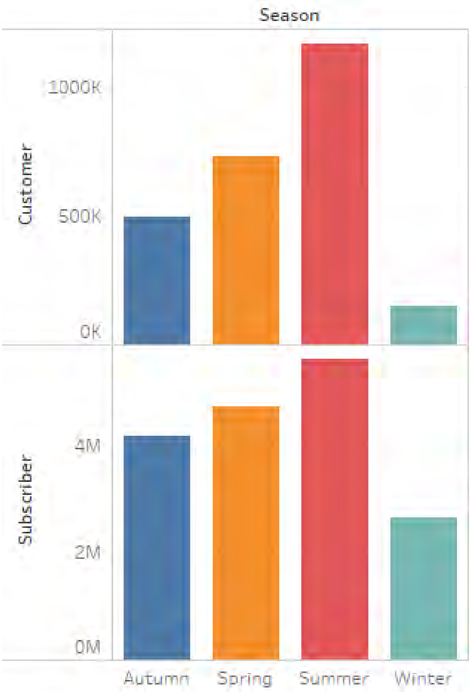


Figure 3.29 Trip Count by Seasons

Figure 3.28 Trip Count by Season (User-Specific)

Figure 3.30 further confirms that the weather influenced the users' usage patterns. The amount the trips made were remarkably high in clear weather condition and significantly decreased with the condition getting worse.

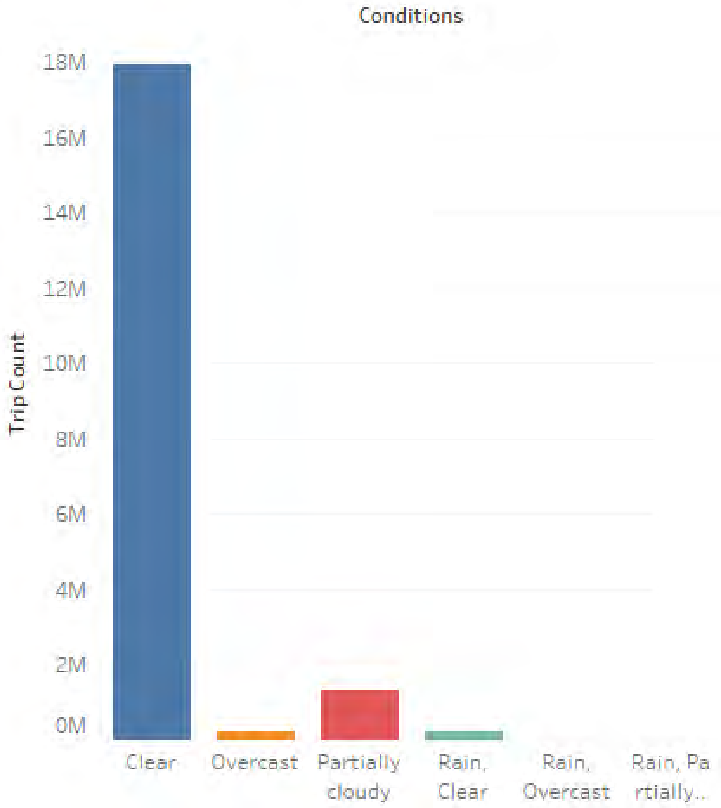


Figure 3.30 Trip Count by Weather Condition

Figure 3.31 reveals that there has been a steady rise in the number of trips made when the temperature increased. It shows that users preferred warmer temperatures than the colder temperatures. What strikes is the dramatic decline in the trip count when the temperature exceeded a certain degree.

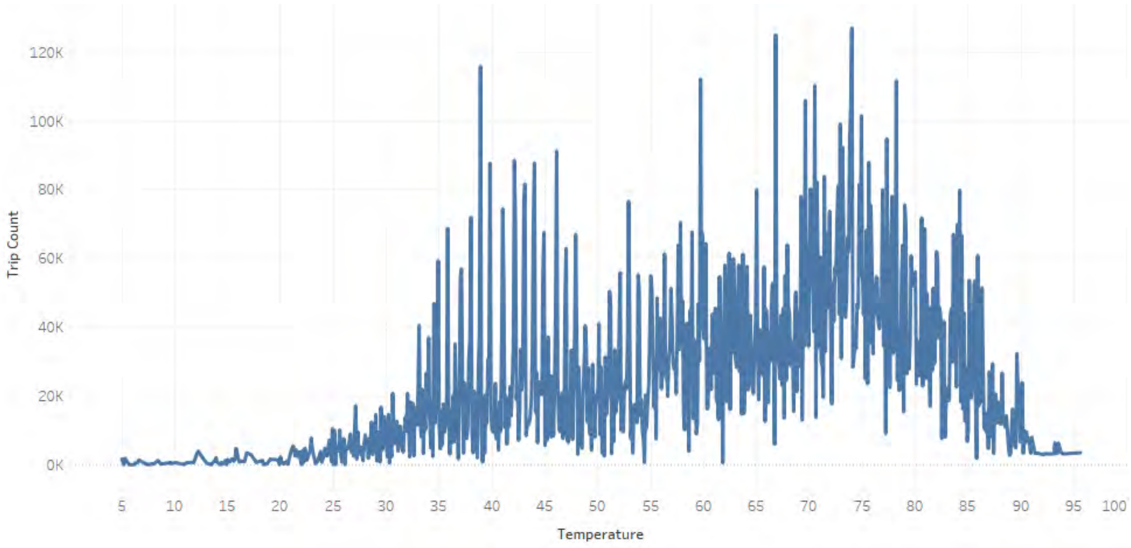


Figure 3.31 Average Trip Count by Temperature

Other weather conditions, when combined with the temperature, also greatly influence the trip count. When the relative humidity reached a certain degree outside the human comfort zone, the trip count decreased (Figure 3.32). Similarly, more trips were made when the wind speed was within a manageable level; then, they decreased as the wind speed increased (Figure 3.33).

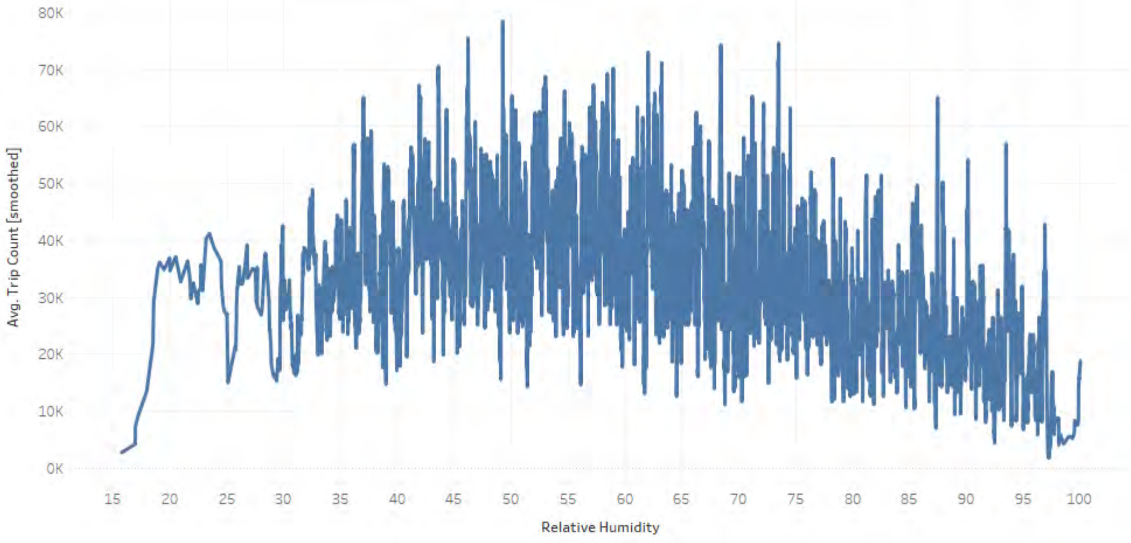


Figure 3.32 Average Trip Count by Relative Humidity

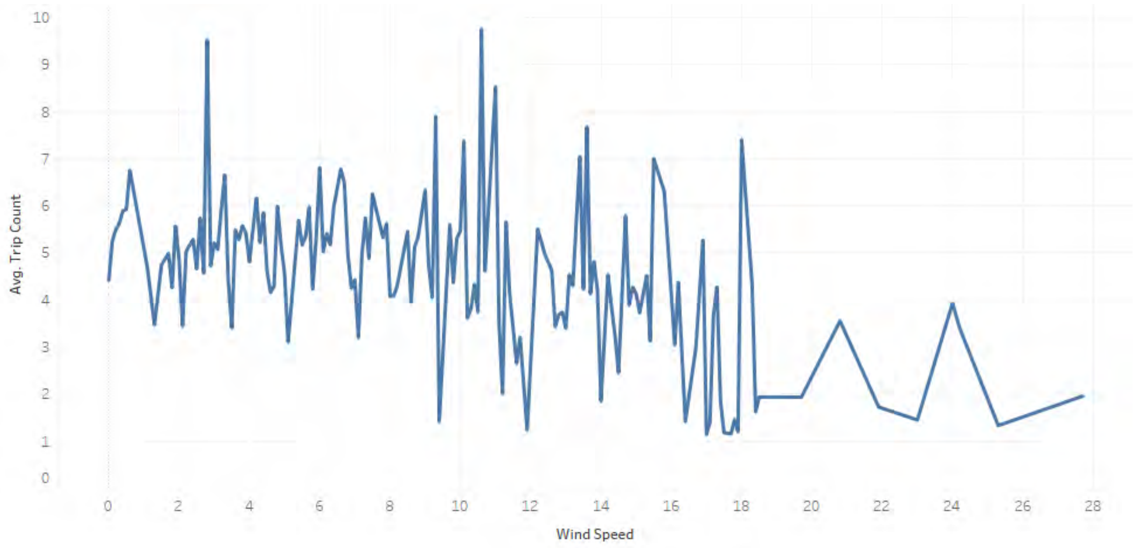


Figure 3.33 Average Trip Count by Wind Speed

More visualisations are in the Appendix.

3.5 Data Modelling

This section presents the different models used in this study, their hyperparameter, and the different algorithms used to find the best set of hyperparameters, and finally, their implementation.

3.5.1 Random Forest

In 2001, *Leo Breiman* introduced the Random Forest algorithm [51]. It is an ensemble machine learning algorithm that uses Classification and Regression Trees (CARTs); hence it can be used either as a classification or a regression model. RF fits several trees by selecting a random subset (bootstrap sample) of the predictors from the original data and outputs a classification or a regression prediction. The final output depends on the original problem definition, whether it is a classification or a regression problem; it is either the mode of the classes for classification or the mean prediction for regression. RF is less prone to overfitting due to the Law of Large Numbers (LLN) [51], as a result of training multiple trees simultaneously. This theorem states that the output from multiple training is more likely to be close to the expected value and thus obtaining a smaller variance compared to an individual model.

The data used for training an RF model do not require normalisation or pre-processing; the latter depends on the programming language and library used. The number of trees trained is a hyperparameter that can be tuned to obtain an optimal model. Further hyperparameters are discussed in subsection 3.5.5

3.5.2 Gradient Boosting Regression Tree

Gradient Boosting Regression tree is another ensemble machine learning algorithm based on the work of *Leo Breiman* and developed by *Friedman* [52]–[54]. It can be used for both classification and regression problems. It behaves similar to RF in the manner of fitting multiple trees, but it instead fits them in a sequential manner. The trees in Gradient Boosting are considered weak learners individually, and it builds a robust model by trying to minimise the errors of the previous trees each time a tree is added. Thus, attributing to a longer training time than RF.

Similarly, the training data does not require normalisation or pre-processing; the latter also depends on the library used. GBRT has fewer hyperparameters than RF involved in tuning. Further details on hyperparameter optimisation supplied in subsection 3.5.5

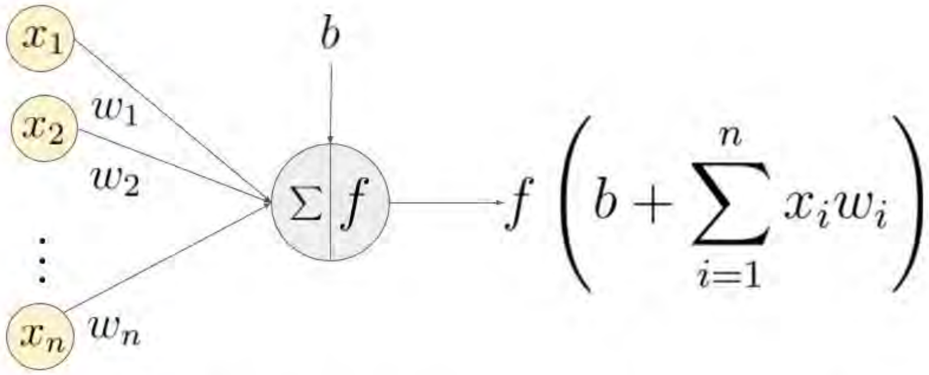
3.5.3 Artificial Neural Network

Artificial Neural Networks (ANNs) are computational models that try to replicate the way the human brain process information. ANN consists of multiple neurons, which are the processing units and interconnected by nodes. A simple of ANN consists of a single neuron i (also called Perceptron) that receives the sum of multiplication of n inputs $(x_1, x_2, \ldots, x_n)$ by their weights $(w_1, w_2, \ldots, w_n)$, in which a bias is added, then an activation function is applied to get a single output, as seen in Equation 3.6 and Figure 3.34.

$$f(x) = b + \sum_{i=1}^{n} x_i w_i$$

Equation 3.6

Different ANNs have different topology according to the problem definition; consequently, the activation function to be used—further discussion of activation functions in subsection 3.5.3.2.



An example of a neuron showing the input (x_1 - x_n), their corresponding weights (w_1 - w_n), a bias (b) and the activation function f applied to the weighted sum of the inputs.

Figure 3.34 ANN Example [55]

3.5.3.1 Multilayer Perceptron (MLP)

Multilayer Perceptrons (MLP) with Back Propagation (BP) learning algorithm is a kind of feedforward artificial neural network, also called a multilayer feedforward neural network. MLP consists of three layers: an input layer, a hidden layer, and an output layer, all consisting of non-linear (linear for the output activation function in case of regression) activated neurons, where each layer only affects the layer on the right, hence the name feedforward. (Figure 3.35) [56]

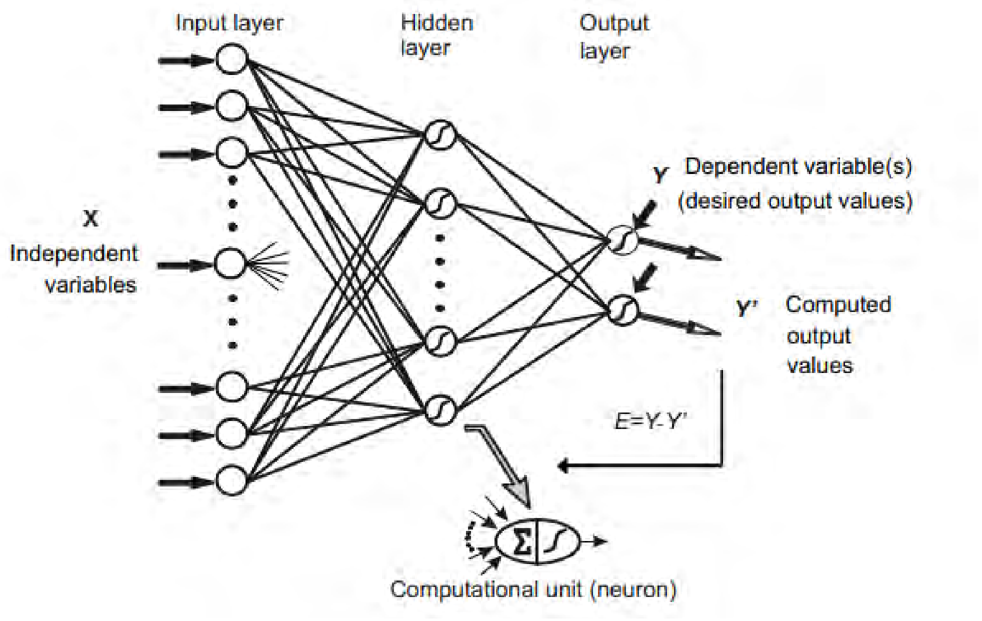

Input layer
Hidden layer
Output layer
Y Dependent variable(s) (desired output values)
X Independent variables
Y' Computed output values
E=Y-Y'
Σ
Computational unit (neuron)

Figure 3.35 MLP Example [56]

An MLP with a BP learning algorithm tries to minimise the cost function; obtaining a closer predicted value to actual value, by adjusting the initial weights used in the feedforward step by means of backpropagation.

3.5.3.2 Activation functions

Activation functions are used in neural networks to produce an output from the previous layer, which is used as an input for the second layer. The activation function used relies on the problem definition, whether it is a regression or a classification problem.

