Metropolia

Ville Hänninen

Time Series Analysis and Forecasting for Taxi Traffic

Metropolia University of Applied Sciences

Bachelor of Engineering

Industrial Management

Bachelor’s Thesis

25 April 2022

Abstract

Author:	Ville Hänninen
Title:	Time Series Analysis and Forecasting for Taxi Traffic
Number of Pages:	58 pages + 0 appendices
Date:	25 April 2022
Degree:	Bachelor of Engineering
Degree Programme:	Degree Programme in Industrial Management
Professional Major:	Industrial Management
Specialisation option:	ICT-business management
Supervisors:	Rakel Peltola, senior lecturer
	Sonja Holappa, senior lecturer

Digitalization has brought ever increasing amounts of data to companies' disposal, but they also struggle to realize the potential that lies within this accumulated data. With the emergence of tech-giants such as Google and Facebook utilizing more and more of the data they procure companies have woken up to the possibility of finding new ways to utilize the data they have. It can take these companies years if not decades to wake up to these new potentials. Although already using technology and digital services, taxi companies can be sitting on over decades worth of data without realizing the potential hiding in this. Presence of historical data and constant stream of new data provides an excellent opportunity to create services that would benefit both the company who owns the data, as well as the taxi drivers who create this data. Traditionally many of the decisions made by taxi drivers has been based on intuition and expertise accumulated throughout years of working in the same area. This creates a problem when the average length of a taxi drivers' career is in decline.

The objective of this thesis was to analyze and identify the best performing forecasting model for the available taxi traffic data. This thesis was based on data that was collected from taxi traffic prior to this thesis. As the data was naturally time series data, time series analytics and forecasting methods were explored. Linear regression, ARIMA, and automatic ARIMA models were compared for their forecasting performance. Four simple forecasting models were utilized in addition to provide a reference point for the bottom-line performance the forecasting models should outperform to be considered better than blind guessing. Python and R languages were utilized to pull, aggregate, analyze, and forecast the data.

This thesis concluded that ARIMA and automatic ARIMA models performed the best, and only had marginal difference between each other while also clearly outperforming the baseline models. This outcome provides the customer company with the required foundation for future work that would be needed to be performed before the completed software could be utilized by the taxi drivers to forecast a location's profitability. Therefore, recommended future work would be to explore the possibility of an AI model for the data and optimizing the solution. The optimization would include work such as exploring different aggregation methods, analyzing the performance of the forecasting model with differing volumes of data, and transforming the code into an API.

Keywords: time series, data analysis, forecasting, ARIMA

Tiivistelmä

Tekijä:	Ville Hänninen
Otsikko:	Aikasarja Analyysi ja Ennustaminen Taksi Liikenteestä
Sivumäärä:	58 sivua + 0 liitettä
Aika:	25.04.2022
Tutkinto:	Insinööri (AMK)
Tutkinto-ohjelma:	Tuotantotalouden tutkinto-ohjelma
Ammatillinen pääaine:	Tuotantotalous
Suuntautuminen:	ICT-liiketoiminnan johtaminen
Ohjaajat:	Rakel Peltola, lehtori
	Sonja Holappa, lehtori

Digitalisaatio on tuonut dataa yrityksille kiihtyvällä tahdilla, mutta yrityksillä on ollut haasteita ymmärtää mahdollisuudet mitä tästä datasta löytyy. Suur-yritykset kuten Google ja Facebook ovat käyttäneet dataa toiminnassaan jatkuvasti enemmän, mikä on puolestaan herättänyt muita yrityksiä datassa olevista mahdollisuuksista. Tämä muutos on ollut hitaampaa yrityksissä, joiden päätoimi ei ole tietotekniikassa, joilla voi kestää vuosi kymmeniä ennen uuden teknologian hyödyntämistä. Taksi yritykset ovat heränneet tähän tarpeeseen uusien yritysten kuten Uber ja Lyft ilmaantumisen myötä, jotka ovat sijoittaneet huomattavia summia teknologiaan.

Vaikka taksi yritykset hyödyntävät teknologiaa useilla eri alueilla ja ovat näin kerryttäneet yli vuosikymmenen ajan dataa ilman että tätä dataa on hyödynnetty kuin pintatasoisesti. Historiallisen datan saatavuus ja jatkuva uuden datan saaminen on pohjia erinomaiselle mahdollisuudelle luoda uusia palveluja, jotka hyödyttäisivät niin dataa kerääväää yritystä kuin dataa tuottavia taksi kuskeja. Suurin osa päätöksistä, joita taksi kuskit tekevät työpäivänsä aikana ovat heidän asiantuntemuksensa varassa, jonka he ovat kartuttaneet vuosien työkokemuksen avulla. Tieto siitä minne kuskin kannattaa suunnata päivän minäkin tuntina voi olla haastavaa monista syistä ja siksi loistava kandidaatti datan avulla luotavalle tietopohjaiselle päättämiselle.

Tämä opinnäytetyö käyttää hyväkseen saatavilla olevaa dataa ja antaa suosituksen parhaasta ennuste mallista. Data tuodaan ensin tietokannasta, puhdistetaan, kootaan kahteen eri aikasarjaan, analysoidaan ja ennustetaan. Lopuksi ennusteiden tarkkuudet analysoidaan, josta paras ennuste malli voidaan valita. Tämä opinnäytetyö vastaa kysymyksiin siitä pystyykö ennustemallit tuottamaan parempia ennusteita kuin pohjamallit ja mikä ennustemalli tuottaa tarkimman tuloksen. Tämä luo tarvittavan pohjan tulevalle työlle joka tulisi tehdä, jotta taksi kuskit saisivat käyttöönsä lopullisen ennustetyökalun, jonka avulla he voivat tehdä päätöksiä sijainnin kannattavuudesta.

Avainsanat: aikasarja, data analytiikka, ennustaminen, ARIMA

Contents

List of Abbreviations

ACF:	Autocorrelation function. Plots correlations between observations with the lagged version of the observations as the function.
AICc:	Akaike information criterion with correction. A model to predict error in the forecasting models.
DT:	Data table. A way to display information with rows and columns.
MAE:	Mean absolute error. Function to calculate model error which represents error values equally.
PACF:	Partial autocorrelation function. Plots correlations between observations while controlling for correlations between other lags.
RMSE:	Root-mean-square error. Function to calculate model error which represents higher error values more.
TS:	Time series. Series of data represented in a chronological order.

1 Introduction

The commissioner for this thesis was Routa Digital which is an IT start-up company with numerous ongoing software development projects. With their focus on the Nordics, Routa Digital offers various services to their customers; consultation, digital transformation, and software engineering. Besides services, Routa Digital is also able to offer products for multiple industries such as logistics, e-commerce, travel, and wellness. This selection of offerings is quite diverse, and thus the possibilities of the value that can be brought to the customers ranges from ready-made solutions to fully customized services. (Routa Digital 2021)

There have been many different opportunities for data analytics in Routa Digital since their customers are from many different business sectors. Due to this there was an opportunity to consider multiple different topics for this project, which were mainly different utilizations of time series forecasting for varying types of datasets. From these topics one was chosen. This was due to the availability of data in accordance with the timeline for this thesis project, and flexibility of using the solution for future applications in other projects as well.

The objective for the chosen topic was to create a forecasting model based on the data from the customer company. This forecasting model would be able to provide an accurate prediction of the demand of taxis at the customer pick-up area that the customer company can utilize. This same information can then be further relayed to the taxi companies and drivers. The product of this thesis, the forecasting model, could then be integrated as part of the larger mother project for the customer company. The outcome from this thesis then was a forecasting method that can be integrated as part of a larger project for the customer company.

1.1 Current State

All around the world data is being produced at exponential rate by companies and almost every company has a large amount of untapped potential when it comes to data, data collection, and data utilization. Many companies are unable to see the value hiding in their data, while also lagging the knowledge to utilize this further. This is especially true in companies that do not have their core competence in data, or IT in general, such as taxi companies. (Statista 2021)

Taxi drivers for example need to make several decisions by themselves throughout their working day. What route to take, where to head to after the current customer, how long they should wait in their current location before switching elsewhere. These decisions are often left up to intuition, aided by some devices such as a GPS or list of incoming flights, but this does not help with knowledge on where the taxis would be needed the most at any given time. Several places continuously provide customers for the taxis such as airports and train stations, but the specifics on how many taxis and at what time is still unknown. This unknown creates uncertainty for anyone related in the taxi industry, which can be alleviated with proper use of data, analytics, and forecasting.

This thesis focuses on a customer pick-up area with defined entry and exit borders for the taxis, giving continuous data on the vehicle traffic going through the area. This provides an accurate current picture of the situation at the location at any given point in time. Each observation that comes into the database includes a timestamp associated with it, providing a clear timeline. This way of representing data is called a time series.

Time series has numerous different analytics and forecasting methods that have been developed as the data is naturally ordered in time. This single differentiating characteristics makes time series unique from other types of data in that time series observations have natural ordering, do not have relations to geographical locations, and the ordering is one-way in the sense that the observations are derived from the past values, not future ones. Due to this, methods for analytics and forecasting unique to time series was utilized with this data.