The following activation functions were used in this thesis, which are the most popular ones used for a regression problem:

- Linear; the output is not confined by any range. (Figure 3.36)

$$f(x) = x, (-\infty, \infty)$$

- Rectified Linear Unit (ReLu): returns a positive output. (Figure 3.37)

$$f(x) = \begin{cases} 0\ for\ x \leq 0 \\ x\ for\ x > 0 \end{cases} = \max\{0, x\}$$

- Softplus: a variation of ReLu, it is the smooth version of ReLu. (Figure 3.38)

$$f(x) = \ln(1 + e^x)$$

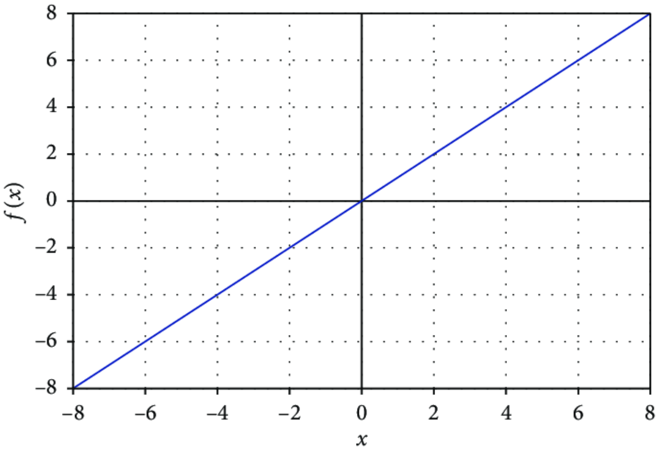


Figure 3.36 Linear Activation Function [57]

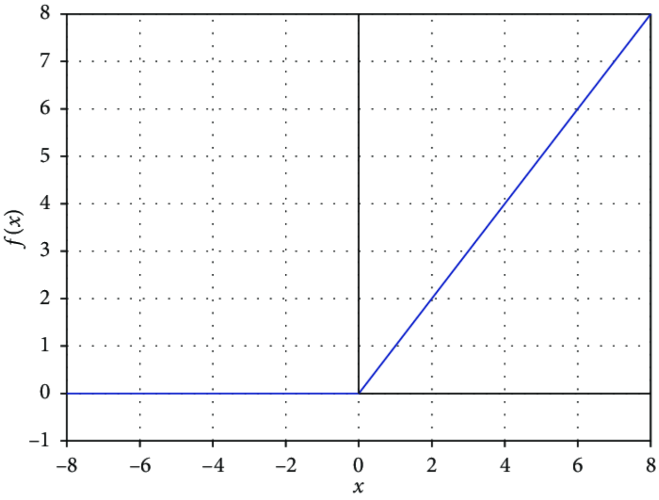


Figure 3.37 ReLu Activation Function [58]

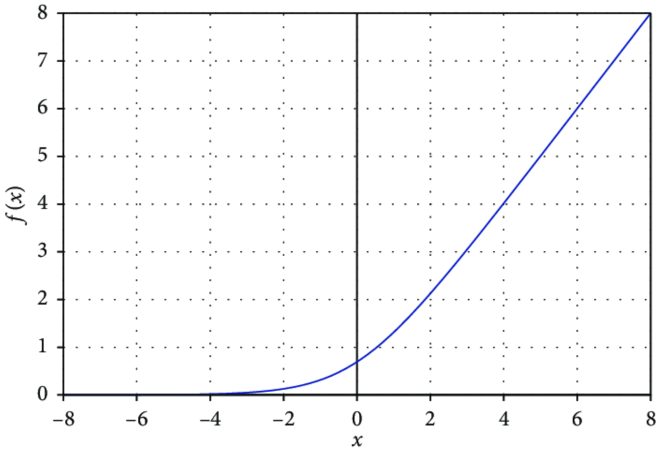


Figure 3.38 Softplus Activation Function [59]

3.5.4 Autoregressive Integrated Moving Average (ARIMA)

Time series is a sequence of data that is recorded in a timely manner over regular intervals. Time series data have different features that might not all be present, such as trend, seasonality, and cyclicality. A trend is an increase or decrease in values over time. Seasonality and cyclicality infer that the data have a pattern that either repeated with or without fixed intervals, respectively. Forecasting a time series falls into two categories: univariate time series forecasting and multivariate time series forecasting. The former uses only the previous values of the time series, while the latter uses additional predictors (i.e., exogenous data) to forecast.

ARIMA is a univariate time series model. It is also written as $ARIMA(p, d, q)$, where p is the order of the AR term, d is the differencing order, and q is the order of the MA term. The AR term describes the number of lags of the observations to use as predictors. The I term describes whether or not the data are differenced. The MA term describes the number of lags of the errors to use as predictors. [60]

The general formula of an ARIMA model can be written as:

$$\emptyset(B)(1 - B)^d X_t = \theta(B) Z_t, \quad \{Z_t\} \sim WN(0, \sigma^2)$$

Equation 3.7 ARIMA Equation [61]

Where $\emptyset(Z)$ and $\theta(Z)$ are polynomials of degrees p and q, respectively. [61]

Data require differencing if it is non-stationary, which indicates that the data is a function of time. Various tests are conducted to test stationarity, including the Augmented Dickey-Fuller test (ADH Test) and Kwiatkowski-Phillips-Schmidt-Shin (KPSS test) (trend stationary). ADF test checks if the data has a unit root, thus indicating a non-stationary data. While KPSS test checks if the data is stationary around a deterministic trend. [62]

Bike-sharing demand is a stochastic phenomenon; therefore, it was necessary to perform both tests. The results have shown that the demand is difference stationary, thus differencing the data yielded stationary stochastic data—further results in Chapter 4.

Moreover, the autocorrelation (ACF) and partial autocorrelation (PACF) plots are used to find the order for the MA term and AR term, respectively. The ACF plot describes the correlation between the error of the lagged observations. The PACF plot describes the correlation between the time series and its lags. In this study, the AutoArima from the Pmdarima library was used at the end of the time series analysis to find the order of the ARIMA model with the lowest AIC, i.e. Akaike Information Criterion. AIC is a performance metric usually used for time series models; the lower the AIC, the better the model and its predictability power.

Traditionally, the ARIMA model is used for non-seasonal time series. In case the data exhibit defined seasonality, Seasonal ARIMA (SARIMA) is used instead. None the less, seasonal ARIMA does not perform well with long seasonality and cannot deal with multiple seasonality as well. Therefore, a modified version of the ARIMA is used, where the seasonality is transformed into Fourier terms. This modified version is referred to as ARIMAX, where the X refers to exogenous variables. The Fourier term, in this case, are the exogenous variables. [63], [64]

The data used in this study are hourly data. Hourly data exhibit three types of seasonality; yearly, weekly, and daily. The data used are only a year-long; therefore, only weekly and daily seasonality are considered.

3.5.5 Hyperparameter Optimisation

Hyperparameter tuning is a fundamental part when training a model, as hyperparameters determine how the model is learning. For the machine learning models used in this study, these hyperparameters need to be initialised first before training. However, finding the optimal set of hyperparameters is a complicated task. Consequently, several time-saving optimisation algorithms are used to achieve better performance. An optimisation algorithm searches the hyperparameter space to find the best combination of hyperparameters to minimise the loss function of the validation set or cross-validation on the training set.

The most common optimisation algorithms are [65], [66]:

- Grid Search: It is an exhaustive search algorithm, in which a model is built from all possible combinations of hyperparameters in the space of given to build a model to find the optimal set of hyperparameters.
- Random Search: a model is built from different randomly selected combinations of hyperparameters from the space of given hyperparameters to find the optimal set. It is more favourable with high dimensionality space.
- Bayesian Optimisation: similar to random, yet it searches the space rather intelligently. It constantly evaluates the selected hyperparameters based on the current model and updates them accordingly.

Figure 3.39 shows how different optimisation algorithms behave in a given space of hyperparameters.

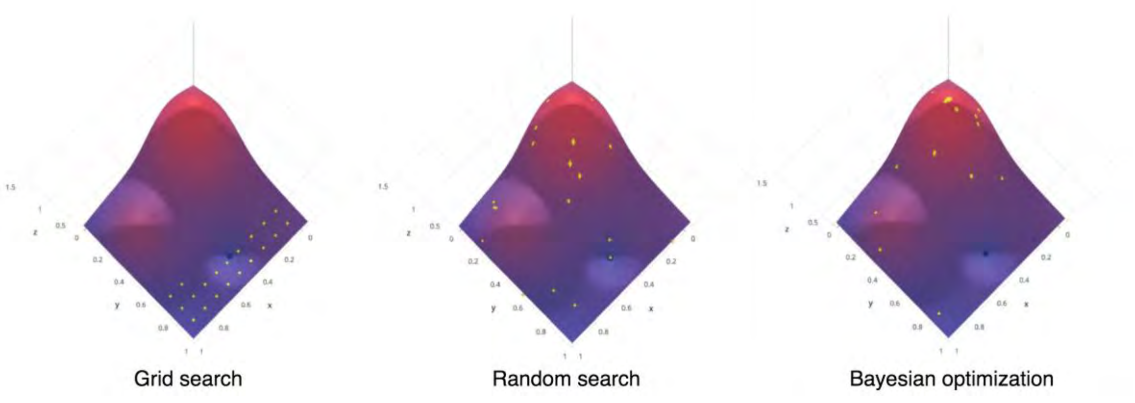


Figure 3.39 Different Hyperparameter Optimisation Algorithms [66]

The following subsections provide details about the different hyperparameters tuned in each model.

3.5.5.1 Random Forest Hyperparameters

The initial hyperparameters were the default set by the library (sklearn), except for maximum depth (10), out-of-bag score (True), and evaluation function (Mean Absolute Error). The optimal set of hyperparameters was obtained using Bayesian Optimisation with 50 iterations, and they are [67]:

- Number of estimators: 'the number of trees in the forest', with a range of (1, 2000)

- Maximum depth: ‘the maximum depth of the tree’, with a range of (1, 50)
- Minimum samples split: ‘the minimum number of samples required to split an internal node’, with a range of (1, 10)
- Minimum samples leaf: ‘The minimum number of samples required to be at a leaf node’, with a range of (1, 10)
- Maximum features: ‘the number of features (variables) to consider when looking for the best split’, with a range of (1, 122), where the 122 are the number of variables after transformation.

3.5.5.2 Gradient Boosting Hyperparameters

The initial hyperparameters were the default set by the library (sklearn), except for maximum depth (10), and evaluation function (Mean Absolute Error). Unlike RF, GBRT has fewer hyperparameters. The hyperparameter optimisation was done using Bayesian Optimisation with 50 iterations, and they are [68]:

- Number of estimators: ‘The number of boosting stages to perform ‘, with a range of (1, 2000)
- Learning rate: ‘learning rate shrinks the contribution of each tree by learning_rate. There is a trade-off between learning_rate and n_estimators‘, with a range of (-3, 1)
- Minimum samples split: ‘the minimum number of samples required to split an internal node’, with a range of (1, 10)
- Maximum depth: ‘maximum depth of the individual regression estimators ‘, with a range of (0, 25)
- Minimum samples leaf: ‘The minimum number of samples required to be at a leaf node’, with a range of (1, 10)
- Maximum features: ‘the number of features (variables) to consider when looking for the best split’, with a range of (1, 122), where the 122 are the number of variables after transformation.

3.5.5.3 Neural Network Hyperparameters

In MLP, constructing a simple network aid in preventing overfitting, and thus the process of hyperparameter tuning made easier. It is recommended using Random Search to obtain the optimal parameters, but the space of the given hyperparameter is considerably small; therefore, Grid Search has been used. The MLP trained for each station consists of three layers, an input layer, a hidden layer, and an output layer. The hyperparameters tuned were optimised separately in the same order given, either separately or in pairs as stated:

- Batch size and number of epochs: batch size is the number of samples drawn from the data, where the sample itself is referred to as a mini-batch. Neural networks use these mini-batches to update the weights. That being said, the batch size determines how many times the neural network update its weight per epoch. As a result, the neural network trains faster. The most common batch sizes used in the literature are 32, 64, 128, and 256, which were used in tuning this hyperparameter. On the other hand, an epoch is one cycle in which the whole data is trained. Usually, the number of epochs used are large to allow the network to learn. Nevertheless, when the obtained error of the validation data set gets higher, it means that the network is overfitting. Therefore, choosing the right number is crucial. The numbers selected for the epochs were 20, 30, and 50 since the MPL is simple. The two hyperparameters were optimised together, as they influence each other.

- Optimisation algorithms: also called optimiser, it the algorithm the neural network uses to update its weight and reduce the error. The most common optimisers are Stochastic Gradient Descent (SGD), Root Mean Square prop (RMSprop) and Adaptive Moment Estimation (Adam). These optimisers were included in hyperparameter tunning, along with Adam with Nesterov momentum (Nadam)
- Learning rate (LR) and momentum: Depending on the optimiser obtained from the previous step. The learning rate and momentum were tunned, as some optimisers include only the learning rate while other requires both parameters. The range for the learning rate was (0.001, 0.01, 0.1, 0.2, 0.3), while for the momentum was (0.0, 0.2,0.4,0.6,0.8,0.9)
- Network weight initialisation: it describes how the initial weight was distributed between the different variables. The distributions included were, uniform, normal, LeCun uniform, Glorot uniform, Glorot normal, He normal, and He uniform.
- Neuron activation function: as discussed previously, only the three functions above were used; ReLu, Softplus, and Linear.
- Output activation function: ReLu, Softplus, and Linear.
- The number of neurons: the number of neurons in the input layer, that yields a minimum error.

The initial MLP was trained with 122 neurons in the input layer (equals the number of variables), the weights were initialised using a uniform distribution and ReLu as an activation function. The Linear activation function was used in the output layer.

3.5.5.4 ARIMAX

ARIMAX do not require any hyperparameters tuning. However, a stepwise search similar to Grid Search was used to find the best order of Fourier terms for daily and weekly seasonality using AutoArima to include in the final model.

3.5.6 Implementation

The hourly aggregated data set of each station was split into training and testing sets. For reasons of comparison between the different models, the data dating from Nov. 01, 2018, till Oct. 31, 2019, was used as the training set and the first week of November (Nov. 01, 2019, till Nov. 07, 2019) was used as the testing set.

Further inspection of each station data, 8 Ave & W 33 Station had zero demand for more than 24 hours consecutively, which may indicate closure for maintenance or inspection. Therefore, the following station with the highest hourly demand in both check-ins and check-out was used instead, as seen in Figure 3.11 and Figure 3.10.

The implementation of machine learning models followed the architecture seen in Figure 3.40.

3.5.6.1 Data Transformation

The dependent variables used for machine learning models were:

1. Month: 1 – 12

2. Day: name of the weekday
3. Hour: 0 – 24
4. Weekend: 0 or 1, 1 indicating a weekend
5. Season: name of the season
6. Holiday: 0 or 1, 1 indicating a holiday
7. Working day: 0 or 1, 1 indicating a working day
8. Weather type: a detailed description of the weather condition in a given hour, contains 52 unique values, either a single or multiple types combined, such as partially cloudy, light rain, mist, light snow, heavy snow, lightning without thunder, thunderstorm, fog, etc.
9. Conditions: a concise description of the weather condition in a given hour, contained only six unique values. The conditions were, clear, partially cloudy, 'rain, clear', 'rain, overcast', overcast, and 'rain, partially cloudy.'
10. Temperature
11. Apparent temperature
12. Relative humidity
13. Precipitation
14. Precipitation cover
15. Dew point
16. Visibility
17. Wind speed

The first nine variables are categorical, and the remaining are numerical. RF and GBRT do not necessarily require the data to be transformed. Nonetheless, using the sklearn library, this option is not provided; consequently, they cannot handle categorical data. On the other hand, NN requires the data to be normalised and transformed.

As a result, the categorical variables were transformed using a one-hot encoder, and the numerical variables were normalised using standardisation. This process was repeated for three models.

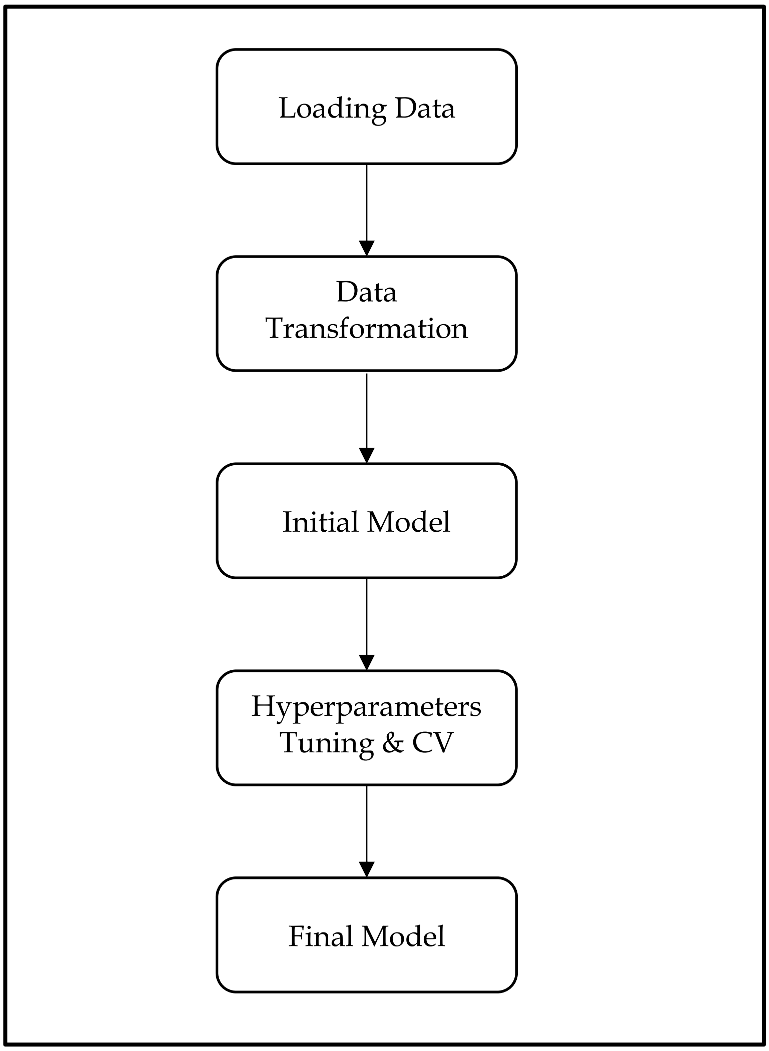


Figure 3.40 Implementation Architecture

3.5.7 Cross-Validation (CV)

Cross-validation is usually necessary for hyperparameter tuning. The data is split into k-fold, where one fold is held back, and the model is trained using the seen data, the process is repeated k times. The data used is a time-series data; hence, the time-series split provided by the sklearn library was used. It splits the data at a fixed interval without shuffling. In this study, 3-fold cross-validation was used.

3.5.8 Evaluation Metrics

The model performance is evaluated using three categories: computational time, Mean Absolute Error (MAE), and Root Mean Squared Error (RMSE).

3.5.8.1 Computational Time

The time it took each algorithm to complete a model using the training set is the main criterion to be considered. However, the time consumed for hyperparameters optimisation is also considered.

3.5.8.2 Mean Absolute Error

Mean absolute error (MAE) ‘measures the average magnitude of the errors in a set of predictions, without considering their direction’ [69].

$$MAE = \frac{1}{n}\sum_{j=1}^{n} |y_j - \hat{y}_j|$$

3.5.8.3 Root Mean Squared Logarithmic Error

Root mean squared logarithmic error (RMSLE) is a quadratic scoring that measures the average magnitude of the log of the error.

$$RMSLE = \sqrt{\frac{1}{n}\sum_{j=1}^{n} (\log(y_j) - \log(\hat{y}_j))^2}$$

Chapter 4

Results & Conclusion

4.1 Results

This section presents the results of the four models performed. GBRT and ARIMAX model tend to produce negative values; it is a common statistical practise to convert those negative values to zero in such case. Accordingly, the error terms were recalculated after obtaining the prediction dataset, which gave slightly different errors due to eliminating all the negative values.

The following machine learning subsections are organised as follow.

- The hyperparameter optimisation results
- The error terms for the final model (training set)
- The computational costs for training the initial models and final models, and time consumed to obtain the hyperparameters
- Visualisations of the actual values and the predicted values.

The ARIMAX model subsection is organised as follow.

- ACF and PACF plot
- Fourier terms result for daily and weekly seasonality
- ARIMAX model and residual plots
- ARIMAX error terms
- ARIMAX computational time
- Visualisations of the actual values and the predicted values

4.1.1 Random Forest

4.1.1.1 Hyperparameter Optimisation Results

Table 4.1 RF Hyperparameters of Check-out Stations

Station Name	Max. Depth	Max. Feature	Min. Samples Leaf	Min. Samples Split	No. of Estimators
Allen St & Stanton St	22	29	3	7	1689
W 41 St & 8 Ave	34	108	1	2	1595
1 Ave & E 16 St.	45	45	1	7	460

Table 4.2 RF Hyperparameters of Check-in Stations

Station Name	Max. Depth	Max. Feature	Min. Samples Leaf	Min. Samples Split	No. of Estimators
Allen St & Stanton St	32	113	1	8	50
1 Ave & E 16 St.	36	121	0	2	1686
Broadway & E 14 St	39	23	0	6	621

4.1.1.2 Final Model Error Terms

Table 4.3 RF Model Errors of Check-out Stations

Station Name	MAE	RMSLE
Allen St & Stanton St	3.0147	0.5749
W 41 St & 8 Ave	4.8206	0.5788
1 Ave & E 16 St.	3.8363	0.4346

Table 4.4 RF Model Errors of Check-in Stations

Station Name	MAE	RMSLE
Allen St & Stanton St	2.7024	0.5193
1 Ave & E 16 St.	3.712	0.4533
Broadway & E 14 St	3.5791	0.4064

4.1.1.3 Computational Time

Table 4.5 RF Computational Time of Check-out Stations

Station Name	Initial Model	Hyperparameter Opt.	Final Model
Allen St & Stanton St	1 min 45 sec	10 min 11 sec	39 sec
W 41 St & 8 Ave	2 min 18 sec	13 min 16 sec	2 min 23 sec
1 Ave & E 16 St.	1 min 59 sec	11 min 40 sec	17.6 sec

Table 4.6 RF Computational Time of Check-in Stations

Station Name	Initial Model	Hyperparameter Opt.	Final Model
Allen St & Stanton St	2 min 19 sec	11 min 21 sec	3.94 sec
1 Ave & E 16 St.	2 min 10 sec	18 min 11 sec	2 min 48 sec
Broadway & E 14 St	2 min 20 sec	14 min 8 sec	14 sec

4.1.1.4 Visualisation

The following are the plots of the check-out stations:

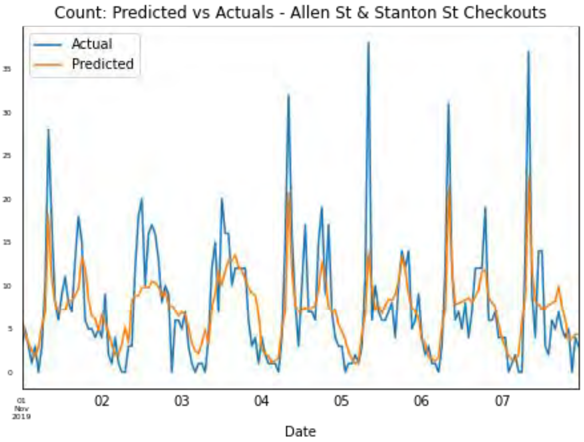


Figure 4.1 RF Predicted vs Actuals - Allen St & Stanton St

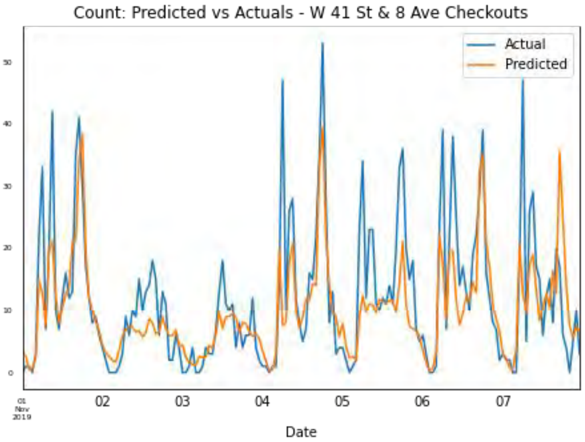


Figure 4.2 RF Predicted vs Actuals - W 41 St & 8 Ave

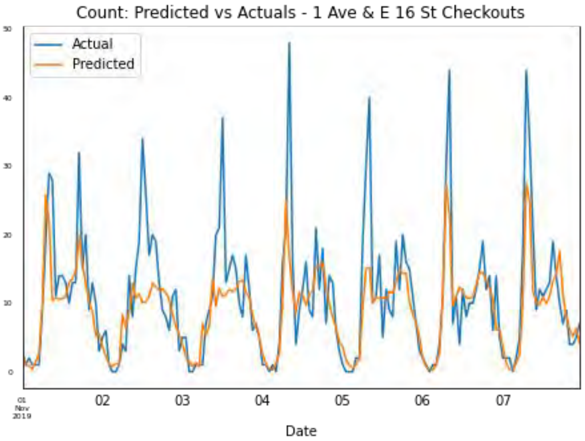


Figure 4.3 RF Predicted vs Actuals - 1 Ave & E 16 St.

The following are the plots of the check-ins stations:

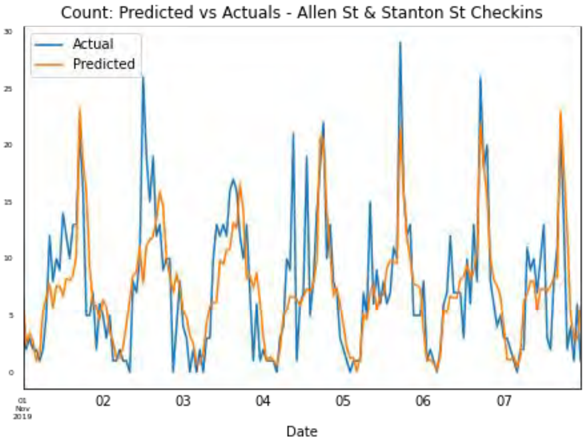


Figure 4.4 RF Predicted vs Actuals - Allen St & Stanton St

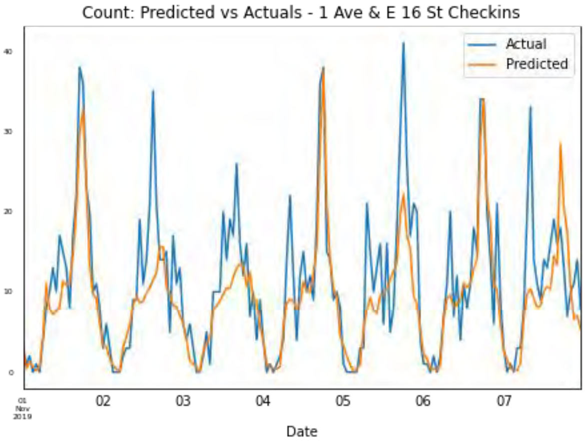


Figure 4.5 RF Predicted vs Actuals - 1 Ave & E 16 St.

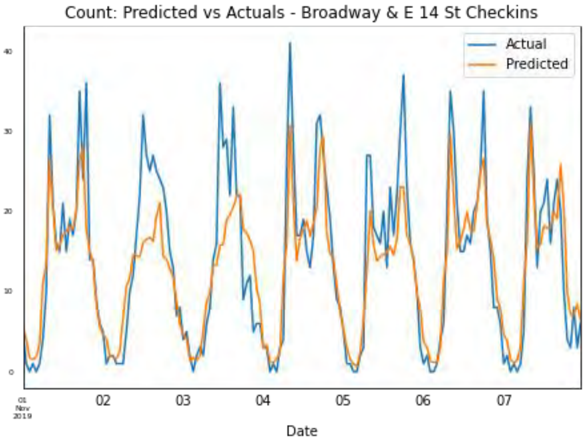


Figure 4.6 RF Predicted vs Actuals - Broadway & E 14 St

4.1.2 Gradient Boosting Regression Tree

4.1.2.1 Hyperparameter Optimisation Results

Table 4.7 GBRT Hyperparameters of Check-out Stations

Station Name	Learning Rate	Max. Depth	Max. Feature	Min. Samples Leaf	Min. Samples Split	No. of Estimators
Allen St & Stanton St	0.7933	4	23	4	9	1257
W 41 St & 8 Ave	0.0813	8	102	2	5	68
1 Ave & E 16 St.	0.0922	16	22	2	4	1428

Table 4.8 GBRT Hyperparameters of Check-in Stations

Station Name	Learning Rate	Max. Depth	Max. Feature	Min. Samples Leaf	Min. Samples Split	No. of Estimators
Allen St & Stanton St	0.0533	6	70	1	7	1499
1 Ave & E 16 St.	0.0522	10	120	2	6	144
Broadway & E 14 St	0.1002	4	15	6	7	524

4.1.2.2 Final Model Error Terms

Table 4.9 GBRT Errors of Check-out Stations

Station Name	MAE	RMSLE
Allen St & Stanton St	2.8095	0.5177
W 41 St & 8 Ave	4.5753	0.5488
1 Ave & E 16 St.	3.6239	0.4192

Table 4.10 GBRT Errors of Check-in Stations

Station Name	MAE	RMSLE
Allen St & Stanton St	2.5578	0.4917
1 Ave & E 16 St.	3.8882	0.4857
Broadway & E 14 St	3.0958	0.3436

4.1.2.3 Computational Time

Table 4.11 GBRT Computational Time of Check-out Stations

Station Name	Initial Model	Hyperparameter Opt.	Final Model
Allen St & Stanton St	41 min 38 sec	16 min 55 sec	
W 41 St & 8 Ave	44 min 3 sec	9 min 53 sec	4.48 sec
1 Ave & E 16 St.	40 min 43 sec	16 min 26 sec	36.6 sec

Table 4.12 GBRT Computational Time of Check-in Stations

Station Name	Initial Model	Hyperparameter Opt.	Final Model
Allen St & Stanton St	41 min	30 min 58 sec	42.9 sec
1 Ave & E 16 St.	40 min 35 sec	11 min 8 sec	10.6 sec
Broadway & E 14 St	42 min 30 sec	14 min 11 sec	2.56 sec

4.1.2.4 Visualisation

The following are the plots of the check-out stations:

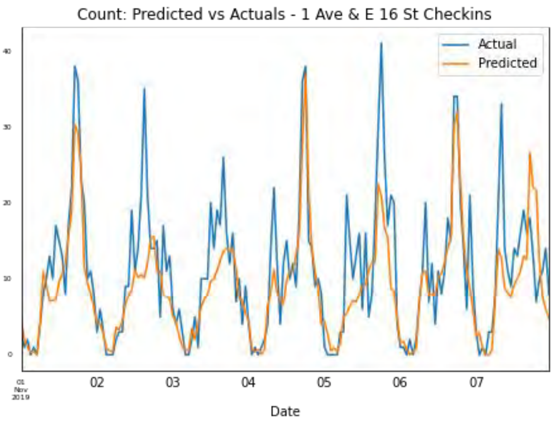


Figure 4.7 GBRT Predicted vs Actuals - Allen St & Stanton St

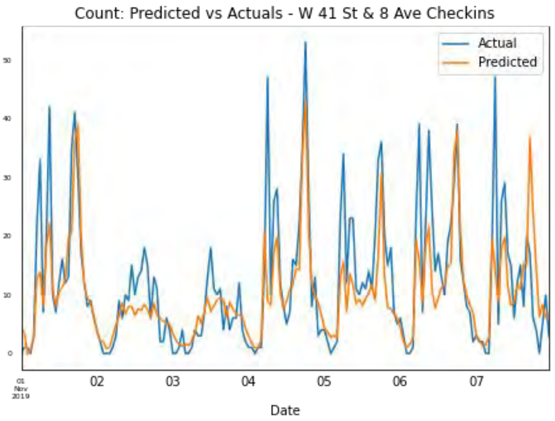


Figure 4.8 GBRT Predicted vs Actuals - W 41 St & 8 Ave

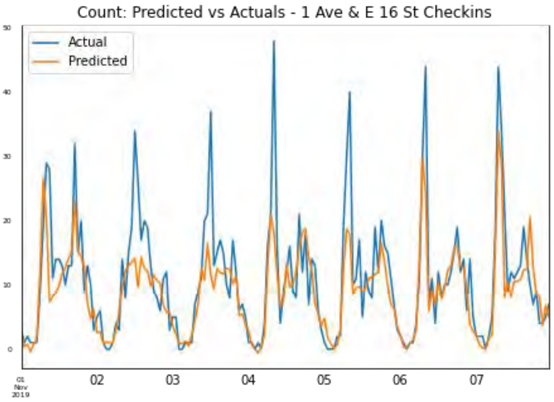


Figure 4.9 GBRT Predicted vs Actuals - 1 Ave & E 16 St.

The following are the plots of the check-in stations:

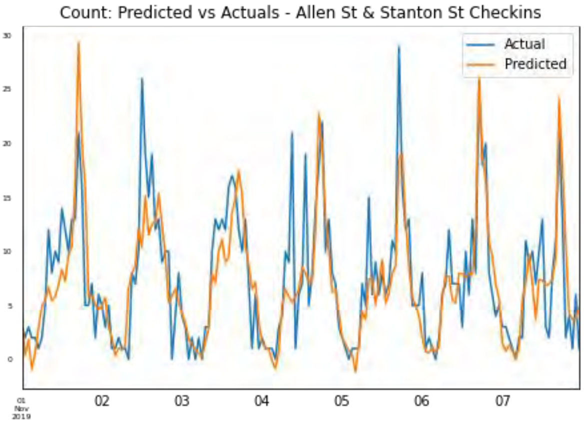


Figure 4.10 GBRT Predicted vs Actuals – Allen St & Stanton

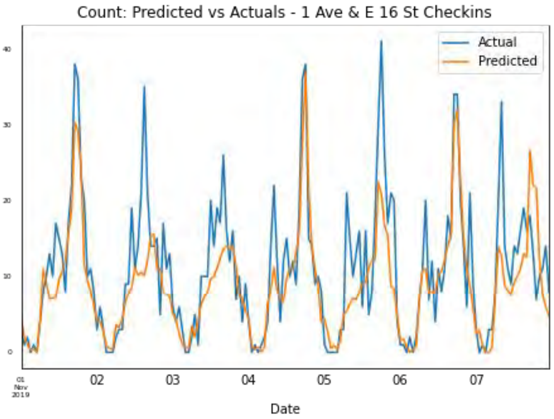


Figure 4.11 GBRT Predicted vs Actuals – 1 Ave & E 16 St

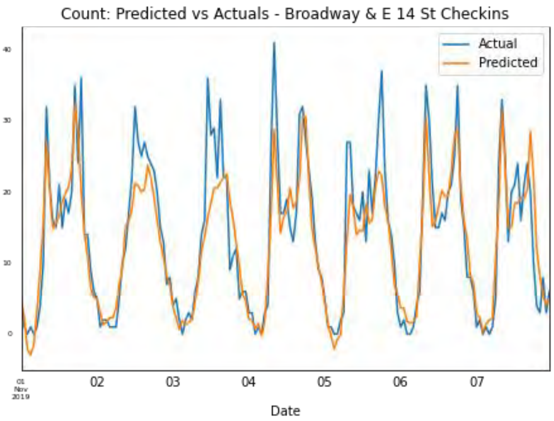


Figure 4.12 GBRT Predicted vs Actuals – Broadway & E 14 St

4.1.3 Neural Network (MLP)

4.1.3.1 Hyperparameter Optimisation Results

1. Batch size and epochs
2. Optimisation algorithm
3. Learning rate and momentum (LR. and/or mom.)
4. Network weight initialisation
5. Input activation function
6. Output activation function
7. Number of neurons

Each number in the table corresponds to the mentioned hyperparameters, respectively.