1.2 Project Plan

The first stage of the project was to gather theory regarding time series analysis and forecasting. This gave the foundation required to work with the data provided in the next stage, which was to collect, clean, and validate the data. Customer of the Routa Digital acted as the source for the data used in this project. The second stage of the project involved training the models with the different time series data that was collected and cleaned, utilizing the different forecasting tools available and putting the theory to test. The final stage of the project was testing the accuracy of each one of the models. From here the best performing model was able to be chosen and decided whether the model performed satisfactory results, and what could be done as future development to improve upon this project.

The scope of the project was limited to a singular location, from which the model can be expanded upon to any other locations with similar available data. But this optional expansion is outside the scope of this project. This project was limited to providing a forecasting model that is given input time series data from an input, processing the provided data according to the model produced by the thesis project, and giving the result forecast as an output. These outputs were then compared against each other to find the best model. Information gathered in this project was then relayed to Routa Digital and the customer, who can utilize the findings in further development.

2 Existing Literature on Time Series

The event data in this project came from the taxi movements, which by its nature provides a time stamp for the events as the taxis enter and exit the pick-up location. This creates a time series. Time series data in general can come in many different forms but should typically have one common characteristic, which is the data being naturally ordered in time. Due to the broad definition and commonly used way of collecting data, time series can be found everywhere. Cars in traffic, income and expenditures of a company, tree growth, and climate are all vastly different from each other, but are all examples of available time series data. Clear differences still exist between the different time series data that are available. The time intervals between observations, time scale, and where the data is taken from, all need to be understood in creation of a forecast to create as accurate of a model as possible. (Palma 2016)

Time series is a sequence of observations recorded at regular intervals, which are typically hourly, weekly, monthly, quarterly, or annual. More specific needs exist for seconds or minute-wise time series which can be used for example for website analytics. Analyzing time series data is required as a preparatory step before developing forecasts of the time series. Time series forecasting has large commercial significance since large amount of business-related data is naturally time series. Demand, sales, visitor counts, stock prices are all examples of time series.

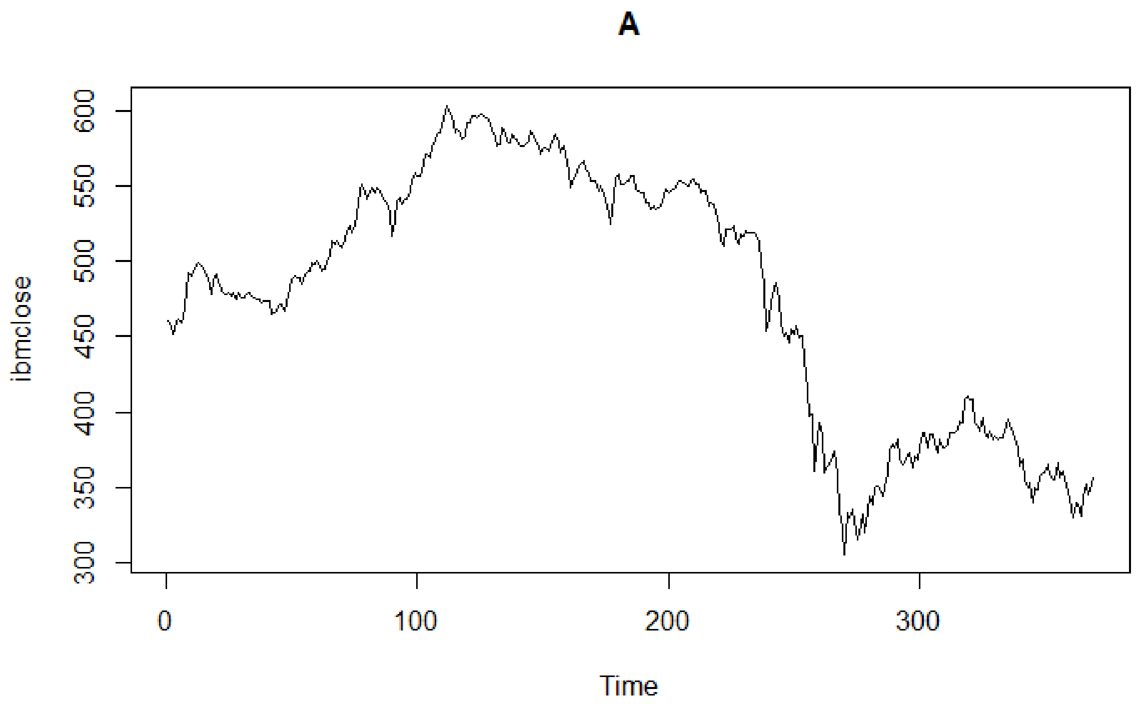
Although time series is primarily used for numeric data and the methodologies often assume numerical data, in most cases data can be transformed from nonnumeric to numeric data. Ratings from customer feedback can be used as an example for this. Customers are asked to rate their experience between good, neutral, and bad, which in turn can be transformed into a numerical range of zero to two, assigning each rating to a number on the scale. Another common problem that comes from working with time series is if the observations in the dataset have been taken at irregular intervals. Time series that have observations at irregular intervals requires more complex techniques, but in some cases, it is possible to approximate the observations into regular intervals and then proceed to use the regular techniques. However, this may lead to different levels of inaccuracy in the analysis depending on the method used to process the irregular intervals. (Palma 2016)

Each set of time series data can have certain characteristics to them, which need to be considered before being able to create analytics and forecasting. Stationarity, seasonal patterns, and trend curve need to first be examined and possibly isolated from the time series before the data can be forecasted. At the same time seasonal patterns and trend curve of a time series can also be considered as analytics on their own, which can lead to insights that have not been discovered from the data before. It is possible to achieve better quality forecasts when these three characteristics found in time series are combined and considered when creating the forecasting models. Due to this, it is important to understand these different characteristics better.

2.1 Stationarity

Stationarity refers to the characteristics of the time series data. It is a term used to distinguish between time series of which statistical properties do not change over time. This is important to time series analytics and forecasting as majority of the models used assume the data to be in stationary format. Although the observations in the time series are allowed to change over time, their properties do not change. An example of these changing properties would be seasonality or trend, while white noise data is stationary. (Palachy 2019; Hyndman & Athanasopoulos 2018)

Figure 1 demonstrates the difference between stationary and non-stationary time series well. Series (A) shows very distinctive trend curve that comes from the stock prices being dependent to the historical values, whereas the series (B) shows no distinct characteristics as this illustrates the daily change that the stock prices had.



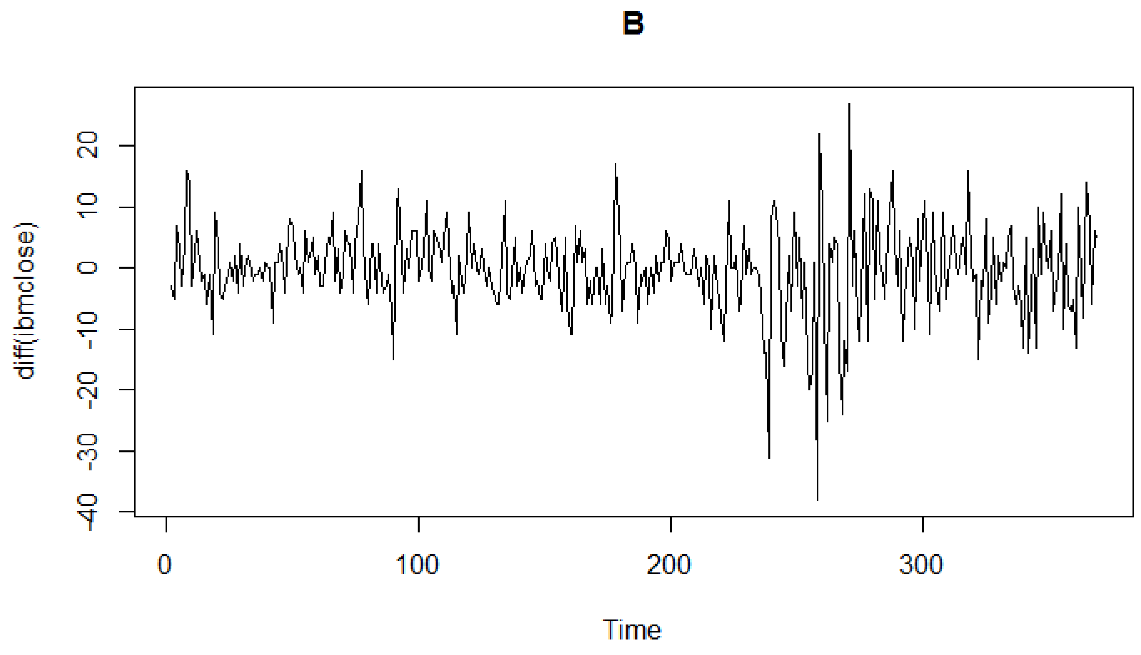


Figure 1. (A) Daily closing IBM stock price; (B) Daily change to the closing IBM stock price. (based on fma data, 2020)

As seen in Figure 1, the difference between stationary and non-stationary time series can be easy to notice by visual inspection, although cases exist where this is not as clear. Figure 2 illustrates plot of the annual number of lynx trapped, which gives an indication of the total population amount in the area. The pattern of the plot would provide a strong indication of the time series being cyclical, and thus being non-stationary.

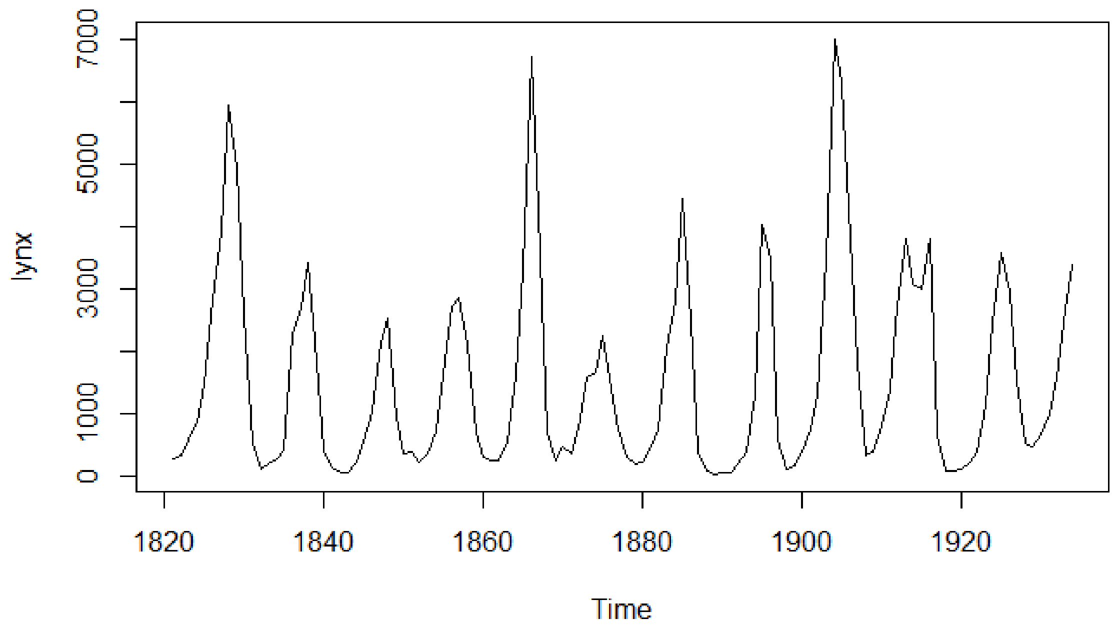


Figure 2. Annual number of lynx trapped in McKenzie river district of northwest Canada: 1821–1934. (based on fma data, 2020)

Closer inspection of Figure 2 however reveals that the cycles are aperiodic, and thus the cycles are not predictable. Due to this, this time series is in fact stationary.

This showcases the importance of careful analysis of the time series data before using any of the analytics or forecasting methods detailed later in the forecasting section. Multiple methods exist that aim to convert non-stationary time series into stationary ones, such as differencing. Many different tools can be utilized to better identify stationary time series from non-stationary ones in more unclear cases such as the one seen in Figure 2.

2.1.1 Autocorrelation and Partial Autocorrelation Function

Autocorrelation function and partial autocorrelation function plots was the main tools that were used in this thesis project to identify non-stationary time series. Correlation measures the relationship between two variables meaning these variables move in coordination with one another. In time series this can be used to see the amount of correlation between observations; two variables where one is typically time. The problem here would be the need to compare individual values one by one to arrive at a proper conclusion. As such, an autocorrelation function, ACF, was created. ACF provides the correlation between the lagged values of a time series, comparing the observations to the past observations and combining these for a total correlation amount for each lag amount. Lag is a measure of how far back the comparison observation is. This provides an accurate view of how related the current value is with the past observations. (Hyndman 2018)

Partial autocorrelation function, or PACF, is a derivative method from ACF. This need comes as there can possibly be certain hidden relations left which are not visible using only ACF. For example, there might be relations between the lagged observation that cannot be seen since they are overshadowed by the ACF relations. Due to this PACF removes the relations found in ACF, leaving behind residuals and these residuals are what PACF then compares to find possible correlations. PACF thus ensures that as little correlation as possible is left in the time series, improving the effectiveness of the analytic and forecasting methods. (Hyndman 2018)

Figure 3 showcases how certain time series characteristics can be detected in the ACF and PACF plots. Top plot illustrates the time series of monthly totals of international airline passengers, which shows an ascending trend with seasonality. ACF plot reveals an observable descending waveform while the PACF plot reveals exponential decay.

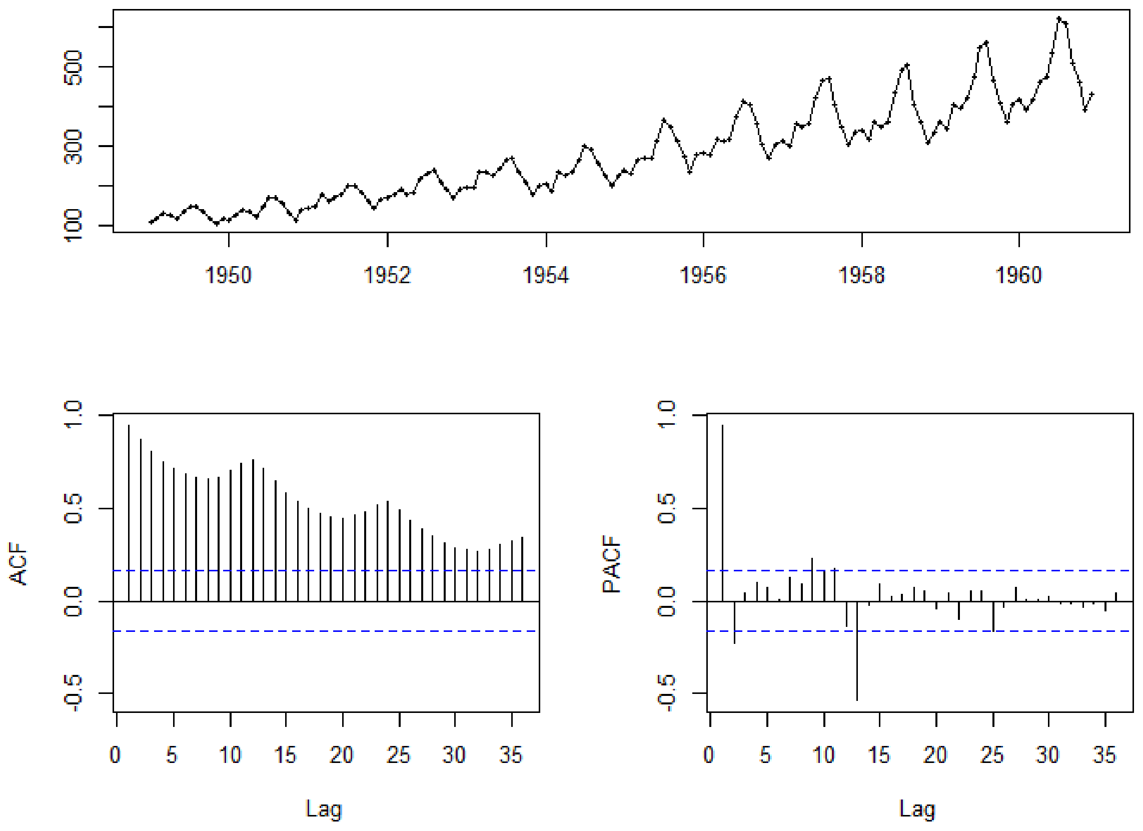


Figure 3. (Top) Monthly totals of international airline passengers (1949–1960). (Bottom-left) ACF plot of the monthly international airline passengers. (Bottom-right) PACF plot of the monthly international airline passengers. (based on fma data, 2020)

As seen in Figure 3, from the top plot it is possible to visually notice the upward overall trend with the number of passengers, while also having apparent seasonal spikes. ACF plot below confirms this seasonality with the waveform observable in the ACF plot along with the slow overall descend which was a result from the trend seen in the time series. PACF plot however reveals exponential decay which would indicate there not being any significant relations between the residuals. However a significant spike was noticeable at lag thirteen, significance of which will be detailed in a later section.

2.2 Characteristics

Time series data can include several different characteristics that make it non-stationary. As detailed in the previous section, these characteristics should be identified and removed to create stationary time series before applying analytics models to them. These different characteristics can be broken down into trend, seasonality, and cycles.

2.2.1 Trend

Trend is long-term change in the time series that shows as gradual increase or decrease over-time. The gradual change happens for any amount of time and then eventually disappears, which can then be followed by another upward or downward trend. The key characteristic in trends is that they are unique, they do not repeat in a set pattern such as seasonality does. Due to this the main way to identify trend is with the relation that the observation has to the past observations, which is why in the ACF plot it was observable by the gradual decrease in the relation amount. (Hyndman 2018)

Many reasons exist why trend would appear in time series data, and this would require closer inspection in the conditions and environment from which the data is collected. Figure 3 demonstrated an upwards trend of the amount of airline passengers which could be explained by decrease in cost of travel by planes, increase in wages, increase in globalization in business, or increased interest in traveling to far away locations; which in turn can be due to media showcasing and raising interest in foreign countries, publications on travel, or famous personnel traveling. As such, it would be very difficult to forecast these changes in the environment and account for all of them ahead of time. But it still would be important to be aware of these changes, and if the possibility arises, considering the larger contributors to changing trends in time series data.