Table 4.13 MLP Hyperparameters of Check-out stations

Station Name	1	2	3	4	5	6	7
Allen St & Stanton St	128, 50	RMSprop	lr. 0.001 mom. 0.2	Normal	ReLu	Linear	91
W 41 St & 8 Ave	64, 30	Adam	Lr. 0.001	LeCun uniform	ReLu	ReLu	122
1 Ave & E 16 St.	32, 30	RMSprop	lr. 0.001 mom. 0.4	Normal	ReLu	Softplus	61

Table 4.14 MLP Hyperparameters of Check-in stations

Station Name	1	2	3	4	5	6	7
Allen St & Stanton St	32, 20	Nadam	Lr. 0.001	Glorot uniform	ReLlu	Linear	91
1 Ave & E 16 St.	32, 20	Nadam	Lr. 0.001	Glorot uniform	ReLlu	ReLlu	122
Broadway & E 14 St	32, 20	Adam	Lr. 0.001	Glorot Normal	ReLlu	Linear	61

4.1.3.2 Final Model Error Terms

Table 4.15 MLP Error of Check-out Stations

Station Name	MAE	RMSLE
Allen St & Stanton St	2.6991	0.5237
W 41 St & 8 Ave	4.0774	0.4956
1 Ave & E 16 St.	3.4179	0.4021

Table 4.16 MLP Error of Check-in Stations

Station Name	MAE	RMSLE
Allen St & Stanton St	2.2846	0.4612
1 Ave & E 16 St.	3.663	0.4719
Broadway & E 14 St	2.8382	0.3225

4.1.3.3 Computational Time

Table 4.17 MLP Computational Time of Check-out Stations

Station Name	Initial Model	Hyperparameter Opt.	Final Model
Allen St & Stanton St	7.65 sec	3 min 16.8 sec	5.77 sec
W 41 St & 8 Ave	7.95 sec	2 min 30.1 sec	8.78 sec
1 Ave & E 16 St.	7.62 sec	4 min 29.6 sec	10.3 sec

Table 4.18 MLP Computational Time of Check-in Stations

Station Name	Initial Model	Hyperparameter Opt.	Final Model
Allen St & Stanton St	7 sec	2 min 54.8 sec	11.3 sec
1 Ave & E 16 St.	7.63 sec	2 min 44.8 sec	7.39 sec
Broadway & E 14 St	8.61 sec	2 min 50.9 sec	11.6 sec

4.1.3.4 Visualisation

The following are the plots of the check-out stations:

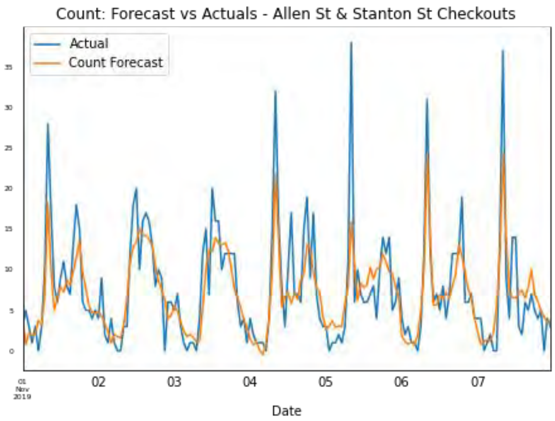


Figure 4.13 MLP Predicted vs Actuals - Allen St & Stanton St

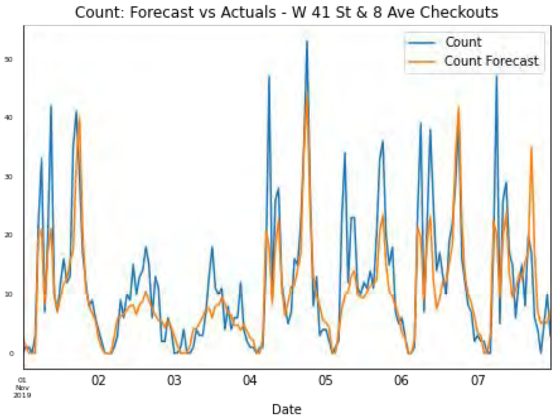


Figure 4.14 MLP Predicted vs Actuals - W 41 St & 8 Ave

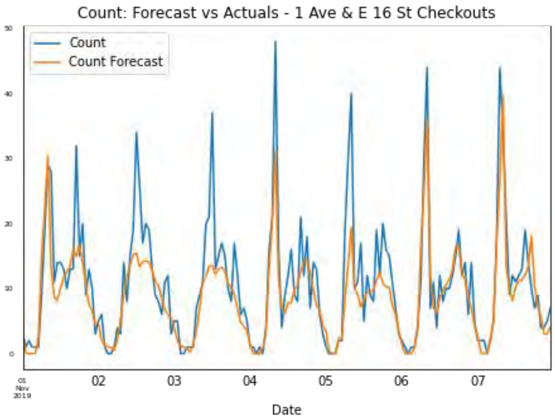


Figure 4.15 MLP Predicted vs Actuals - 1 Ave & E 16 St.

The following are the plots of the check-in stations:

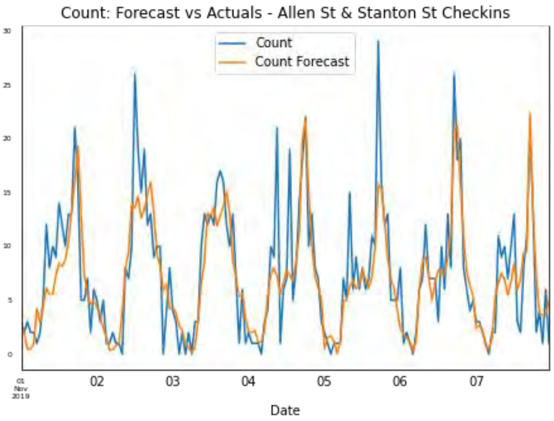


Figure 4.16 MLP Predicted vs Actuals – Allen St & Stanton St

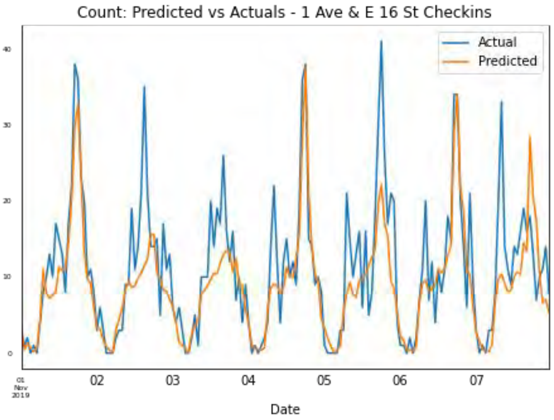


Figure 4.17 MLP Predicted vs Actuals – 1 Ave & E 16 St.

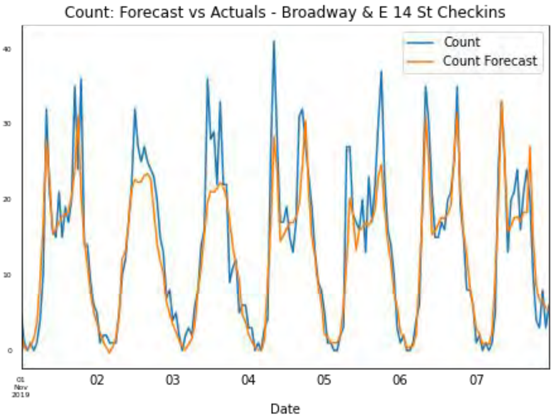


Figure 4.18 MLP Predicted vs Actuals – Broadway & E 14 St

4.1.4 ARIMAX

4.1.4.1 ADF & KPSS Results

All the examined stations were stationary using the ADF test and non-stationary using the KPSS test indicating the series is difference stationary. The time-series data were differenced once to obtain stationarity.

4.1.4.2 Fourier Terms Results

Table 4.19 Fourier terms order of Check-out Stations

Station Name	Order of daily seasonality	Order of weekly seasonality
Allen St & Stanton St	12	2
W 41 St & 8 Ave	12	-
1 Ave & E 16 St.	12	2

Table 4.20 Fourier Terms Order of Check-in Stations

Station Name	Order of daily seasonality	Order of weekly seasonality
Allen St & Stanton St	12	2
1 Ave & E 16 St.	12	3
Broadway & E 14 St	11	3

4.1.4.3 ARIMAX Model & Residual Plots

The obtain the order of the ARIMA model, AutoArima was used with a maximum value of 3 for both q and p.

Table 4.21 ARIMAX Order of Check-out Stations

Station Name	Model order
Allen St & Stanton St	(0, 1, 2)
W 41 St & 8 Ave	(0, 1, 3)
1 Ave & E 16 St.	(0, 1, 2)

Table 4.22 ARIMAX Order of Check-in Stations

Station Name	Model order
Allen St & Stanton St	(0, 1, 3)
1 Ave & E 16 St.	(0, 1, 3)
Broadway & E 14 St	(1, 1, 3)

4.1.4.4 ARIMAX Error Terms

The error terms were recalculated again after obtaining the predictions.

Table 4.23 ARIMAX Errors of Check-out Stations

Station Name	MAE	RMSLE
Allen St & Stanton St	3.2898	0.5979
W 41 St & 8 Ave	5.4595	0.7009
1 Ave & E 16 St.	4.385	0.5278

Table 4.24 ARIMAX Errors of Check-in Stations

Station Name	MAE	RMSLE
Allen St & Stanton St	2.8716	0.5479
1 Ave & E 16 St.	3.9321	0.5436
Broadway & E 14 St	3.8045	0.4275

4.1.4.5 ARIMAX Computational Time

Table 4.25 ARIMAX Computational Time of Check-out Stations

Station Name	Fourier Terms	ARIMAX	Predictions
Allen St & Stanton St	2 h 8 min 59 sec	32.5 sec	1 h 27 min 47 min
W 41 St & 8 Ave	1 h 54 min 7 sec	24.3 sec	1 h 11 min 38 sec
1 Ave & E 16 St.	1 h 58 min 57 sec	28.4 sec	1 h 28 min 56 sec

Table 4.26 ARIMAX Computational Time of Check-in Stations

Station Name	Fourier Terms	ARIMAX	Predictions
Allen St & Stanton St	1 h 56 min 27 sec	28.5 sec	1 h 21 min 25 sec
1 Ave & E 16 St.	4 h 56 min 5 sec	57.2 sec	2 h 24 min 5 sec
Broadway & E 14 St	2 h 48 min 23 sec	1 min 1 sec	32 min 35 sec

4.1.5 Visualisation

The following are the plots of the check-out stations:

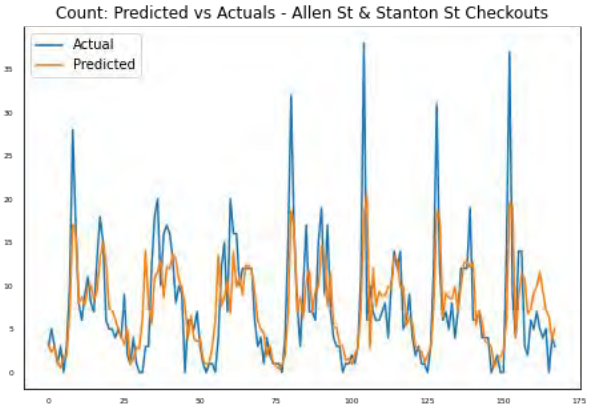


Figure 4.19 ARIMAX Predicted vs Actuals - Allen St & Stanton St

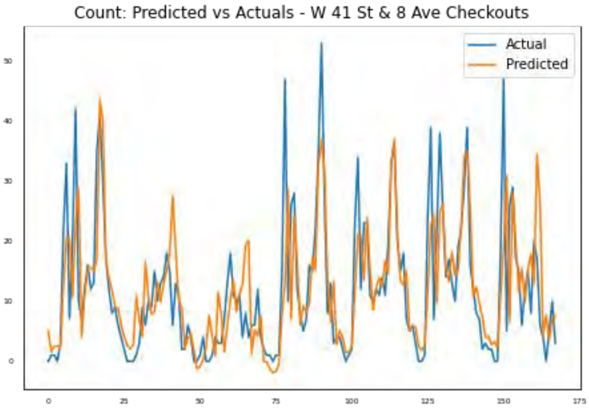


Figure 4.20 ARIMAX Predicted vs Actuals - W 41 St & 8 Ave

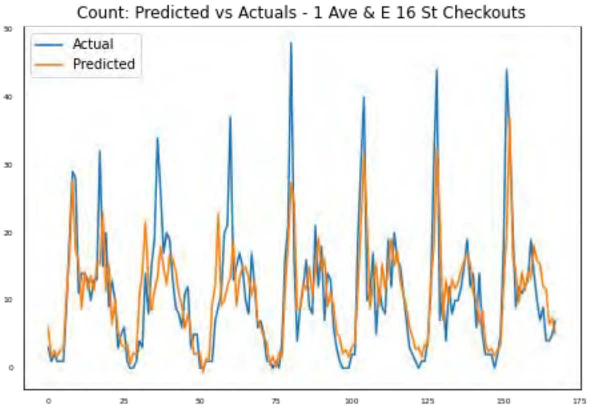


Figure 4.21 ARIMAX Predicted vs Actuals - 1 Ave & E 16 St.

The following are the plots of the check-ins stations:

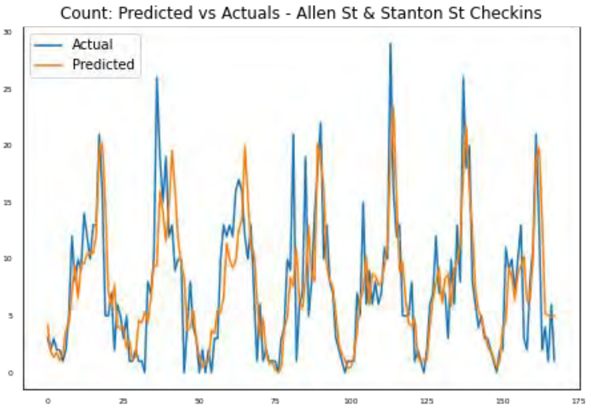


Figure 4.22 ARIMAX Predicted vs Actuals - Allen St & Stanton St

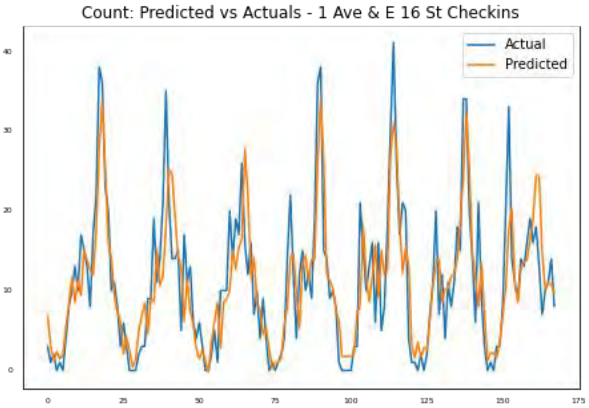


Figure 4.23 ARIMAX Predicted vs Actuals - 1 Ave & E 16 St.

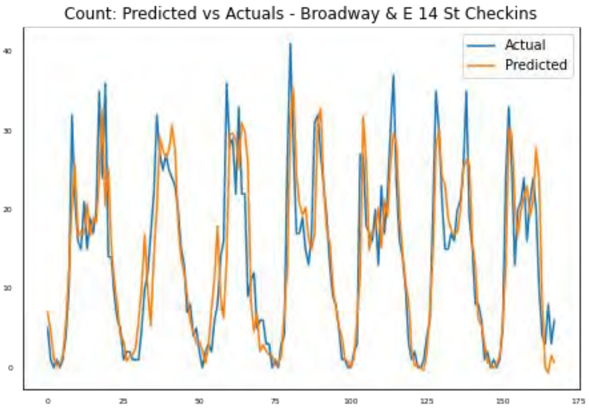


Figure 4.24 ARIMAX Predicted vs Actuals - Broadway & E 14 St

4.2 Findings & Discussion

An initial objective of this study was to analyse the bike sharing system. This study confirms that the demand is affected by weather yet highly associated with time and seasons. These results seem to be consistent with other research which found that neural networks are of the highest performance when compared to ensemble models. It could be argued that the positive results were due to the ability of neural networks to find the relation between highly correlated variables. None the less, both random forest and gradient boosting regression tree are non-linear model, yet extracting the feature importance would positively contribute to better performance.

Stochastic phenomena with complex and multiple seasonality are rather hard to model with traditional time series model, but the results obtained are encouraging. Hourly data exhibit yearly seasonality as well, but the data used was only a year long. It is more likely that the performance would improve with a more extended period. Besides, the weather information provided in the machine learning algorithms could suggest why these algorithms performed better.

4.2.1 Comparison in Terms of Hyperparameters/parameters

RF and GBRT have almost the same number of hyperparameters which makes it less complicated to tune. The hyperparameters were optimised using Bayesian Optimisation, which allowed for all the hyperparameters to be in the same search space. Nevertheless, the search builds a model from a randomly selected combination and try to optimise it. In some cases, the initial hand-selected model may perform better than the optimised one. Moreover, the need to repeat the optimisation or increase the number of iteration might be necessary to obtain the optimal combination. In one of the model, increasing the depth manually, enhanced the performance by two decimal points. The process had to be repeated up to three times in some cases.

On the other hand, neural networks have more complicated hyperparameters and require much expertise. The hyperparameters were being tuned in a step-wise manner using grid search. Therefore, it was challenging to acquire the optimal set; there is always the possibility of a different outcome when repeating the process. In addition, going a step back was needed at times.

ARIMAX has the least amount of parameters. AutoArima was set to a step-wise search similar to a grid search. The value of p and q provided were small to avoid a complex model. In addition, AutoArima indicates 'inf' next to orders that has a unit root close to 1 in order for it not be selected.

4.2.2 Comparison in Terms of Computational Time

RF models were relatively faster than its counterpart; GBRT, due to its parallel computation property, which can be seen when comparing the initial models. The accelerated performance was attributed by the ability of the RF model to build the trees simultaneously, where GBRT builds each tree based on performance the previous one.

ARIMAX models were the most computationally expensive algorithms. The multi-step prediction took twice as much to find the order of the two Fourier terms. Finding the order of the Fourier terms and the model itself, consumed more than training the other models collectively. None the less, machine learning algorithms have the advantage to predict all the values at once, unlike the multi-step model. The durations sometimes fluctuated due to the virtual machine, whether it is using its full CPU capacity or not; this problem was not faced with the other algorithms.

Thus far, regardless of the complexity of the MLP models, they were the fastest to train and tune.

4.2.3 Comparison in Terms of Errors

ARIMAX was the worst algorithm in all the stations. Nonetheless, the plots of predicted and actual values were closest to the pattern of usage in each station but shifted. Comparatively, MLP models had the best performance and the lowest error term in all model. Surprisingly, GBRT models had lower errors than RF models, especially after adjusting the negative values to zero.

4.3 Conclusion & Further Research

This study is set to examine the bike sharing demand characteristics and compare the station-based machine learning algorithms to the traditional time series model ARIMA. This study has found that generally, neural networks have better performance. The findings also suggest that the traditional time-series model with the right modelling could predict the demand accurately. A key strength of the present study was the modelling of an ARIMA model with multiple seasonality for a bikesharing demand forecasting which lays the groundwork for future research into investigating the prediction of micro-mobility with multiple seasonality using traditional time series modelling.