2.2.2 Seasonality

Seasonality refers to changes in time series that happen in a set time period, repeating according to a specific schedule. This means that the time series is affected by when the observation was made. Similar to trend, seasonal changes can be either increasing or decreasing, but they are happening at fixed frequency, which allows the forecast for them. These changes in time series can happen at any interval, be it yearly seasonality such as the amount of people catching the flu during the fall and winter months, or much more often for example the amount of people traveling during Fridays and Sundays. (Hyndman 2018)

Such as with trends, there can be any number of variables contributing to the seasonal patterns in time series, and it can be increasingly difficult to know these beforehand. As is often the case, time series can be used to first discover the seasonal patterns which is then followed by an investigation into what causes the seasonal patterns. There also exists the possibility of single time series having multiple seasonal patterns affecting it at the same time, not only demonstrating simple annual patterns in the data, but also demonstrating weekly patterns. This can get even more complicated if the data is coming in at even more frequent intervals, such as hourly or even minute by minute data. Stores for example can show daily patterns where people shop more often during rush hour periods, and annual patterns such as increased shopping during holidays such as Christmas. (Hyndman 2018)

This complex seasonality can prove a difficult problem that should be approached with consideration. Considering what data the time series is observing and how much data exists can be valuable in determining what should be accounted for. There might not be enough data that some of the longer period seasonality should be considered, such as annual seasonality. Another possibility is that the observable data can be expected to be changing long term, such as climate, that the shorter period seasonality should not be accounted for.

2.2.3 Cycles

Cycles show in the time series data as long periods of change in the observations. Common possibility for many time series is not to show cyclic behavior as the cyclic periods usually span from a few years to decades. The problem can be that not all data includes this many historical observations available. In cases where cyclic behavior has not been observed yet, this change can show up as a trend in the time series, and it will be impossible to tell otherwise without longer term data being available. Due to this, cyclic characteristics in data can be rarer to encounter when compared to trends and seasonality. (Hyndman 2018)

2.3 Forecasting

Forecasting with data is especially important for companies and is the basis for fact-based management. Understanding the demand of a product and being able to anticipate optimal times for different actions is crucial for all companies; when to order in more items or how many new employees should be hired in the coming months are both examples of time series forecasting, even if it can already be done by a person without them acknowledging this consciously. Of course, with data and reproducible forecasting methods this can be done to a higher degree of accuracy and reliability.

With time series it is important to understand how certain datasets will frequently be more precise and easier to forecast than others. It should be understood what affects the observations and how much data is available to be analyzed as both contribute to this. Decade’s worth of daily observations of sunrise times will provide a very accurate forecast whereas predicting lottery numbers will provide very inaccurate results. Then the third factor to the accuracy of the forecast results is whether the forecast itself will influence the future observations. Stock market already uses time series forecasting and depending on the results the models provide will end up affecting the future observations themselves. This becomes even more chaotic when it is considered there being multiple different models trying to predict the same observations, all of which will end up affecting the future observation values. This results in difficulties to predict stock markets that can be affected by any number of outside factors, even if decades of past observations are available to build the model from. (Hyndman 2018)

It becomes important to understand the environment the observations are coming from as time series data can be from widely different sources. Although every environment is ever changing, vast differences still exist between how volatile the environments they come from are. Stock markets that have been constantly changing in the past will continue to do so in the future, whereas it is quite certain that sunrise will still be happening according to the same pattern as in the past. (Hyndman 2018)

Many different forecasting methods exist, the simplest of which just takes the average of past observations, with more complex methods utilizing machine learning and neural networks. These simpler methods are usually not used for forecasting itself but as a baseline that the actual model should be able to outperform. (Hyndman 2018)

This thesis focuses on quantitative forecasting methods, which assumes past numerical data for what would be forecast. Many available methods exist for cases in which no quantitative data is available which would use qualitative methods instead. These qualitative methods will not be used as this project only uses quantitative data.

2.3.1 Explanatory Models

Explanatory models refer to the method of forecasting time series that also takes into account outside factors. This external information can be taken from any possible source but requires knowledge in what affects the observations. For an example hourly electricity demand can depend on current temperature, strength of economy, population amount, current time, and date. These external factors as a result can help explain the observation values. This is different from purely time series forecasting in that time series only considers the historical data of the observations. Time series does not include external factors besides this. Third form of modelling also exists, which has various names such as dynamic regression model or transfer function model. These models involve both of mentioned methods in a mixed model, that includes both the time series observations and the external factors. (Hyndman 2018)

The challenge with using any model that includes external factors is the need to understand what affects the observations. Temperature will affect the energy demand more than daily stock prices. It can be difficult to say beforehand which of these models performs best, therefore if possible, it would be best to compare the results of all the models to find the best fit. This may not typically be realistic due to limitations to available resources; ease of access to the data and the amount of data, time constricts, and computational limitations all play into which approach would be the best for which situation.

2.3.2 Regression Models

As one of the simpler models, linear regression aims to use a variable to predict another variable. This method works by drawing a line with lowest error squared through a plot with the observations of the two-variable laid out on it with the line passing through them in a way to minimize the discrepancy. The error amount then comes from how far away the actual observations are from this prediction line. R-squared is often used as the way of measuring the accuracy of a linear model. (IBM 2021)

Nonlinear regression follows closely to linear regression, with the difference being that the line is polynomial. These forms of models can take the shape of any number of different curves, each one having their own equation. However, it is not possible to measure the error of this curve with the R-squared value. Although nonlinear regression can provide better fit on data, it can be much more resource intensive to use than regular linear regression. (Frost 2017)

2.3.3 ARIMA Models

ARIMA models are widely used for time series forecast models, with its main aim being able to describe the autocorrelations found in the time series data. ARIMA models consist of three main parts; AR, I, and MA which stand for autoregressive, integrated, and moving average respectively. This creates the acronym, ARIMA. ARIMA models are denoted in the form of ARIMA(p, d, q) models. Each of the letters standing for different values; p is the order of the autoregression, d is the degree of first differencing involved, and q is the order of the moving average. As the ARIMA model consists of multiple different models, there exists certain special cases. These special cases are showcased in Table 1, where the first column contains the name of the model and the second column contains the ARIMA denoted model.

Table 1. Special cases of ARIMA models. (Hyndman 2018)

White noise	ARIMA(0,0,0)
Random walk	ARIMA(0,1,0) with no constant
Random walk with drift	ARIMA(0,1,0) with a constant
Autoregression	ARIMA(p,0,0)
Moving average	ARIMA(0,0,q)

Table 1 shows how it is still possible to create purely autoregressive, random walk, or moving average models with ARIMA. Due to the way that ARIMA notation works, it is possible to end up with models that would not count as ARIMA models even if ARIMA function is being used.

Autoregressive, AR, model uses the previous observations in a time series as the predictors for the future values, using linear regression to count for the residuals. As such the time series is “regressed” on the past observations. If ARIMA(p, 0, 0) model is assumed, the value p denotes how many immediately preceding observations will be used for the prediction. (Pardoe etc. 2021)

Moving average, MA, model uses past forecasting residuals in similar way to regression models. Moving average is not often used for accurate forecasting by itself, but instead as a way to smooth out rapidly changing observations for a more general picture. For ARIMA(0, 0, q) model, the value q denotes how many past values are included in the calculation of the average. With higher values the MA value will be less affected by sudden changes in the observed values. (Radečić 2021)

ACF and PACF techniques can be used to acquire estimates for the p and q values. This requires certain assumptions and will not work in all cases. Firstly, both values p and q cannot be positive, while the value d cannot be zero. If the value d was zero, then the model would become purely AR or MA model. Value p can be estimated if the ACF plot decays exponentially or the plot is similar to a sin wave, with a significant spike at specific lag in the PACF plot. Value q can be estimated in similar matter, except with the PACF plot decaying exponentially or having plot similar to a sin wave, and there being a significant spike at specific lag in the ACF plot. For each of these, the value that would be used for the p and q values, would be the lag at which this significant spike was observed. (Hyndman 2018)

Many existing code libraries can be used to automatically determine the values for ARIMA models in cases where the (p, d, q) values cannot be estimated. ARIMA models do not automatically take seasonality into account, but there exist methods for this. ARIMA models that take seasonality into account are called SARIMA models, the added S standing for seasonal.

2.3.4 Neural Network Models

Neural network models go by acronyms ANN (artificial neural network) and SNN (simulated neural network) and are a subset to machine learning, and by connection AI. Neural networks were created to replicate the functionality of the human brain, and thus also received its name. Neural networks have been utilized for deep learning algorithms. ANN works by having multiple nodes between the input and output, each node holding a certain set of instructions and characteristics, connecting to other nodes. Once a certain set of requirements are met, they forward their information to the next node which starts its own process. This goes on until the output is reached and the model provides out an answer. (IBM 2020)

With each of the previously mentioned models, historical data is required to train the model. ANN specifically benefits from large amounts of data but does also require large amounts of processing power to create the models. However, with ANN it is possible to achieve high quality models for tasks that have historically required much manual human labor. Unstructured data has especially benefited from ANN, fields such as speech and image recognition widely use ANN in the current date, with Google’s search function being probably the most recognizable ANN model around. (IBM 2020)

3 Solution Development

New innovative solutions need to be developed as the competition between companies increases, technology develops, and everyone is looking to get an edge over the others. This is also true in the taxi world. Planning where the driver should head next for the best chance to receive a customer has depended largely on the intuition of the drivers and individual data points such as news on local events. But this decision-making process can be made more fact-based with availability to data, analytics, and developed forecasting models.