This would be a fruitful area for further work in investigating time series model that deals with multiple seasonality in the transport sector, as much of research can be seen in microeconomic and the stock market.

A further study could assess how to expand these models to predict the flow of all stations simultaneously using the same micro-period or less, using the likes of bigQuery and other automation tools and techniques.

Bibliography

[1] S. A. Shaheen, S. Guzman, and H. Zhang, 'Bikesharing in Europe, the Americas, and Asia: Past, Present, and Future', *Transportation Research Record: Journal of the Transportation Research Board*, vol. 2143, no. 1, pp. 159–167, Jan. 2010, doi: 10.3141/2143-20.

[2] R. Buehler and A. Hamre, 'Business and Bikeshare User Perceptions of the Economic Benefits of Capital Bikeshare', *Transportation Research Record: Journal of the Transportation Research Board*, vol. 2520, no. 1, pp. 100–111, Jan. 2015, doi: 10.3141/2520-12.

[3] 'Citi Bike Station Map', *Citi Bike NYC*. https://member.citibikenyc.com/map/ (accessed Mar. 30, 2020).

[4] M. Cheng, 'Sharing economy: A review and agenda for future research', *International Journal of Hospitality Management*, vol. 57, pp. 60–70, Aug. 2016, doi: 10.1016/j.ijhm.2016.06.003.

[5] P. Muñoz and B. Cohen, 'Mapping out the sharing economy: A configurational approach to sharing business modeling', *Technological Forecasting and Social Change*, vol. 125, pp. 21–37, Dec. 2017, doi: 10.1016/j.techfore.2017.03.035.

[6] M. Hossain, 'Sharing economy: A comprehensive literature review', *International Journal of Hospitality Management*, vol. 87, p. 102470, May 2020, doi: 10.1016/j.ijhm.2020.102470.

[7] T. Puschmann and R. Alt, 'Sharing Economy', *Business & Information Systems Engineering*, vol. 58, no. 1, pp. 93–99, Feb. 2016, doi: 10.1007/s12599-015-0420-2.

[8] S. Shaheen, N. Chan, A. Bansal, and A. Cohen, 'Shared Mobility: A Sustainability & Technologies Workshop: Definitions, Industry Developments, and Early Understanding', Nov. 2015, Accessed: May 07, 2020. [Online]. Available: https://trid.trb.org/view/1375066.

[9] S. Shaheen, E. W. Martin, A. F. Cohen, and R. S. Finson, 'Public Bikesharing in North America: Early Operator and User Understanding', 2012. /paper/Public-Bikesharing-in-North-America%3A-Early-Operator-Shaheen-Martin/62f8a00c7590065d949aab5256165f0f935bec4b.

[10] S. Shaheen, S. Guzman, and H. Zhang, 'In City Cycling', in *Bikesharing across the Globe*, vol. 9, MIT Press. UC Berkeley: Transportation Sustainability Research Center, 2012.

[11] S. Shaheen, 'Micromobility Policy Toolkit: Docked and Dockless Bike and Scooter Sharing', 2019, doi: 10.7922/G2TH8JW7.

[12] T. Bieliński and A. Ważna, 'Hybridizing bike-sharing systems: the way to improve mobility in smart cities', *Transport Economics and Logistics*, vol. 79, pp. 53–63, Oct. 2018, doi: 10.26881/etil.2018.79.04.

[13] A. Pal and Y. Zhang, 'Free-floating bike sharing: Solving real-life large-scale static rebalancing problems', *Transportation Research Part C: Emerging Technologies*, vol. 80, pp. 92–116, Jul. 2017, doi: 10.1016/j.trc.2017.03.016.

[14] E. Fishman, 'Bikeshare: A Review of Recent Literature', *Transport Reviews*, vol. 36, no. 1, pp. 92–113, 2016, doi: 10.1080/01441647.2015.1033036.

[15] A. Kaltenbrunner, R. Meza, J. Grivolla, J. Codina, and R. Banchs, 'Urban cycles and mobility patterns: Exploring and predicting trends in a bicycle-based public

transport system', *Pervasive and Mobile Computing*, vol. 6, no. 4, pp. 455–466, Aug. 2010, doi: 10.1016/j.pmcj.2010.07.002.

[16] J. W. Yoon, F. Pinelli, and F. Calabrese, 'Cityride: A Predictive Bike Sharing Journey Advisor', in *2012 IEEE 13th International Conference on Mobile Data Management*, Bengaluru, India, Jul. 2012, pp. 306–311, doi: 10.1109/MDM.2012.16.

[17] W. El-Assi, M. Salah Mahmoud, and K. Nurul Habib, 'Effects of built environment and weather on bike sharing demand: a station level analysis of commercial bike sharing in Toronto', *Transportation*, vol. 44, no. 3, pp. 589–613, May 2017, doi: 10.1007/s11116-015-9669-z.

[18] C. Gallop, C. Tse, and J. Zhao, 'A Seasonal Autoregressive Model Of Vancouver Bicycle Traffic Using Weather Variables', *i-manager's Journal on Civil Engineering*, vol. 1, no. 4, pp. 9–18, Nov. 2011, doi: 10.26634/jce.1.4.1694.

[19] K. Gebhart, 'The Impact of Weather Conditions on Capital Bikeshare Trips', p. 25, 2013.

[20] H. I. Ashqar, M. Elhenawy, M. H. Almannaa, A. Ghanem, H. A. Rakha, and L. House, 'Modeling bike availability in a bike-sharing system using machine learning', in *2017 5th IEEE International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS)*, Naples, Italy, Jun. 2017, pp. 374–378, doi: 10.1109/MTITS.2017.8005700.

[21] P.-C. Chen, H.-Y. Hsieh, X. K. Sigalingging, Y.-R. Chen, and J.-S. Leu, 'Prediction of Station Level Demand in a Bike Sharing System Using Recurrent Neural Networks', *2017 IEEE 85th Vehicular Technology Conference (VTC Spring)*, 2017, doi: 10.1109/VTCSpring.2017.8108575.

[22] D. Washimkar, 'Forecast Use of a City Bikeshare System', Dec. 15, 2014. http://darshanwashimkar.github.io/documents/Forecast-Use-Of-A-City-Bikeshare-System.pdf (accessed Apr. 27, 2020).

[23] Y. Li, Y. Zheng, H. Zhang, and L. Chen, 'Traffic prediction in a bike-sharing system', in *Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems - GIS '15*, Bellevue, Washington, 2015, pp. 1–10, doi: 10.1145/2820783.2820837.

[24] S. Feng, H. Chen, C. Du, J. Li, and N. Jing, 'A Hierarchical Demand Prediction Method with Station Clustering for Bike Sharing System', in *2018 IEEE Third International Conference on Data Science in Cyberspace (DSC)*, Guangzhou, Jun. 2018, pp. 829–836, doi: 10.1109/DSC.2018.00133.

[25] J. Liu, L. Sun, Q. Li, J. Ming, Y. Liu, and H. Xiong, 'Functional zone based hierarchical demand prediction for bike system expansion', 2017, vol. Part F129685, pp. 957–966, doi: 10.1145/3097983.3098180.

[26] H. Yang, X. Zhang, L. Zhong, S. Li, X. Zhang, and J. Hu, 'Short-term demand forecasting for bike sharing system based on machine learning', 2019, pp. 1295–1300, doi: 10.1109/ICTIS.2019.8883732.

[27] P. Hulot, D. Aloise, and S. D. Jena, 'Towards Station-Level Demand Prediction for Effective Rebalancing in Bike-Sharing Systems', in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, London United Kingdom, Jul. 2018, pp. 378–386, doi: 10.1145/3219819.3219873.

[28] Z. Yang, J. Hu, Y. Shu, P. Cheng, J. Chen, and T. Moscibroda, 'Mobility Modeling and Prediction in Bike-Sharing Systems', in *Proceedings of the 14th Annual International Conference on Mobile Systems, Applications, and Services - MobiSys '16*, Singapore, Singapore, 2016, pp. 165–178, doi: 10.1145/2906388.2906408.

[29] S. Ruffieux, N. Spycher, E. Mugellini, and O. A. Khaled, 'Real-Time usage forecasting for bike-sharing systems: A study on random forest and convolutional neural network applicability', 2018, vol. 2018-January, pp. 622–631, doi: 10.1109/IntelliSys.2017.8324359.

[30] B. Wang and I. Kim, 'Short-term prediction for bike-sharing service using machine learning', 2018, vol. 34, pp. 171–178, doi: 10.1016/j.trpro.2018.11.029.

[31] 'Citi Bike Membership & Pass Options', *Citi Bike NYC*. http://www.citibikenyc.com/pricing (accessed Mar. 30, 2020).

[32] 'Citi Bike System Data', *Citi Bike NYC*. http://www.citibikenyc.com/system-data (accessed Jan. 29, 2020).

[33] 'Index of bucket "tripdata"'. https://s3.amazonaws.com/tripdata/index.html (accessed Jan. 29, 2020).

[34] 'Weather Data & Mapping | Visual Crossing'. https://www.visualcrossing.com/ (accessed Jun. 03, 2020).

[35] 'Weather Data Documentation – Visual Crossing Weather'. https://www.visualcrossing.com/resources/documentation/weather-data/weather-data-documentation/ (accessed Jun. 03, 2020).

[36] N. US Department of Commerce, 'Heat Index'. https://www.weather.gov/safety/heat-index (accessed May 04, 2020).

[37] N. US Department of Commerce, 'Wind Chill Chart'. https://www.weather.gov/safety/cold-wind-chill-chart (accessed Jul. 04, 2020).

[38] N. US Department of Commerce, 'Dew Point vs Humidity'. https://www.weather.gov/arx/why_dewpoint_vs_humidity (accessed May 05, 2020).

[39] 'Wind Chill'. https://www.weather.gov/media/epz/wxcalc/windChill.pdf (accessed May 04, 2020).

[40] 'Heat Index Equation'. https://www.wpc.ncep.noaa.gov/html/heatindex_equation.shtml (accessed May 04, 2020).

[41] R. J. A. Little and D. B. Rubin, *Statistical analysis with missing data*, Third Edition. Hoboken, NJ: Wiley, 2020.

[42] M. J. Azur, E. A. Stuart, C. Frangakis, and P. J. Leaf, 'Multiple imputation by chained equations: what is it and how does it work?', *International Journal of Methods in Psychiatric Research*, vol. 20, no. 1, pp. 40–49, 2011, doi: 10.1002/mpr.329.

[43] 'Google Maps', *Google Maps*. https://www.google.com/maps/place/New+York,+NY,+USA/@40.7633854,-73.9793661,12.83z/data=!4m5!3m4!1s0x89c24fa5d33f083b:0xc80b8f06e177fe62!8m2!3d40.7127753!4d-74.0059728 (accessed May 07, 2020).

[44] 'type from FOLDOC'. https://foldoc.org/data+type (accessed Aug. 08, 2020).

[45] 'NY citibike 2018', *Google Docs*. https://docs.google.com/spreadsheets/d/1Aum0anWxPx6bHyfcFXWCCTE8u0xtfenIls_kPAJEDIA/edit?usp=embed_facebook (accessed Jun. 06, 2020).

[46] 'Stations JSON file'. https://layer.bicyclesharing.net/map/v1/nyc/stations (accessed Jun. 06, 2020).

[47] 'Stations JSON file'. https://feeds.citibikenyc.com/stations/stations.json (accessed Jun. 06, 2020).

[48] O. Calgary, 'Borough Boundaries', *NYC Open Data*. https://data.cityofnewyork.us/City-Government/Borough-Boundaries/tqmj-j8zm (accessed Apr. 24, 2020).

[49] 'When is ...? Dates of holidays, religious festivals & events'. https://www.calendarpedia.com/ (accessed May 09, 2020).

[50] E. Eren and V. E. Uz, 'A review on bike-sharing: The factors affecting bike-sharing demand', *Sustainable Cities and Society*, vol. 54, p. 101882, Mar. 2020, doi: 10.1016/j.scs.2019.101882.

[51] L. Breiman, 'Random Forest', *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001, doi: 10.1023/A:1010933404324.

[52] L. Breiman, *Arcing The Edge*. 1997.

[53] J. H. Friedman, 'Greedy Function Approximation: A Gradient Boosting Machine', *The Annals of Statistics*, vol. 29, no. 5, pp. 1189–1232, 2001, Accessed: Jul. 13, 2020. [Online]. Available: https://www.jstor.org/stable/2699986.

[54] J. H. Friedman, 'Stochastic gradient boosting', *Computational Statistics & Data Analysis*, vol. 38, no. 4, pp. 367–378, Feb. 2002, doi: 10.1016/S0167-9473(01)00065-2.

[55] 'Understanding Activation Functions in Deep Learning'. https://www.learnopencv.com/understanding-activation-functions-in-deep-learning/ (accessed Aug. 14, 2020).

[56] Y.-S. Park and S. Lek, 'Artificial Neural Networks', in *Developments in Environmental Modelling*, vol. 28, Elsevier, 2016, pp. 123–140.

[57] 'Figure 11: Linear activation function. It is one of the simplest...' *ResearchGate*. https://www.researchgate.net/figure/Linear-activation-function-It-is-one-of-the-simplest-activation-functions-and-it-is_fig4_339172446 (accessed Aug. 14, 2020).

[58] 'Figure 12: ReLU activation function. is activation function is also...', *ResearchGate*. https://www.researchgate.net/figure/ReLU-activation-function-is-activation-function-is-also-fast-to-compute-but-it-allows-a_fig3_339172446 (accessed Aug. 14, 2020).

[59] 'Figure 14: Softplus activation function. is one is also slower to...', *ResearchGate*. https://www.researchgate.net/figure/Softplus-activation-function-is-one-is-also-slower-to-compute-than-ReLU-and-linear_fig6_339172446 (accessed Aug. 14, 2020).

[60] H. G. Daellenbach and R. L. Flood, *The Informed student guide to management science*. London: Thomson, 2002.

[61] 'Statistical Learning and Data Analytics for TS', Technical University of Munich, Jul. 23, 2019, Accessed: Aug. 04, 2019. [Online].

[62] 'Stationarity and detrending (ADF/KPSS) — statsmodels'. https://www.statsmodels.org/dev/examples/notebooks/generated/stationarity_detrending_adf_kpss.html (accessed Aug. 04, 2019).

[63] R. J. Hyndman, 'Forecasting with daily data | Rob J Hyndman', Sep. 17, 2013. https://robjhyndman.com/hyndsight/dailydata/ (accessed Aug. 01, 2020).

[64] R. J. Hyndman, 'Forecasting with long seasonal periods | Rob J Hyndman', Sep. 29, 2010. https://robjhyndman.com/hyndsight/longseasonality/ (accessed Aug. 01, 2020).

[65] 'Hyperparameter tuning for machine learning models.', *Jeremy Jordan*, Nov. 02, 2017. https://www.jeremyjordan.me/hyperparameter-tuning/ (accessed Jul. 15, 2020).

[66] 'Common Problems in Hyperparameter Optimization', *SigOpt*, Mar. 29, 2017. https://sigopt.com/blog/common-problems-in-hyperparameter-optimization/ (accessed Jul. 14, 2020).

[67] '3.2.4.3.2. sklearn.ensemble.RandomForestRegressor — scikit-learn 0.23.2 documentation'. https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html (accessed Jul. 15, 2020).

[68] '3.2.4.3.6. sklearn.ensemble.GradientBoostingRegressor — scikit-learn 0.23.2 documentation'. https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingRegressor.html?highlight=gradient%20boostingregressor#sklearn.ensemble.GradientBoostingRegressor (accessed Jul. 15, 2020).

[69] 'MAE and RMSE — Which Metric is Better? | by JJ | Human in a Machine World | Medium'. https://medium.com/human-in-a-machine-world/mae-and-rmse-which-metric-is-better-e60ac3bde13d (accessed Jul. 10, 2020).