As the taxis goes through an area border this generates data of the individual event. This data includes many columns of information, the majority of which was not needed for the purpose of this project. The data that was used for this project includes the location, time and identification for each event that is logged when a taxi enters or exits a location. This data creates a timeline of individual events to which different timeseries methods can be performed. Since the number of logged users was relatively low at the time of the project, the amount of data was also relatively low and did not include historical data. This could result in forecasting models that perform poorly in this project but later when more data is captured, can prove to perform with higher accuracy.

The data used resides in a database from which it was fetched, and cleaned to find duplicate data, missing values and reformat data types into required types. After this the cleaned data was ran through several different forecasting models to find out the best performing model. For the baseline several simple methods were used; mean, naïve, seasonal naïve, and drift. After finding the best model to forecast the data with, the model would need to be analyzed to find optimum amount of data used for the forecast, and how far observations could be forecast without losing significant amount of accuracy.

The development utilized both Python and R, and multiple different criteria were considered when choosing which programming language would best fit this thesis. The software that would be developed after this thesis project will utilize

Python, but the workflow with R is more flexible and faster for analyzing and developing the solution. Downloading the data from the database with Python is simple to separate from the complete software, and it will not have changes to the code after this thesis is completed. Due to this the code for the data download was made using Python from the start. R was utilized for the analysis of the different forecasting models due to the flexibility and fast workflow that R allows. The application was broken into multiple separate parts. The main .R file called other .R files that were broken down according to their usage, such as data importing, aggregation, and forecasting models.

3.1 Data Cleaning

Raw data was first imported from the customer’s database. For the development purposes the data was imported once using python and stored as .csv file during the development. This imported data was then used for the model building and validation of the model by creating the comparison of which model performs the best. This model can then be used for the future development of the program. Listing 1 shows the code used to fetch the data from the database with the information using generalized names for security reasons.

```
import mysql.connector
import simplejson as json
import pandas as pd
from sklearn.model_selection import train_test_split

def fetch_from_db_export_to_json(sql_query):
    try:
        connection = mysql.connector.connect(host=HOSTADDRESS',
                                             database='DATABASE_NAME',
                                             user='USERNAME',
                                             password='PASSWORD',
                                             auth_plugin='NATIVE_PASSWORD')

        ##Execute the sql query to the connected database
        cursor=connection.cursor(buffered=True)
        cursor.execute(sql_query)

        ##Insert the query into a pandas dataframe
        table_rows = cursor.fetchall()
        df = pd.DataFrame(table_rows, columns= cursor.column_names)

        ##Close the connection to the database and return the result
        if connection.is_connected():
            connection.close()
            cursor.close()
            print("MySQL connection is closed")
        return df

    except mysql.connector.Error as e:
        print("Error reading data from MySQL table", e)

##Create the query to fetch the relevant data
sql_query_x = """SELECT event_id, event_date, event_taxi_id
                FROM tbl_event;"""

sql_query_y = """SELECT event_location_id
                FROM tbl_event;"""

result_x = fetch_from_db_export_to_json(sql_query_x)
result_y = fetch_from_db_export_to_json(sql_query_y)

X_train, X_test, y_train, y_test = train_test_split(result_x, result_y,
test_size=0.35, random_state=42)

##Export the data into a file, this is used for development, removed in
deployment
X_train.to_csv('taxi_events_x.csv', index=False)
y_train.to_csv('taxi_events_y.csv', index=False)

X_test.to_csv('taxi_events_x_test.csv', index=False)
y_test.to_csv('taxi_events_y_test.csv', index=False)
```

Listing 1. A Python subroutine that performs SQL query to fetch the development data and output is .csv files.

This .csv file, that was imported with the Listing 1 code, was then imported to R where the development and analytics was done. The data was split into two groups; train and test sets. The train set of the data was used to form the different forecasting models, while test set was used to analyze the performance of the different forecasting models. Listing 2 shows the single line of code required to import the data into R from the .csv file.

```
data <- read.csv("taxi_events_1.csv")
```

Listing 2. R code for importing a .csv file into a data frame.

With the simple line in Listing 2 the .csv file had been imported as data frame. Data frame is flexible and easy to use format for data analysis where the information is stored in two-dimensional matrix. After the data was imported it could be examined for possible complications that might affect data analysis. Listing 3 shows the code used to check for duplicate and missing values.

```
str(data)
check_dupes <- data %>% get_dupes(event_id)
check_na <- sum(is.na(data))
```

Listing 3. R code for examining the data.

The result from Listing 3 showed how there was virtually no cleaning required to do for the data. No duplicates or missing values were found from the data. The date format at this stage was text, which needed to be changed into a date format. But overall, the data was high quality and required minimal cleaning for the purposes of this thesis and can be aggregated in the next step.

3.2 Aggregating the Data

For the forecasting models, the raw data needed to be turned into a periodic time series. This time series was required to be in “ts” and “data.frame” data types for the forecasting models. The raw data was transformed into two separate time series; amount of demand at the location and the wait time at the location.

3.2.1 Aggregating Demand Data

The demand was calculated based on the number of cars going through a specified border within set period. Multiple methods for the demand calculation were considered and the specified location ID method was decided due to the real-world constraints that makes the number of taxis that would go through the border without a customer insignificant, and due to the simplicity of implementing the solution. A more robust method that also considers empty taxis going through a border can be developed, but this was decided to be outside the scope of the thesis. Listing 4 illustrates the code used to aggregate previously cleaned data into the demand time series.

```
location_demand <- function(data, period="1 hour", location){

      data[['event_date']] <- strptime(data[['event_date']], format="%FT%T")

      demand <- data %>% filter(event_location_id == location)
      demand$intervals <- cut(demand$event_date, breaks = period)
      demand <- demand %>% dplyr::count(intervals)
      demand$intervals <- strptime(as.character(demand$intervals),
format="%F%T")

      datats <- xts(demand$n, order.by = demand$intervals)
      output <- list(datats, demand)

      return(output)
}
```

Listing 4. R code for aggregating the raw data into the demand based on the input location ID.

Listing 4 turns the input .csv file of the data into the demand time series for the specified location ID, output being a list of the aggregated data in two forms; timeseries and data frame. The function required three inputs; data, period, and location. Data provided to the function was the data frame that was cleaned in the previous step, period specifying how long periods the time series should be divided into, and location index which details the specified border that was used for the calculation.

3.2.2 Aggregating Wait Time Data

The wait time calculation was based on the time stamps of the individual events. The events were grouped and sorted together by the users individualized identification number and arranged by time. Each event was then compared against the previous one and the difference in the time was saved to a new column. This provides the time it took the taxi to reach the border location from the previous border. The used method also provides the opportunity of detailing how long it took the taxi to enter back through the first border after going through an exit border. After getting the times between the borders, the data was grouped to keep track of the event series that is the series of borders a taxi goes through between entering and exiting the area. Listing 5 illustrates the code used to aggregate previously cleaned data into the wait time -time series.

```
wait_time <- function(data,
                      period="1 hour",
                      location_id_in,
                      location_id_out,
                      wait_location_id){

      starts <- location_id_in
      exits <- location_id_out
      wait_location_id <- wait_location_id

      #Changing the format of the time column
      data[['event_date']] <- strptime(data[['event_date']], format="%FT%T")

      #Calculating the wait times for the taxi_id's
      data <- data %>%
      filter(event_taxi_id != 9412) %>%
      arrange(event_taxi_id) %>%
      group_by(event_taxi_id) %>%
      mutate(group_index = row_number()) %>%
      mutate(wait = event_date - lag(event_date, 1))

      #Group events together according to the starting location_id
      group_index <- 0
      #data <- ungroup(data)
      data$wait <- as.numeric(data$wait)
      data[is.na(data)] = 0

      for(i in index(data)){
            if(data$event_location_id[i] %in% starts){
                  group_index <- group_index + 1
            }
            data$group_index[i] <- group_index
      }

#Group the table according to the group_index and create aggregated summary of
the event, assumes that the last wait event is also the
      output <- data %>%
      arrange(group_index) %>%
      group_by(group_index) %>%
      summarise(start_taxi_id = first(event_taxi_id),
            end_taxi_id = last(event_taxi_id),
            total_time = (sum(wait) - first(wait)),
            last_time = first(wait),
            wait = last(wait),
            start = first(event_date),
            exit = last(event_date)) %>% ungroup()

      output <- output %>% filter(total_time != 0)
      return(output)
}
```

Listing 5. R code for aggregating the raw data into the wait times and grouped series based on the entrance, exit, and wait location numbers.

The code in Listing 5 inputs the raw data from the .csv file and provides the output as data.frame with the calculated wait times as well as other high importance information. This added information could be used for further analytics and troubleshooting if any bugs arose during the model development, but this additional information would become obsolete once the code would be developed into forecasting tool as it would not be used in the forecast itself.

Listing 5 first turns the date information that was stored in string form into date form and groups the data together based on taxi ID's. These two steps allowed arranging the events in chronological order, and the calculation of the wait times between each event. Due to the way this was calculated, getting the difference between the two event dates, this solution also returns the time it took for the taxi to reach an entrance border after leaving the area through an exit border. This was simply excluded from the total time calculations but could be used for analysis if this information becomes of interest in the future.

From the data it can be seen that it currently includes small number of observations which are relatively similar to each other, and large gaps in time in between. This was due to the data being gathered from small number of users at the time, for a relatively short amount of time, which compared to the future intended use was only a fraction. This may affect the forecasting capability of the data and thus it might become necessary to redo the analytics after more data has been gathered to recalculate the best forecasting model. For example, with ARIMA forecasting the parameters for the model may change as more observations are added but ARIMA has still been able to be identified as the best model for this specific type of data.

3.3 Forecasting Models

Forecasting was carried out in multiple steps. This was divided between the two primary datasets that were used and further broken down for each of the forecasting models that were used. Both sets had multiple forecasting models used to find the most accurate one of them. Simple models were used to find a base line to which it was possible to compare the forecasting models to. These were mean, naïve, seasonal naïve, and drift model.

Mean, or average model, takes the average of all the historical values and forecasts this as all the future values. With naïve forecast all the future values are set to be same as the last observation in the series, where seasonal naïve forecast uses the value from the previous same season, as in same day from the previous week. Lastly, drift method which uses the naïve method as its basis but allows the forecast to drift based on the average change in the dataset. These simple forecasting methods are used as base lines due to their simplicity and ease of calculation. This ensures that whatever forecasting model is produced and used, is better at the forecasting than these simple methods.

3.3.1 Demand Forecasting

Demand was calculated by the number of cars going through a specified border within a given period of time. Due to the low number of events the data was divided into periods of one day. Before starting forecasting, it was necessary to look more into the data itself. Trend, seasonality, and autocorrelation was all investigated to obtain a better understanding of the data that was being used.

In Figure 6 the demand data could be observed more closely with its different characteristics. Seasonality, trend, and remainder could be seen along with the actual data. These plots were in a single column with the timestamp being shared between the four plots. The bar on the right side of each plot helped demonstrate relative size of the observations, thus allowed for easier comparison between the four different plots.

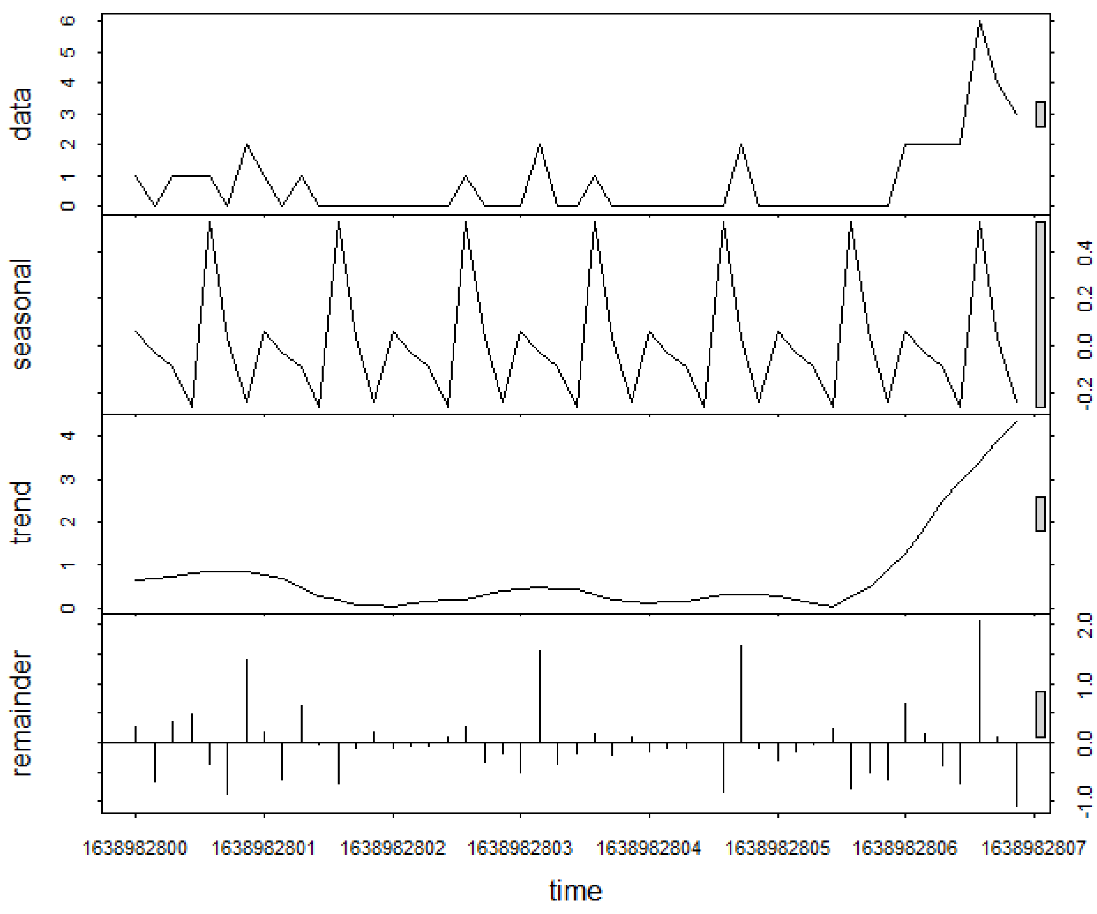


Figure 4. Seasonality, trend, and remainder of the demand data.

As seen in Figure 6, data could be first observed having a long period of little to no observations, with a much larger amount appearing right towards the end. Small amount of seasonality could be observed in the data, while a very clear and strong trend curve was observable in the data towards the end of the time series. The time series also included several large spikes in the remainder. From this it was clear that there has been a large shift in the data towards the end of the observation period. From these characteristics, trend and autocorrelation was looked more closely into, while seasonality in this dataset can be difficult to identify, especially if the seasonality occurred during a longer period. With this in mind, seasonality was still taken into consideration in these models as minor amount of seasonality was found in the data. In case this test will be replicated in the future to find better models with seasonality, the code would already accommodate that.

Figure 7 includes the ACF plot of the demand data where multiple lags were taken from the data, with each pillar in the plot representing one of the lags. The usual patterns found in the ACF plots are wave pattern, spikes, gradual decline, and exponential decline; latter two being mutually exclusive with the exponential decline indicating stationarity.

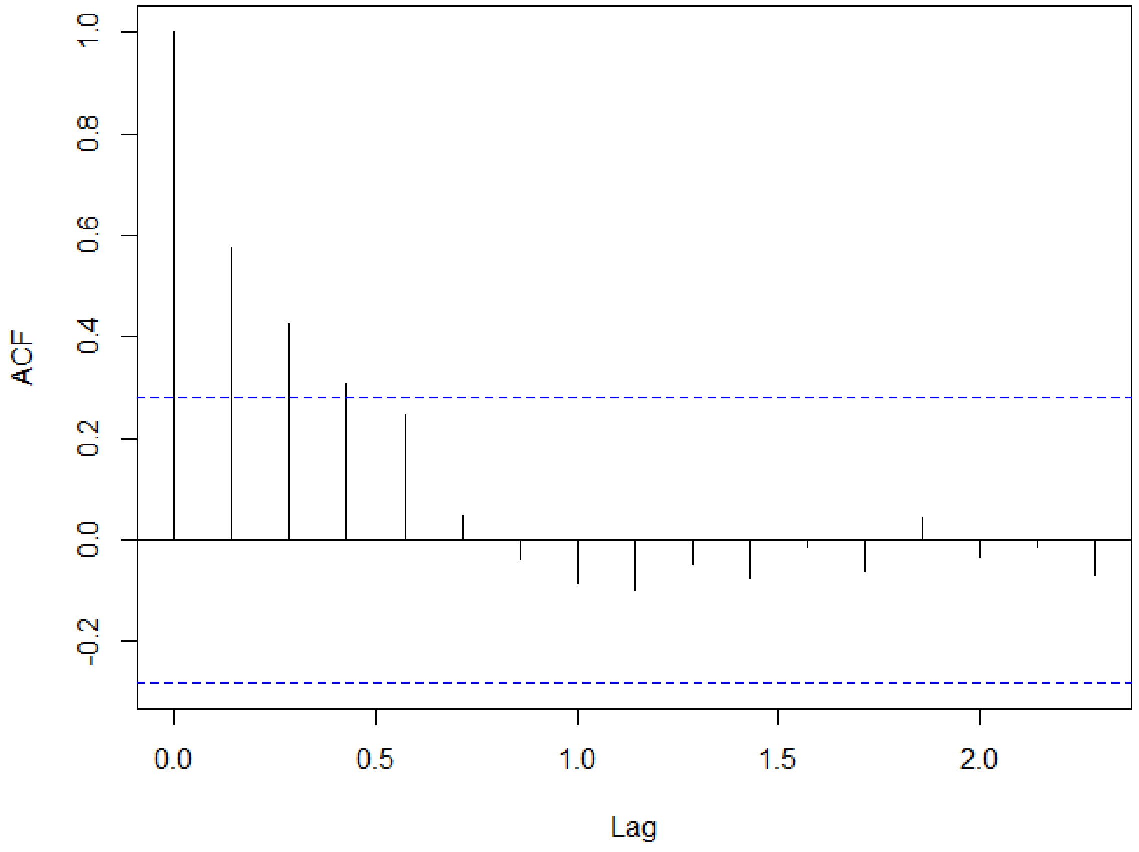


Figure 5. ACF plot of the demand data.