Appendix

A. Further Visualisation

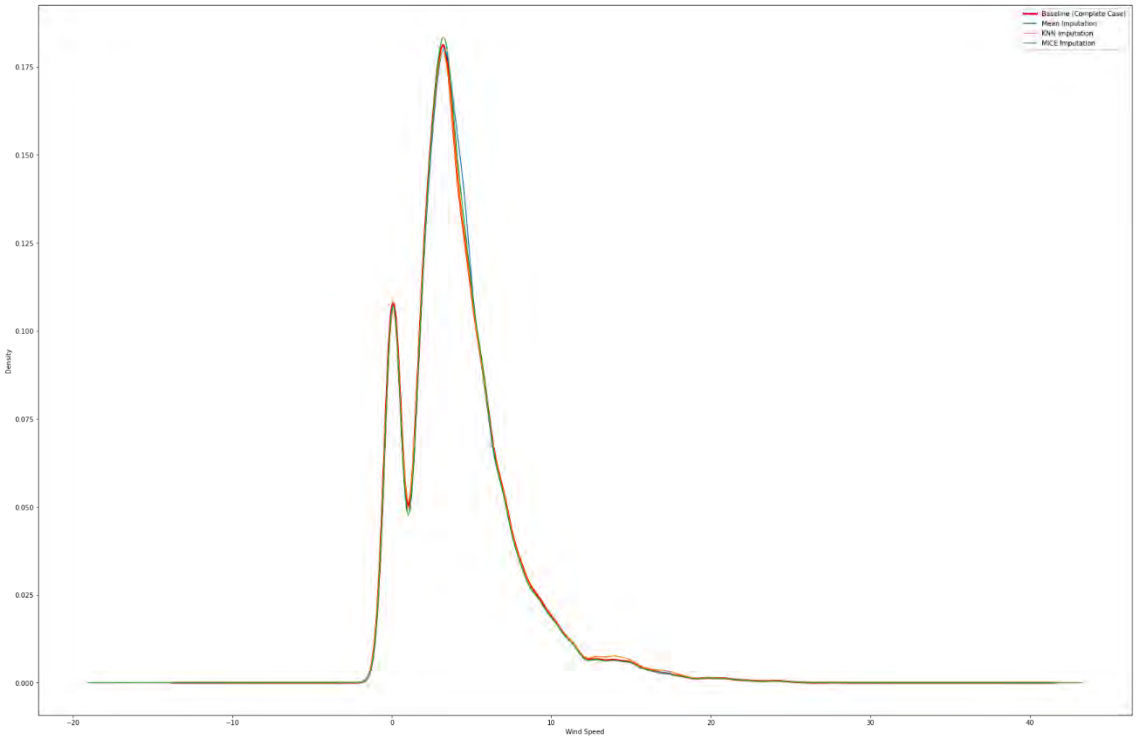


Figure A.1 Wind Speed Density Plot

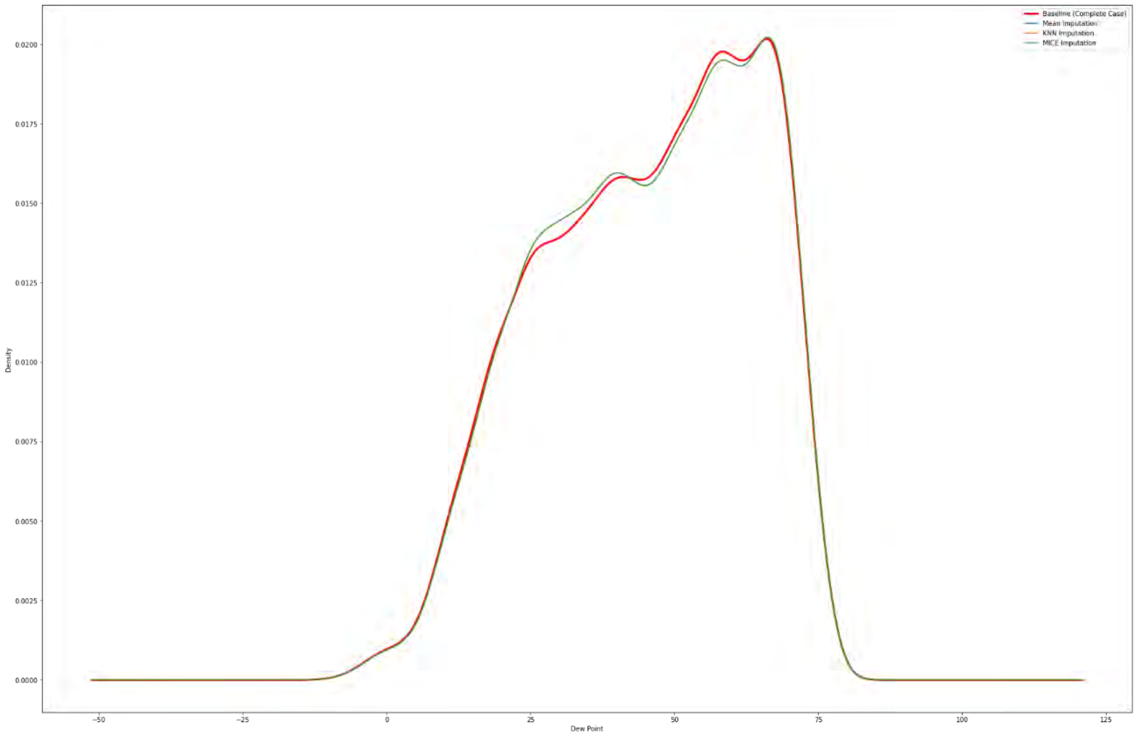


Figure A.2 Dew Point Density Plot

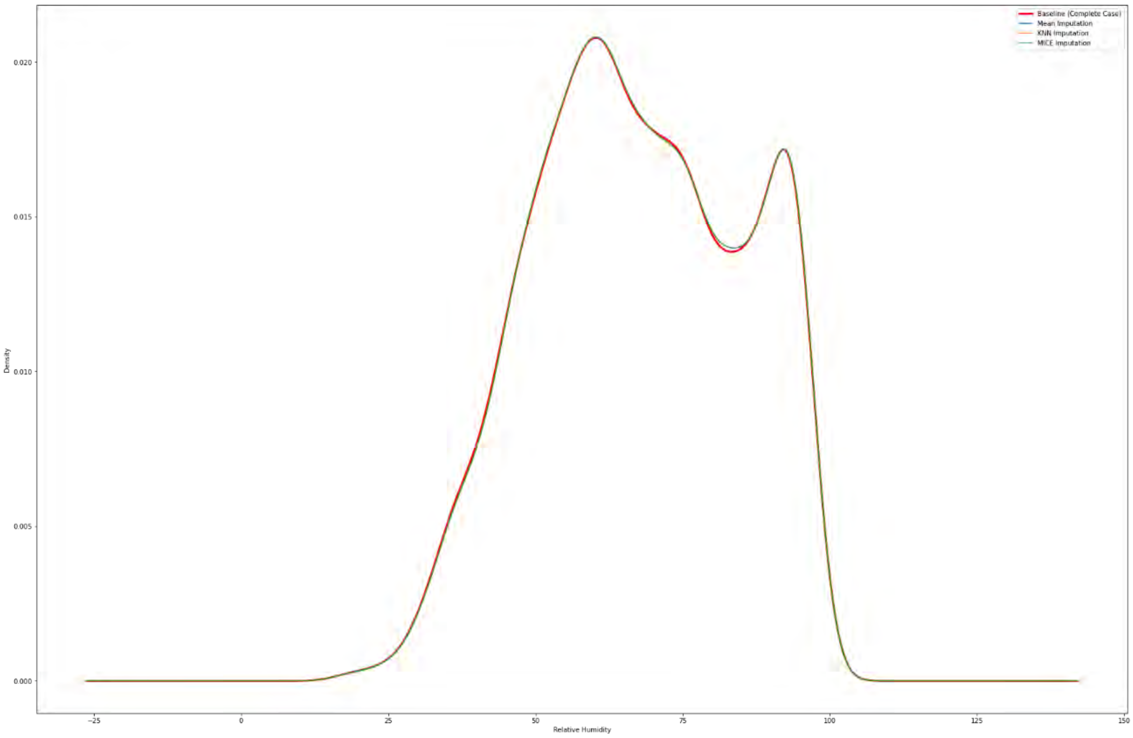


Figure A.3 Relative Humidity Density Plot

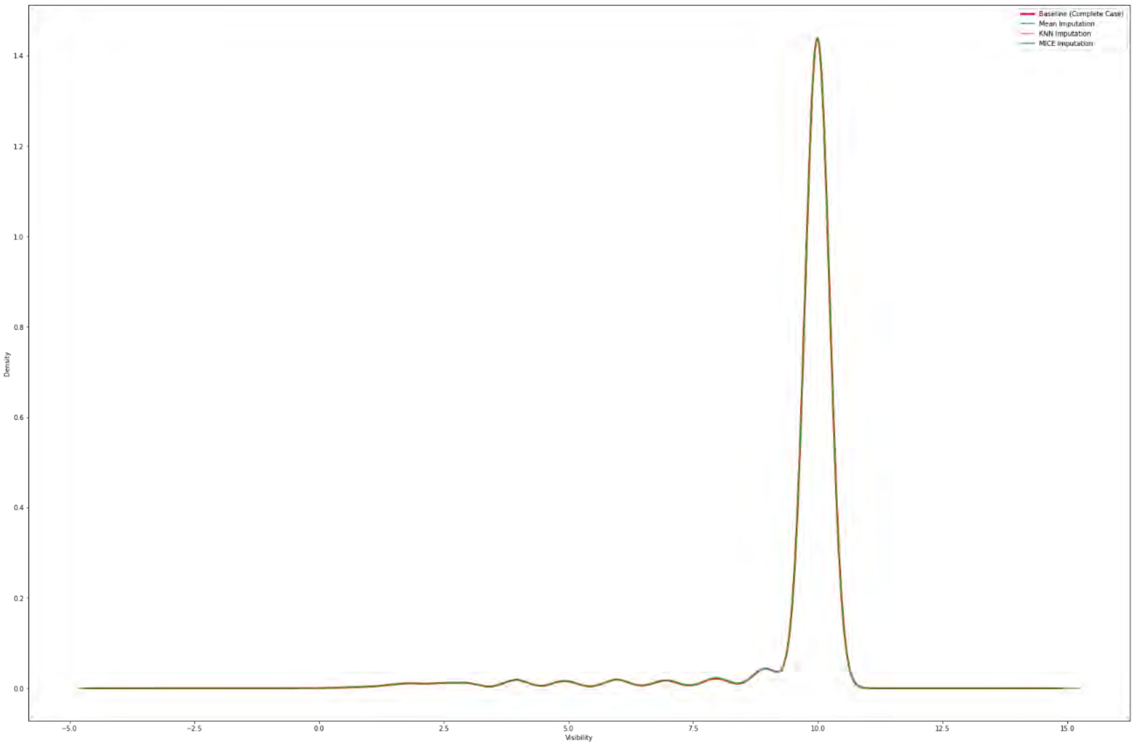


Figure A.4 Visibility Density Plot

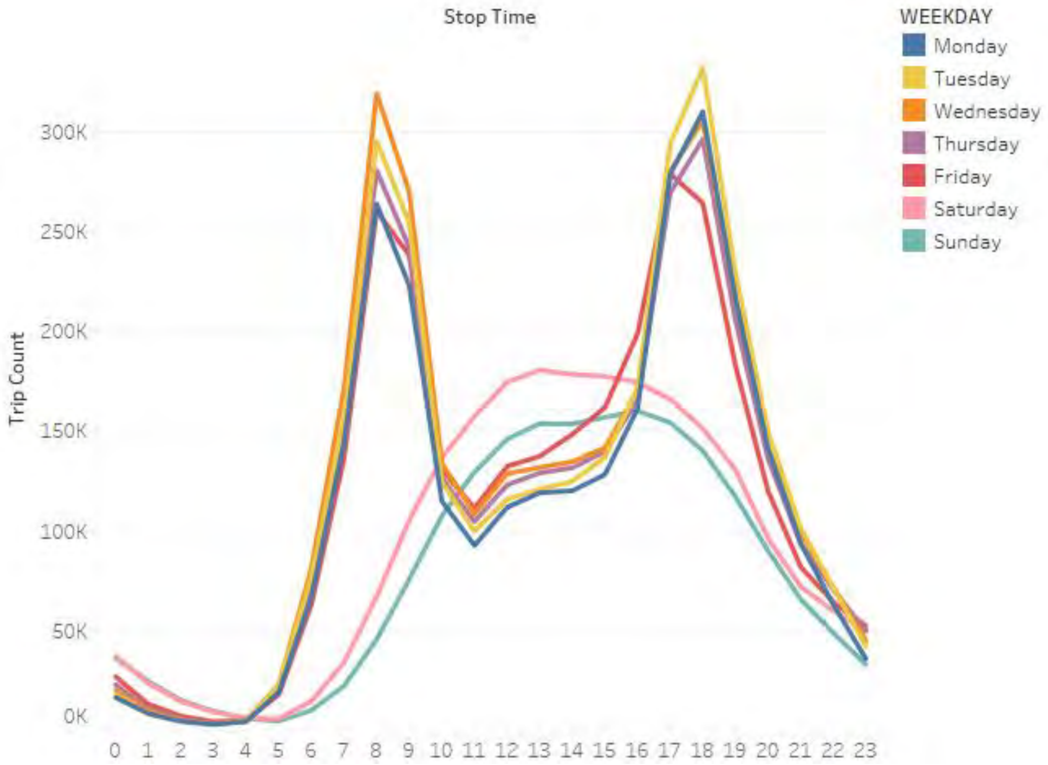


Figure A.5 Day of the Week (Hourly) vs Trip Count – Subscribers

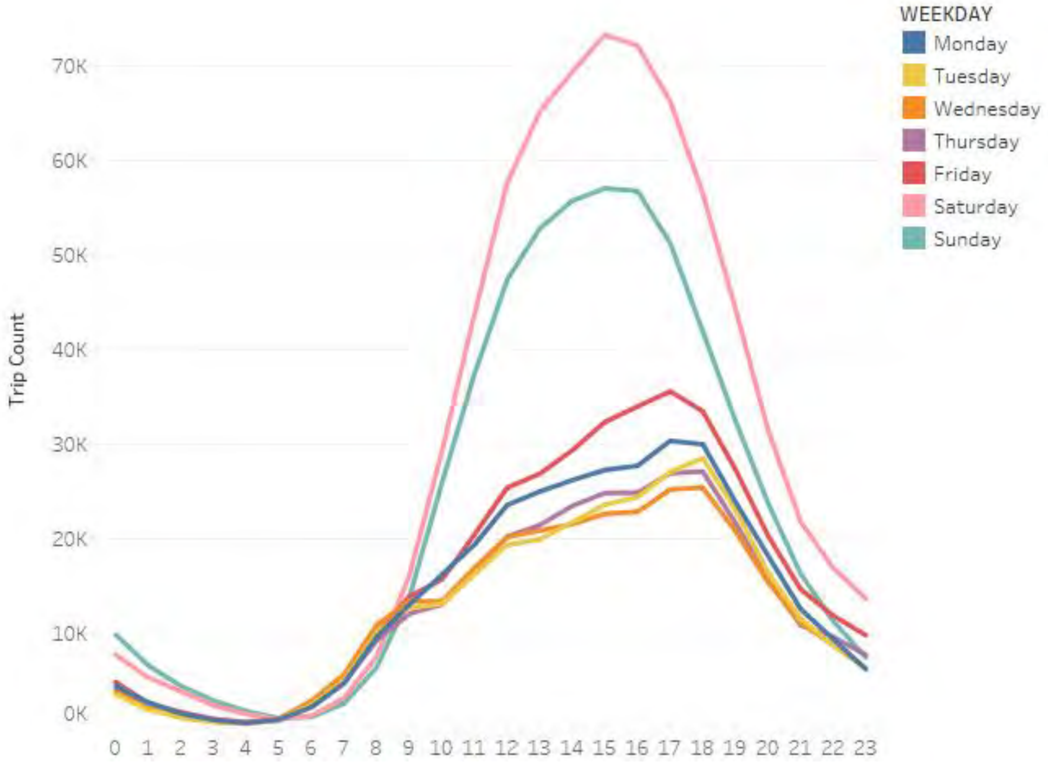


Figure A.6 Day of the Week (Hourly) vs Trip Count – Customers

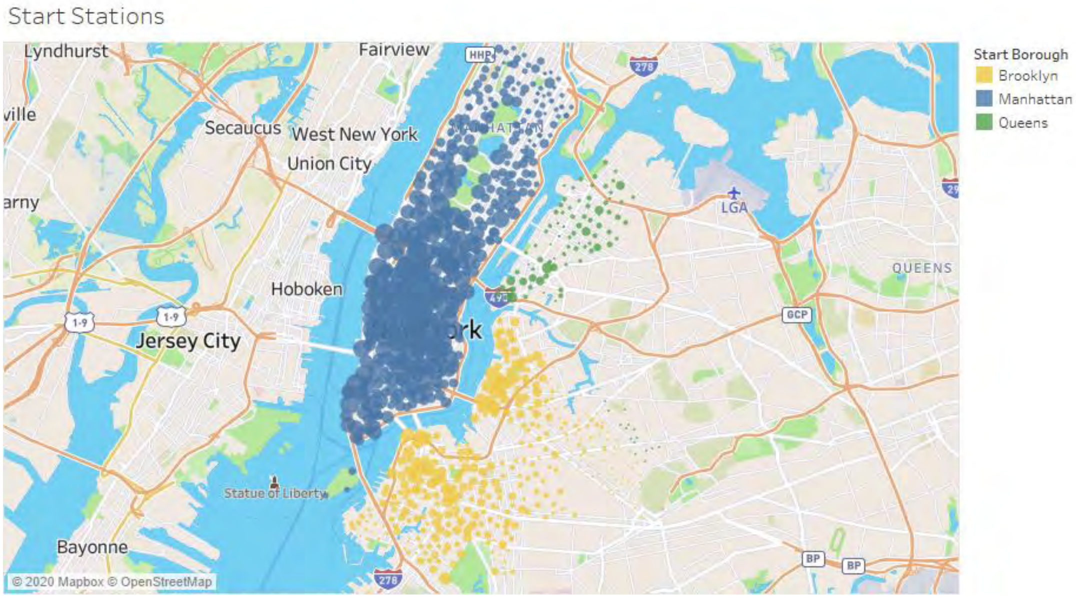


Figure A.7 Start Stations by Demand Density