Figure 7 could be observed to be a relatively gradual decline, which comes from the trend that was observed before. The ACF plot also did not reveal any significant form of seasonality, which would be observed as a wave-like form in the plot. Figure 8 illustrates the same ACF plot as in Figure 7 with the same data, with the only difference being that the data was once differentiated to get rid of the correlations found when inspecting Figure 7.

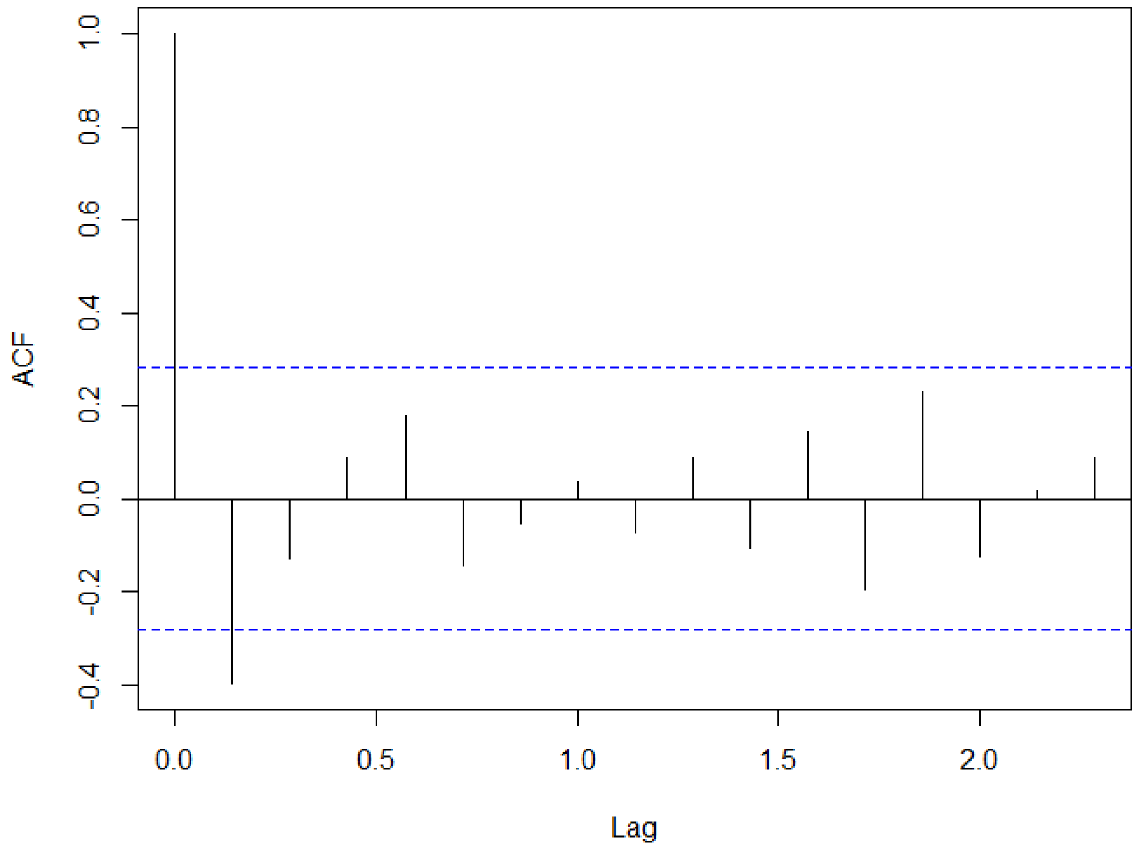


Figure 6. Once differentiated ACF plot of demand data.

As seen in Figure 8, the previously observed gradual decline has been replaced with an exponentially decaying one. Still one spike was over the significance line at the first lag. From Figure 8 the data could be determined to have been transformed into a stationary time series, as no further correlations between the observations seemed to exist. Figure 9 shows the partial ACF plot of the same demand data to confirm no other relations are found that are not visible with ACF.

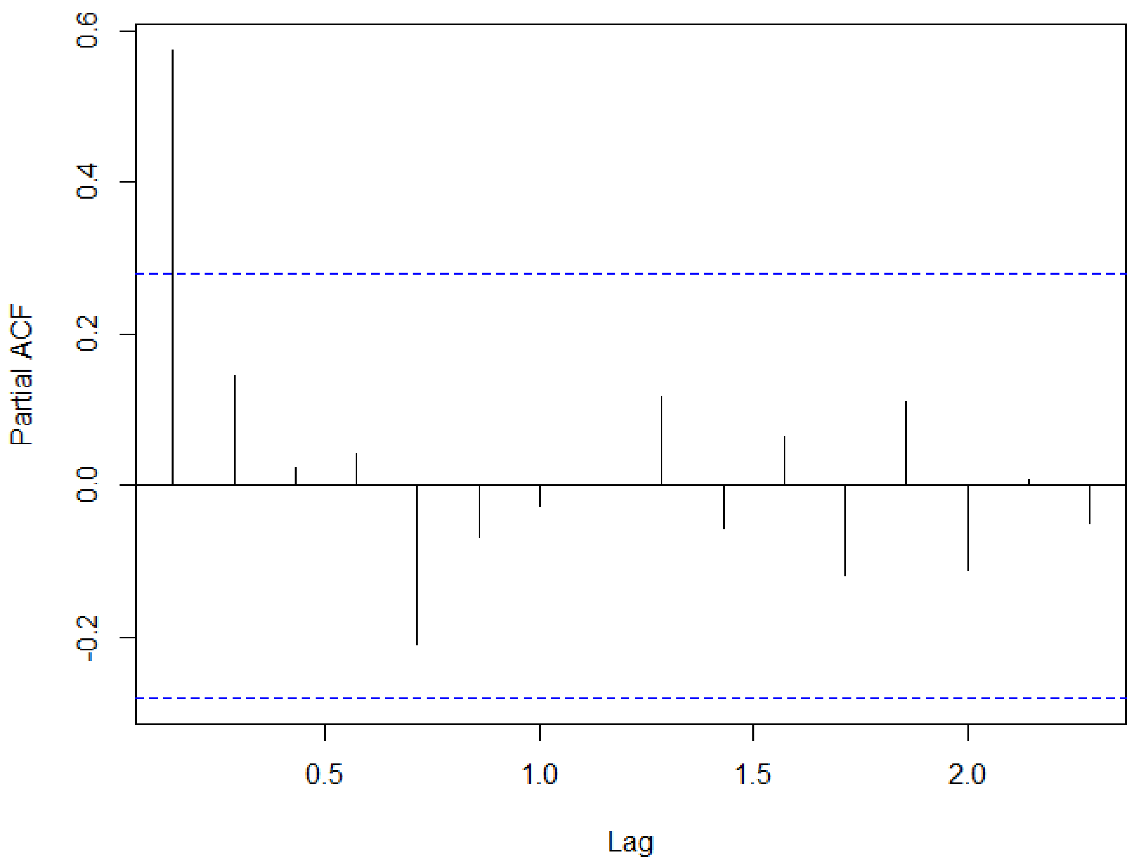


Figure 7. PACF plot of the demand data.

As seen in Figure 9, an exponential decline in the graph could be observed. From this, it could be assumed that there were no hidden relations between the observations that the ACF plot was not able to capture.

3.3.1.1 Baseline models

With a better understanding of the data, forecasting could be done. First the baseline models had been created to have an accuracy level to compare the other forecasting models to. Mean, naïve, seasonal naïve, and drift models were used for this purpose. Code used for this is shown in Listing 6, where a function was written to create each of the four baseline models which then stores the models in a list and proceeds to output this said list.

```
naive_model <- function(df, i){
     naive1 <- meanf(df$n, i)
     naive2 <- rwf(df$n, i)
     naive3 <- snaive(df$n, i)
     naive4 <- rwf(df$n, drift=TRUE, h=i)

     output <- list(naive1, naive2, naive3, naive4)
     return(output)
}
```

Listing 6. R-code for creating mean, naïve, seasonal naïve, and drift forecasting models.

In Listing 6 the created function input a data frame of the time series, and an integer value for how many periods into the future the model should forecast. Each of the models were created on their own line, coerced into a list, and the list was then provided as an output by the function. Figure 10 shows the output of this function, displaying a plot for each of the baseline models; mean, naïve, seasonal naïve, and drift model. Blue lines indicating the forecasted values with the dark grey and the light grey shaded areas in the plots referring to 80% and 95% prediction intervals, respectively.