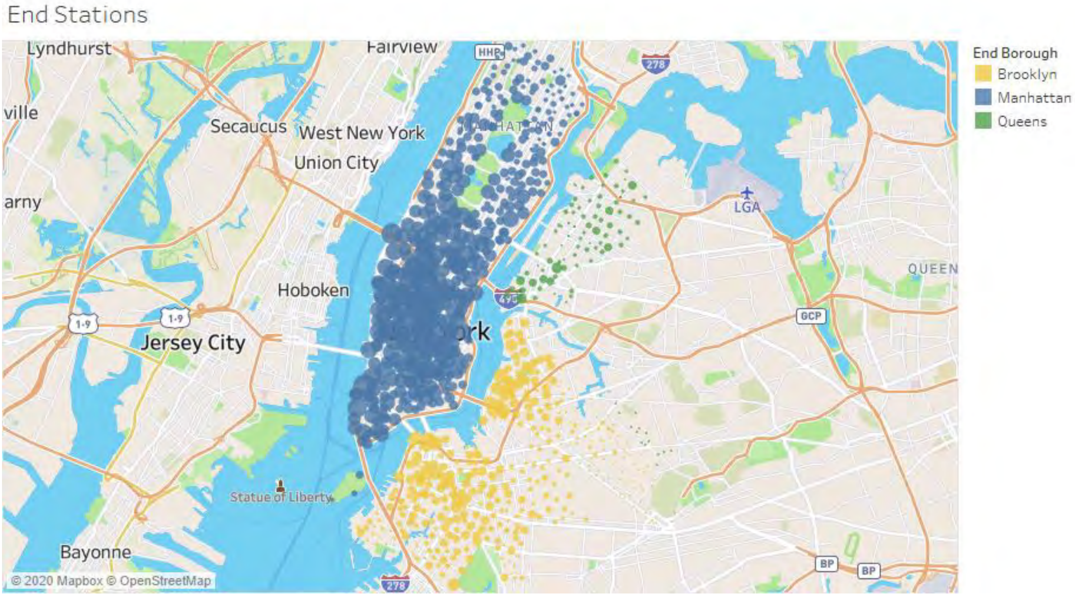


Figure A.8 End Stations by Demand Density

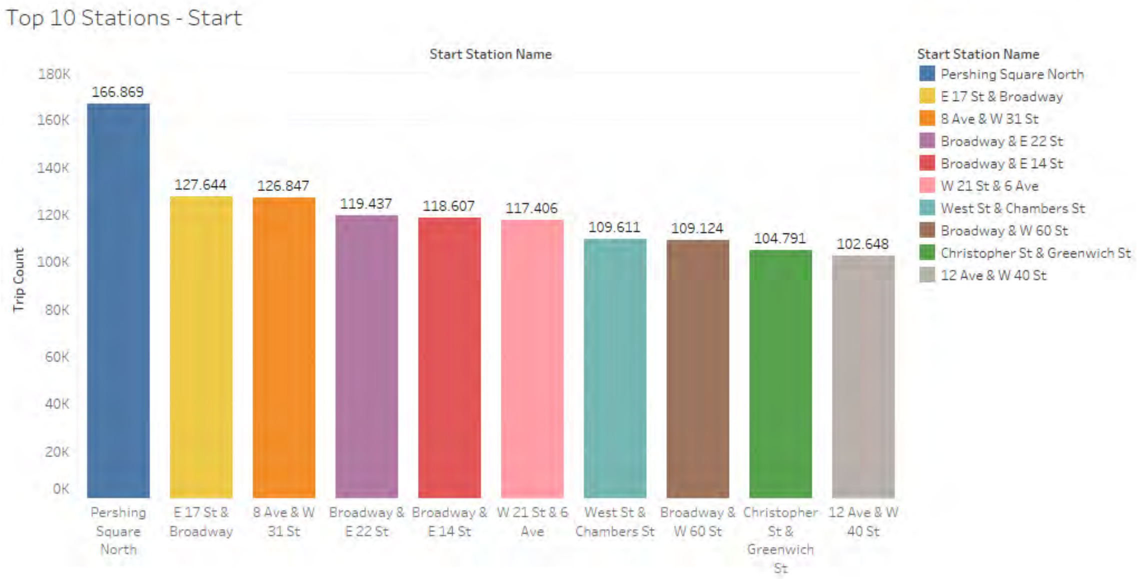


Figure A.9 Top 10 Stations – Start

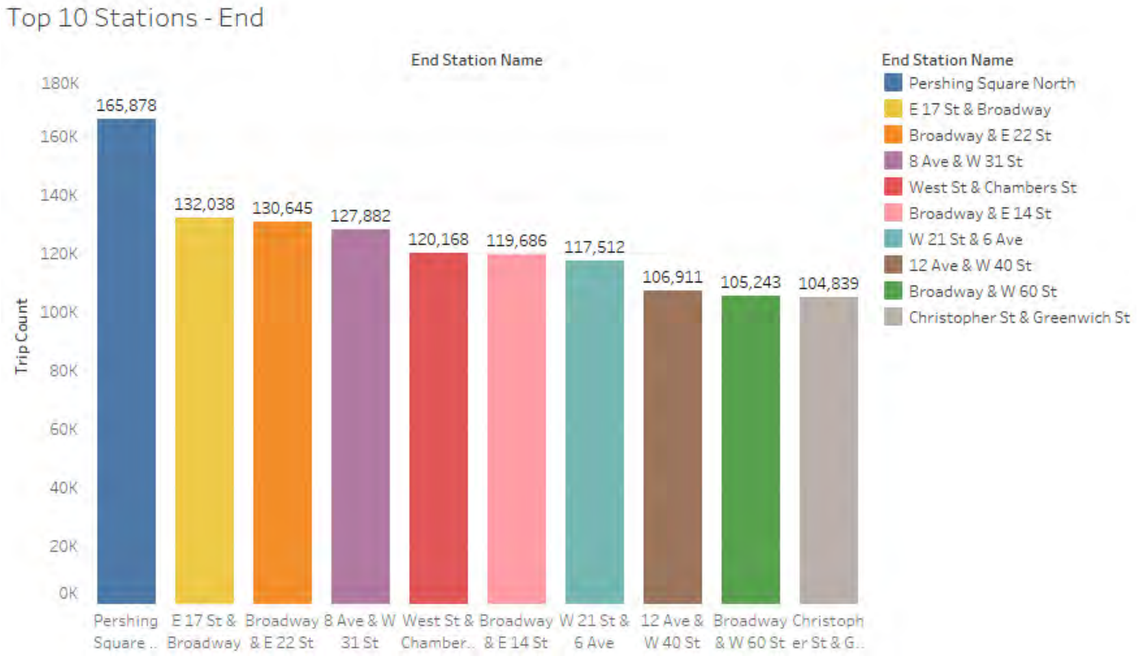


Figure A.10 Top 10 Stations – End

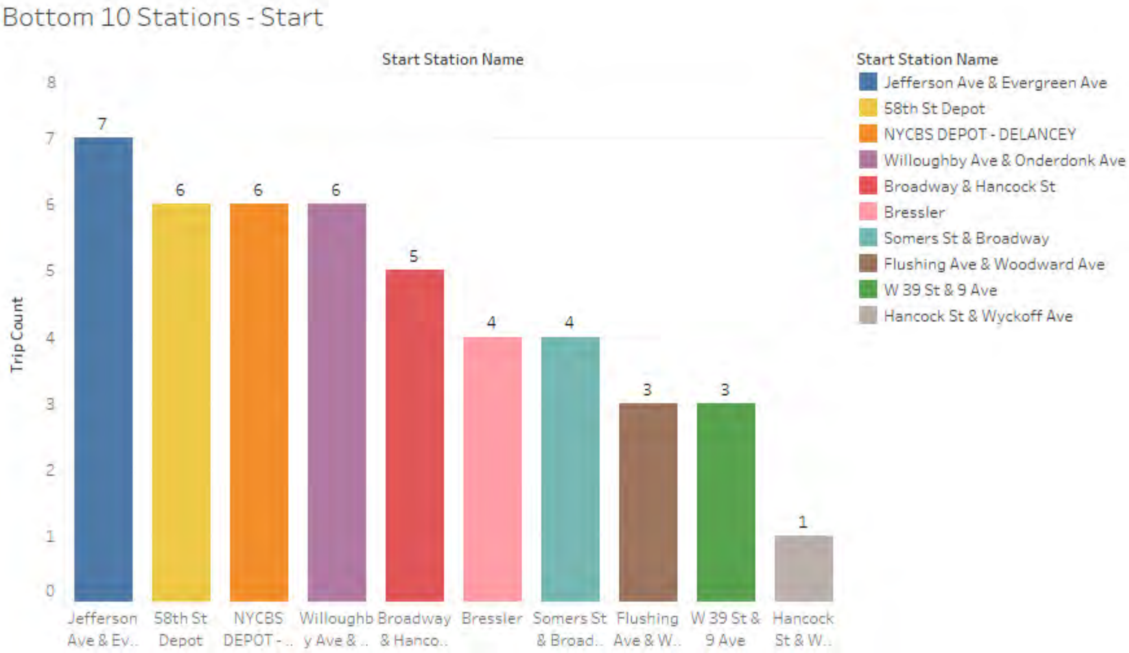


Figure A.11 Bottom 10 Stations – Start

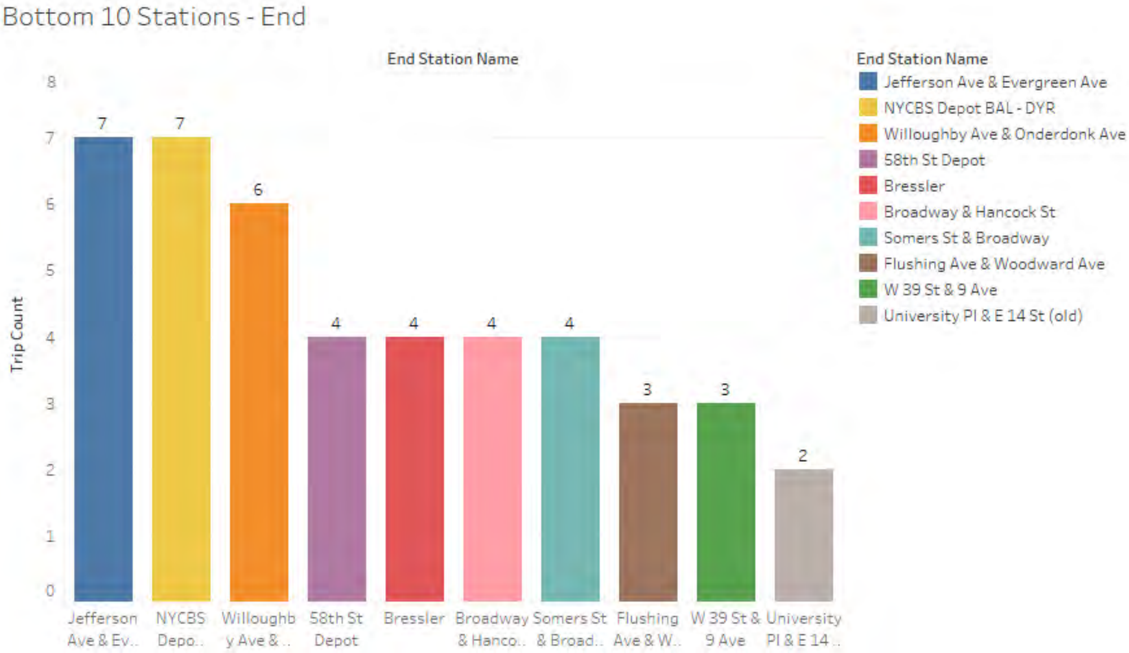


Figure A.12 Bottom 10 Stations – End

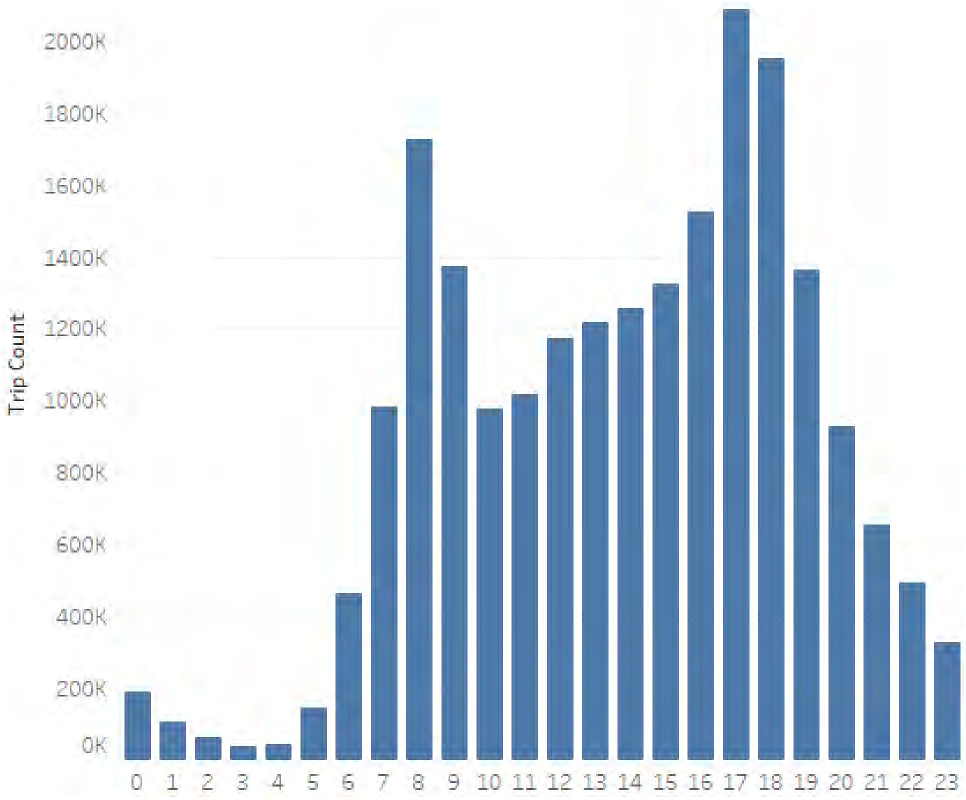


Figure A.13 Daily Trip Count Distribution – Start

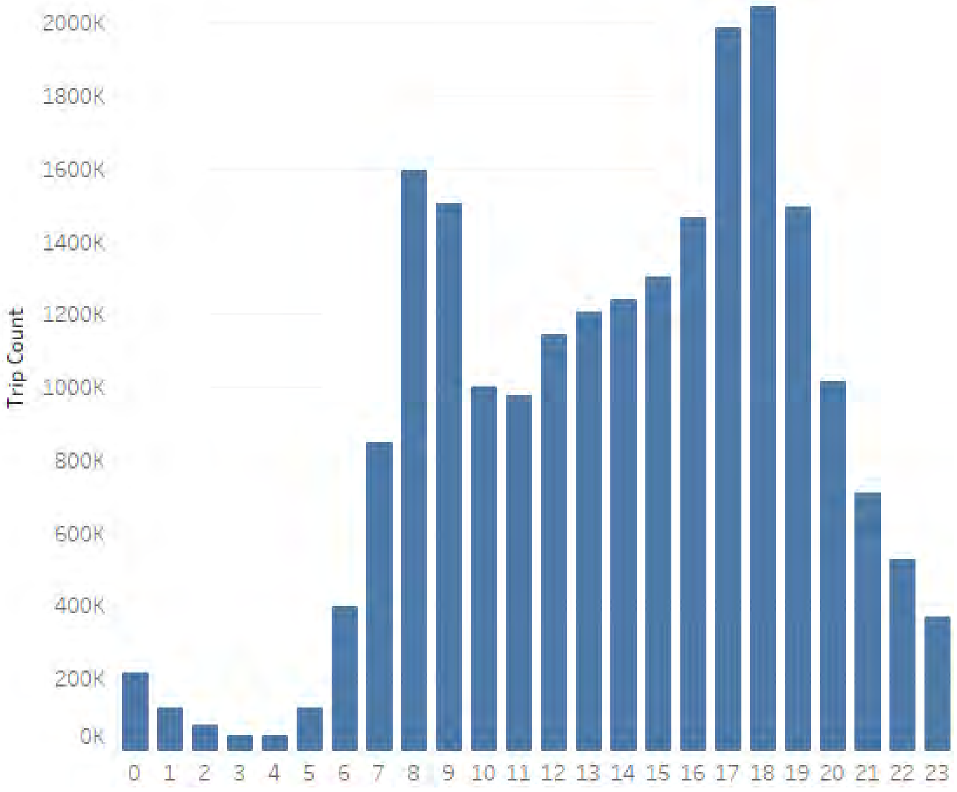


Figure A.14 Daily Trip Count Distribution - End

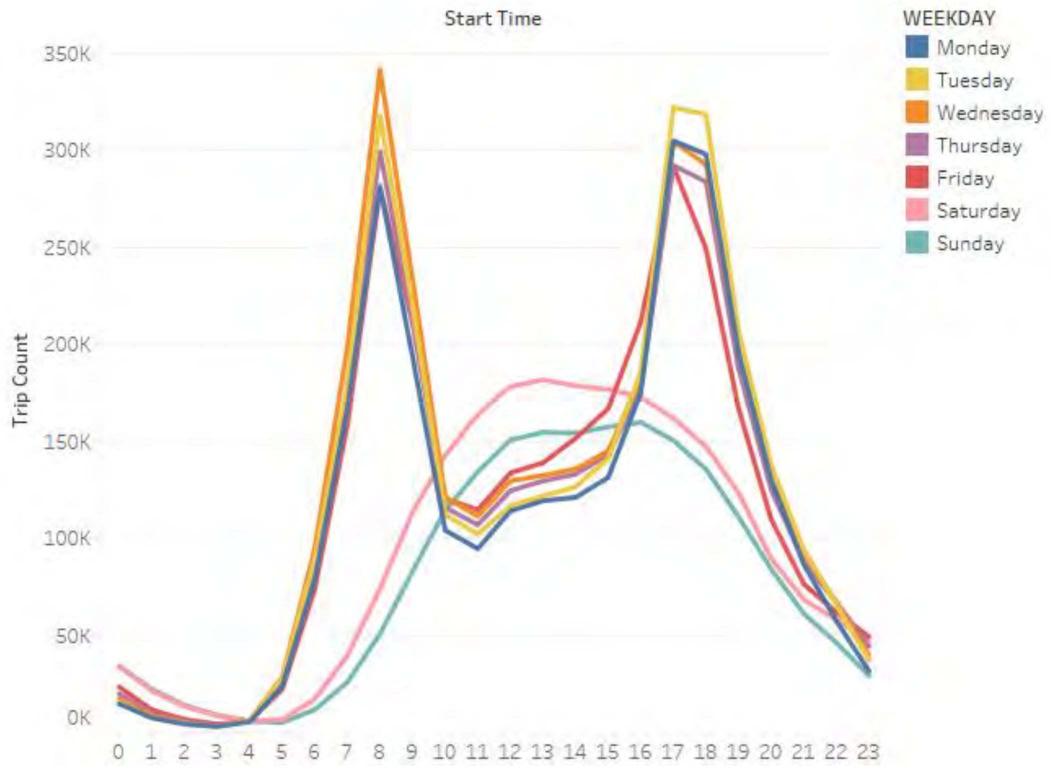


Figure A.15 Daily Trip Count – Start (Subscribers)

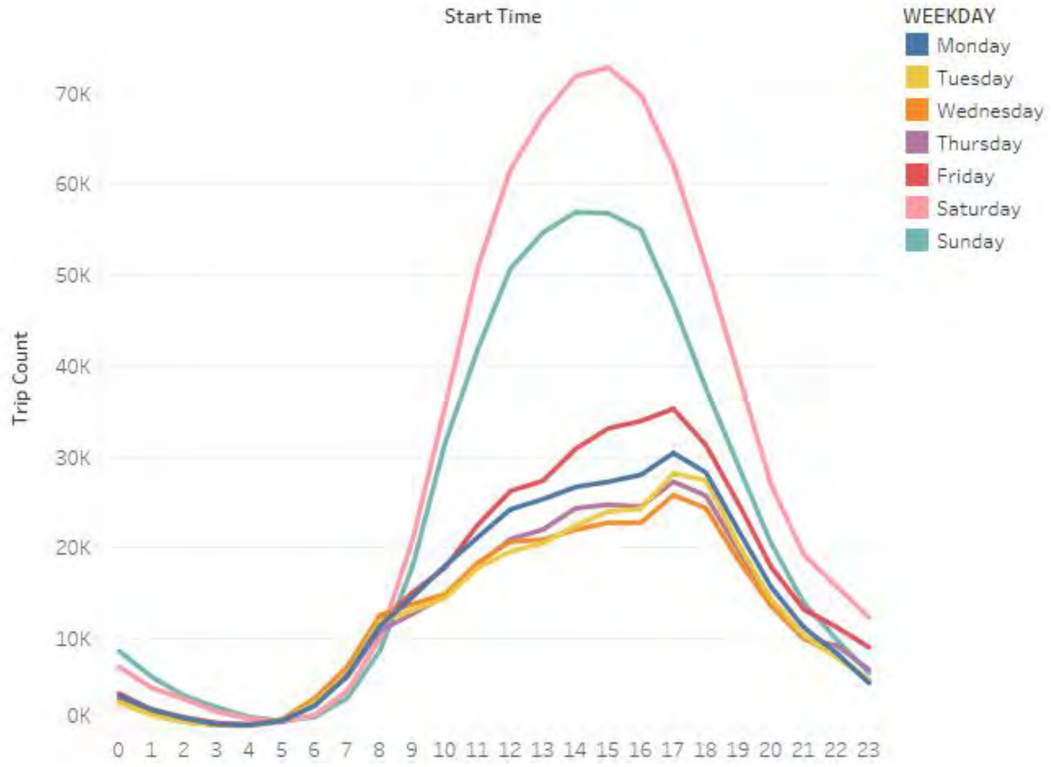


Figure A.16 Daily Trip Count – Start (Customers)

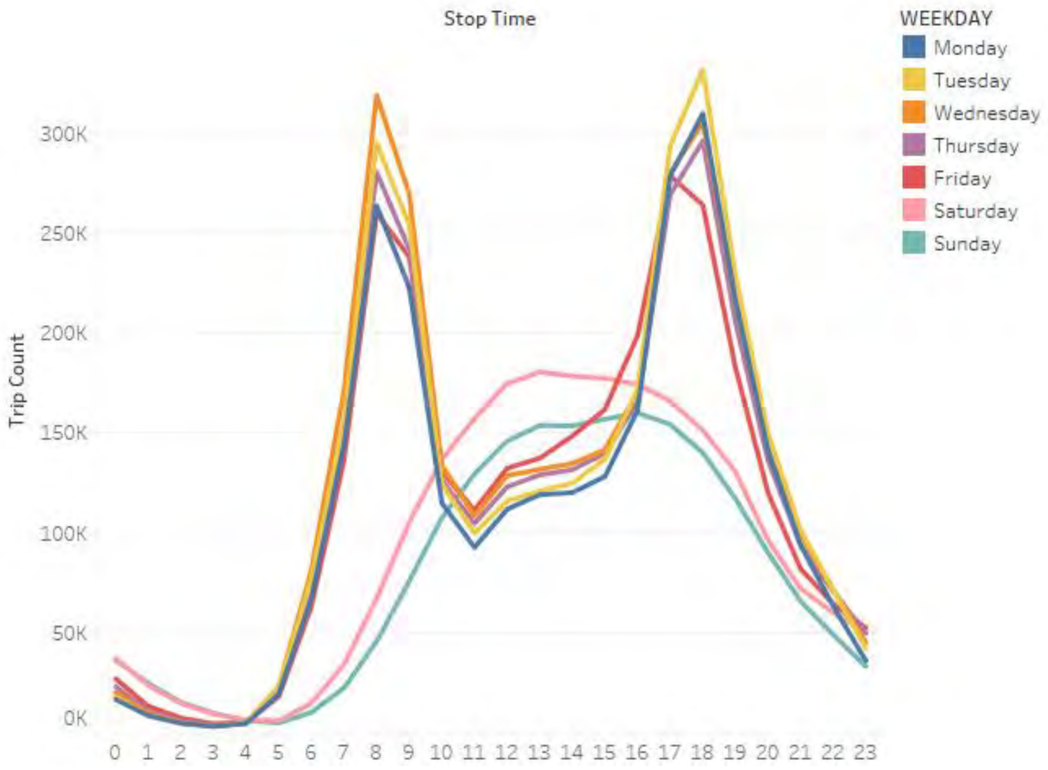


Figure A.17 Daily Trip Count – End (Subscribers)

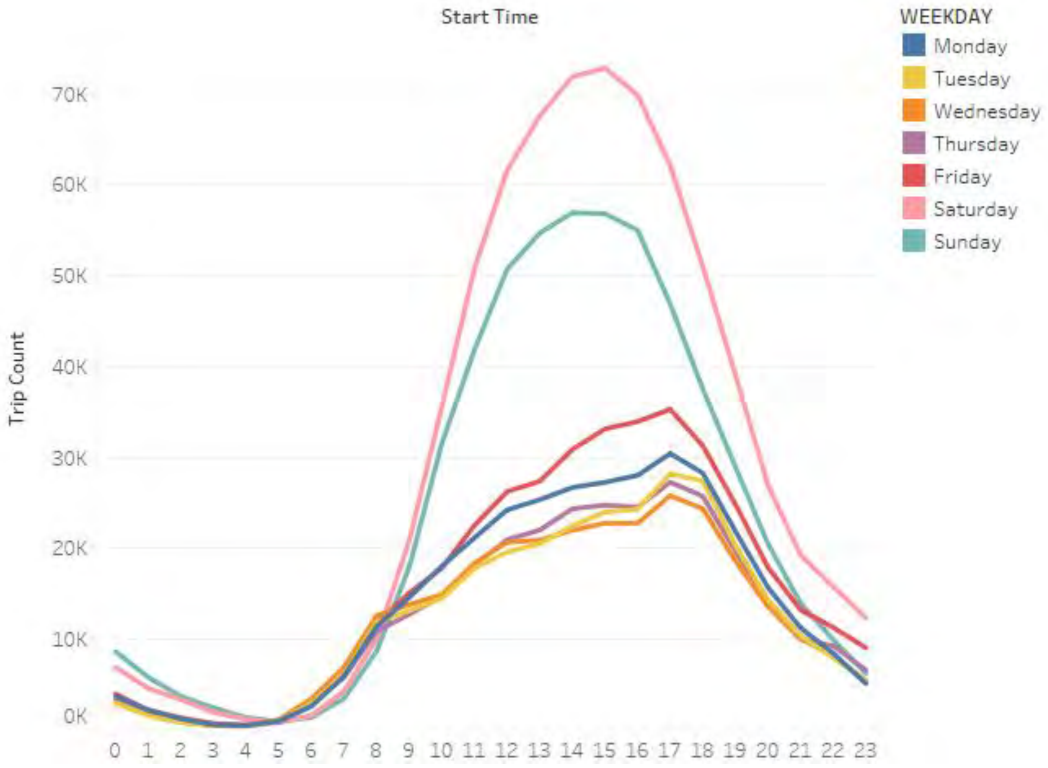


Figure A.18 Daily Trip Count – End (Customers)

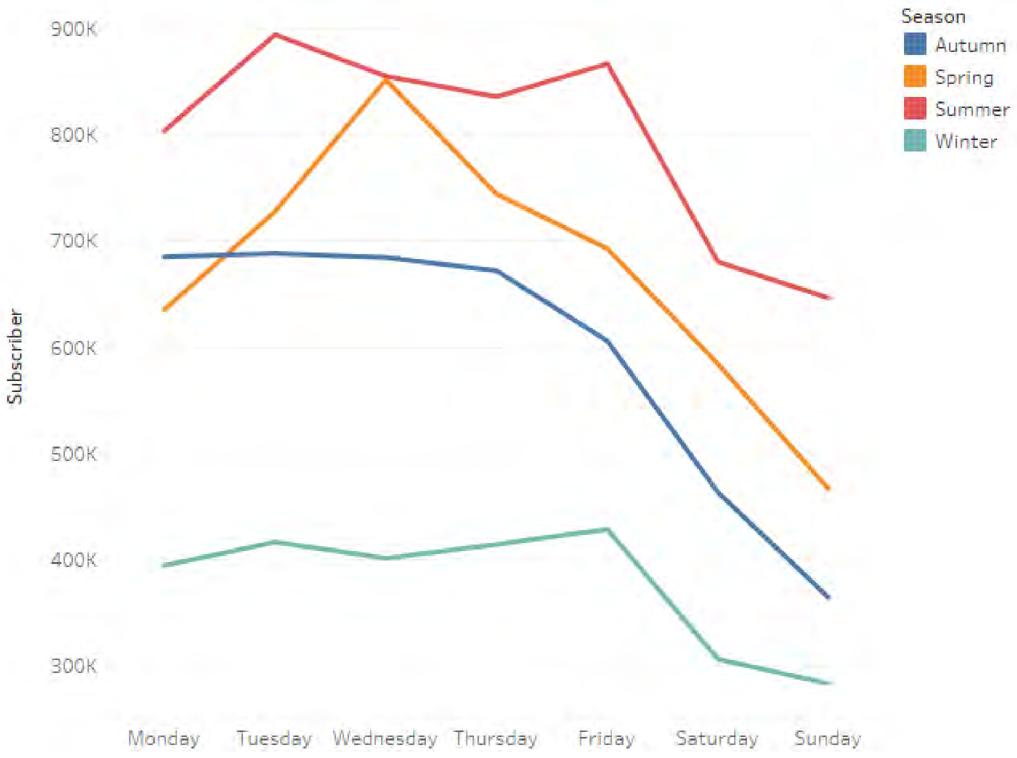


Figure A.19 Weekly Count by Season (Subscribers)

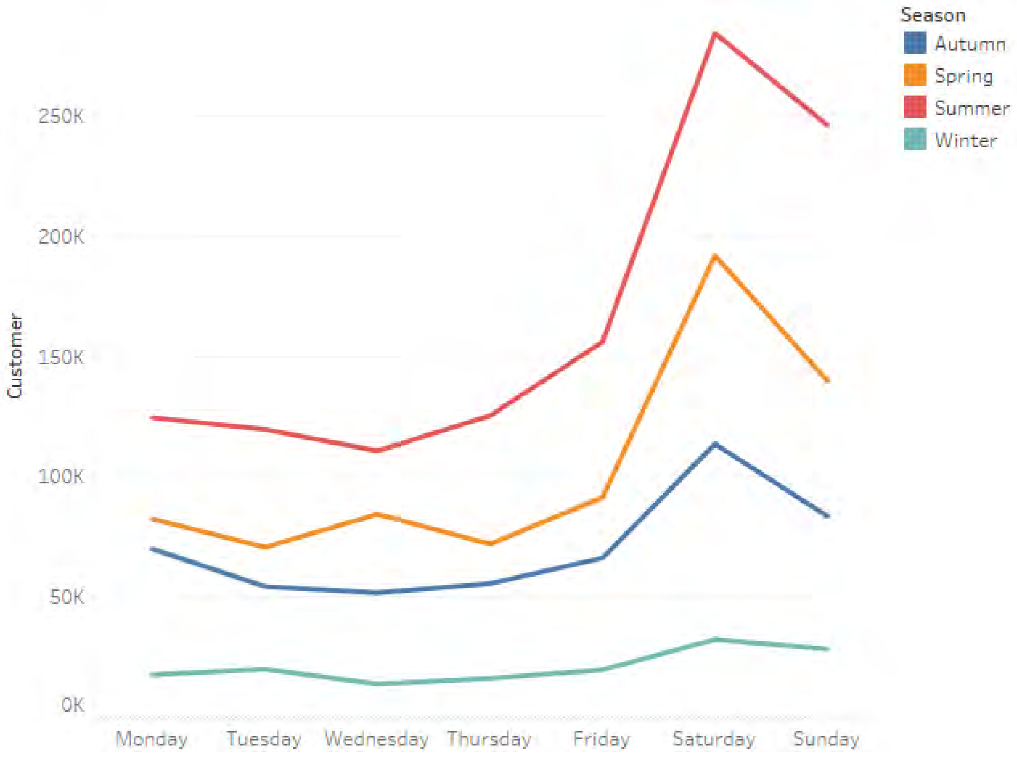


Figure A.20 Weekly Count by Season (Customers)

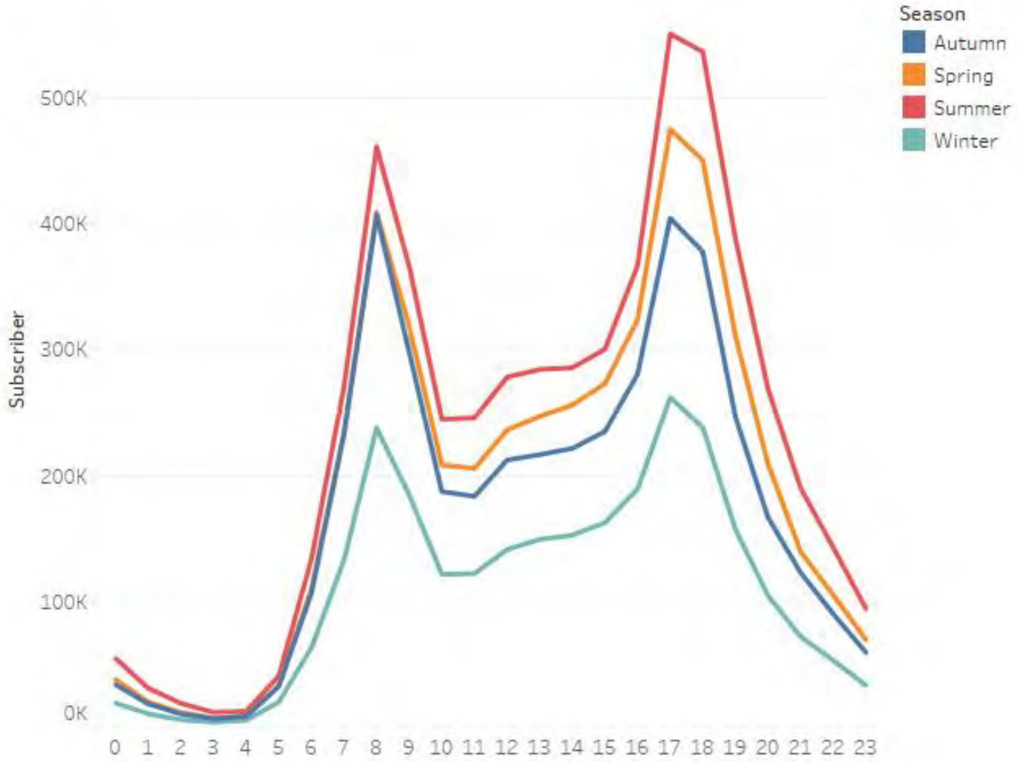


Figure A.21 Hourly Count by Season (Subscribers)

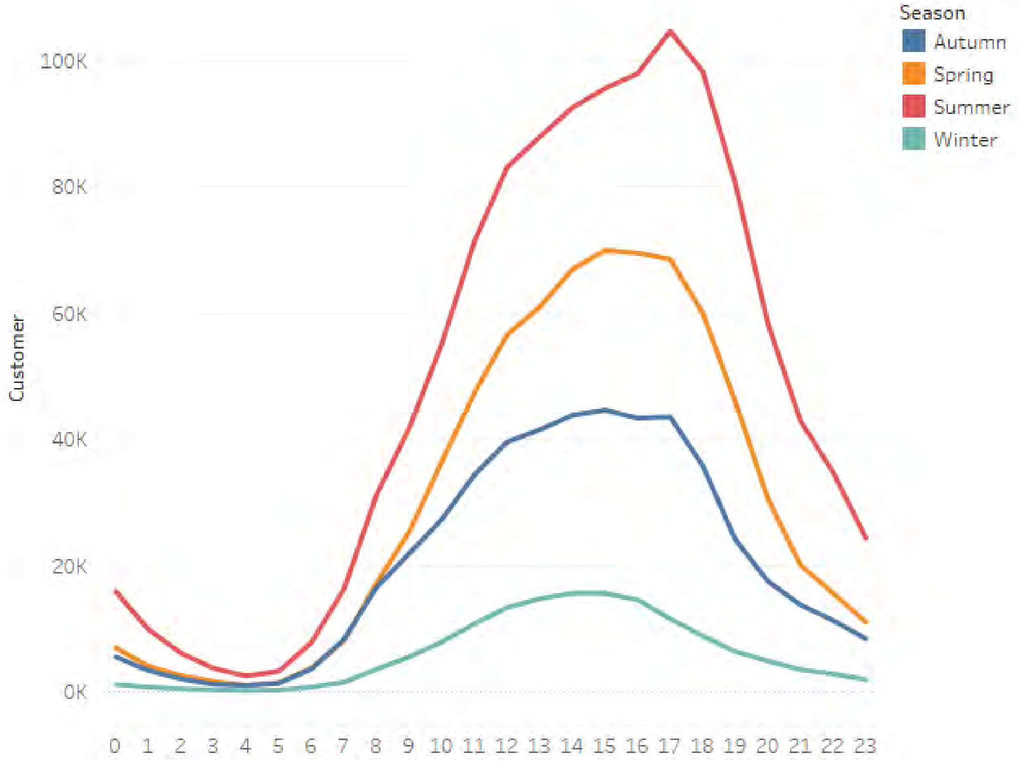


Figure A.22 Hourly Count by Season (Customers)