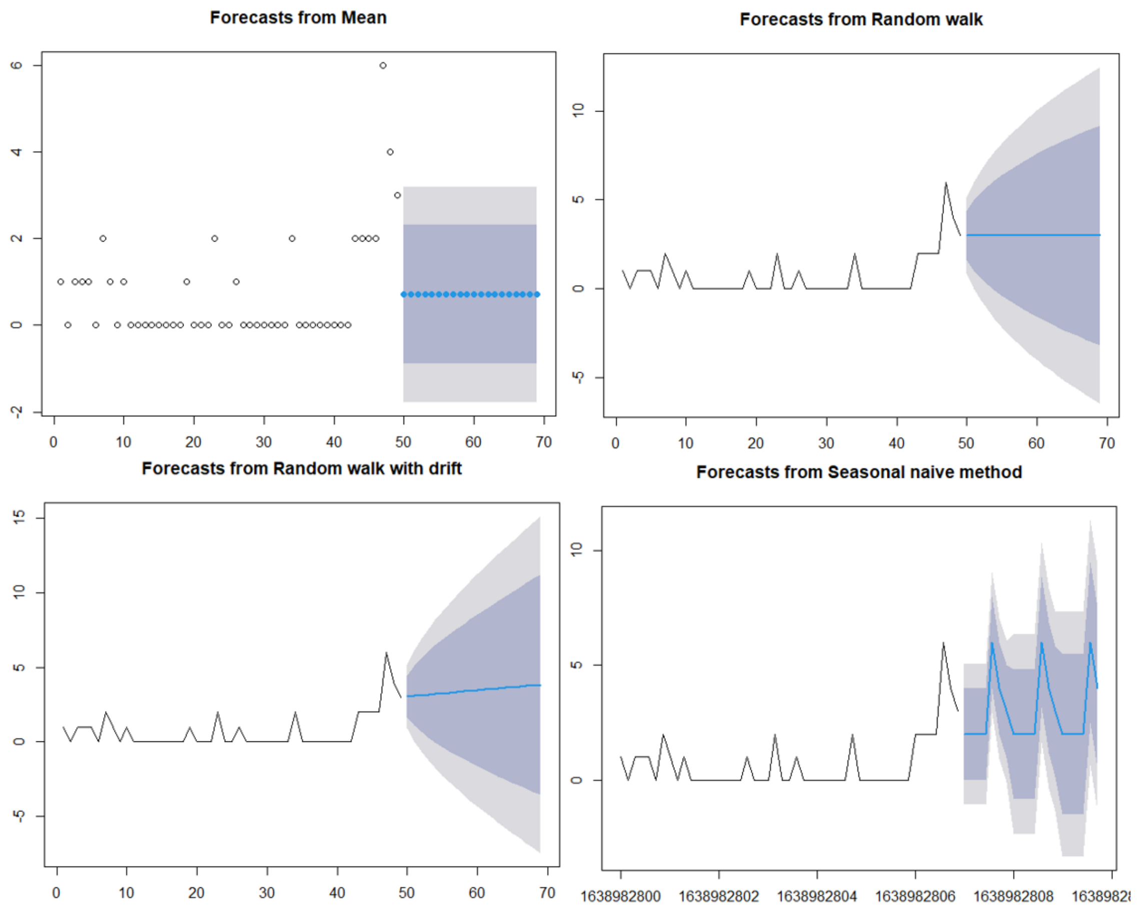


Figure 8. Plots for each of the baseline forecast methods.

Figure 10 showed each of the results providing mostly a straight line, with the seasonal naïve method being notable outlier with the clear wave like pattern. This pattern came directly from the seasonality that the seasonal naïve method was able to detect from the data. After a longer collection time for the time series data, it could be possible detect more accurate seasonality.

3.3.1.2 Linear regression model

The first of the potential forecasting candidates was linear model. This model attempts to fit a linear line in a way that minimizes the discrepancy. The exact method used is called simple linear regression, as the time series that was being used contains only two variables; time variable and the value that was forecasted, which in this case was the demand. If there were multiple variables that were used for the forecasting, this would be called multiple linear regression. Many potential candidates exist for what could improve the forecast accuracy in the multiple linear regression depending on what is being forecasted. For taxi demand this could for example be gross domestic product, weather forecast, schedule for nearby large events, or ongoing holidays. But this is time consuming task that requires more advanced knowledge into the environment the taxis operate in and can prove additional difficulties as where this additional data would be sourced from. This can be included in further development in the future as more accurate forecasting model. Listing 7 shows the code used to create the function that calculates the linear regression model and produces the forecasted values from this model.

```
linear_model <- function(df, i){
      fit <- lm(n ~ intervals, data = df)
      fcast <- forecast(fit, future, h=i)
      return(fcast)
}
```

Listing 7. R-code for calculating the linear model forecast.

Listing 7 inputs the time series data and an integer value for how many periods into the future the model should forecast. This was then fed into the lm() -function, which calculated the linear model. Finally, this model was fed to the forecast() -function and output was provided for the generated forecast. From this output, a plot could be formed. Figure 11 visualizes the plot received from the forecasted linear regression model.

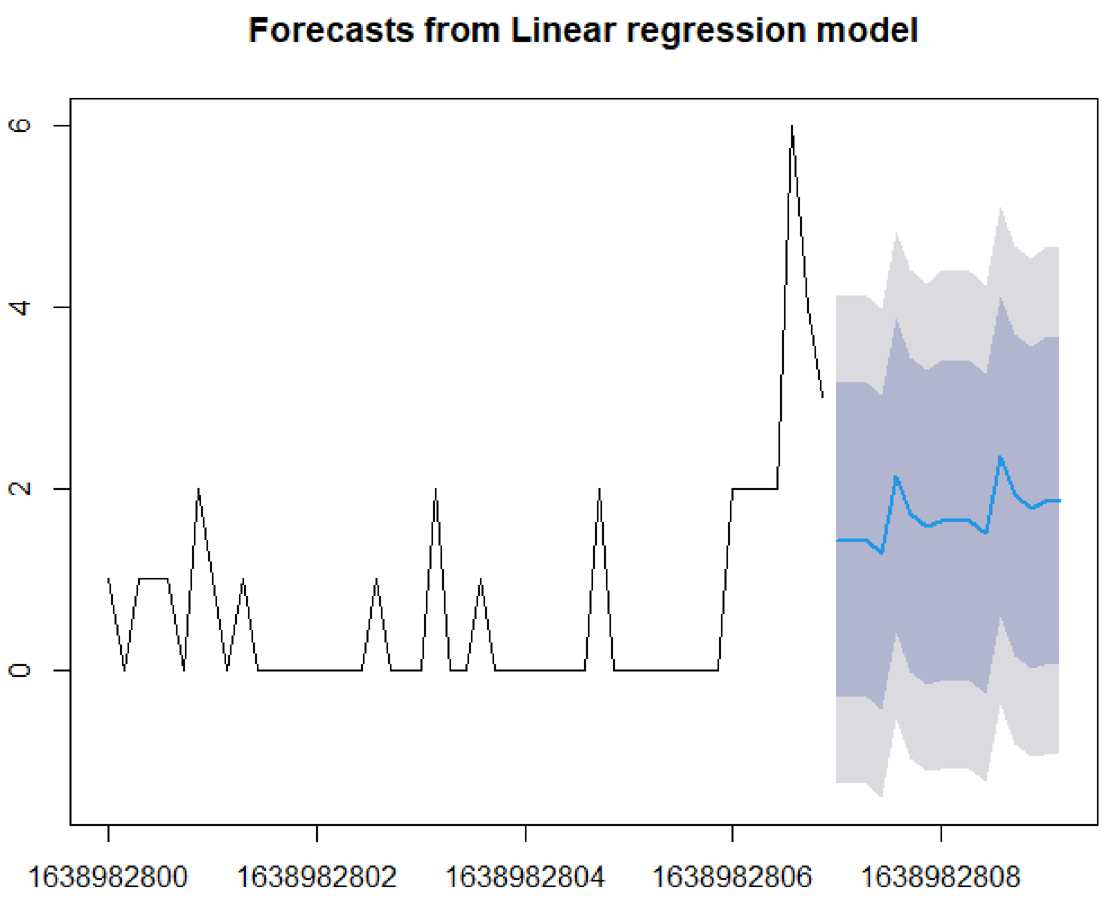


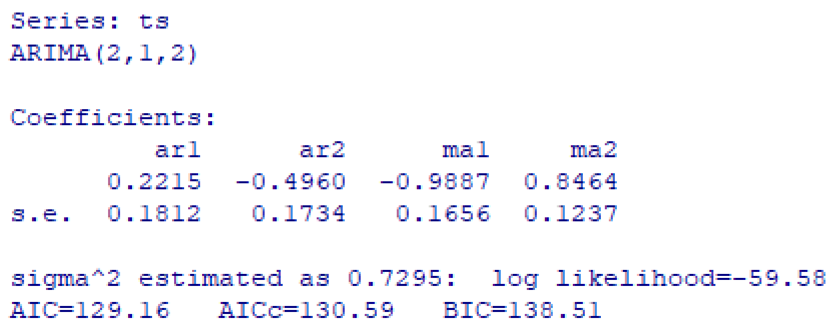
Figure 9. Forecast plot using linear regression model.

In the plot visualized on Figure 11, it could be seen how different off a result was received from the baseline models. Unlike most of the baseline models, the forecast plot was not completely straight, and the observed seasonality was not as drastic as the seasonal naïve model was.

3.3.1.3 ARIMA model

Manual ARIMA model was used next. This required looking back at the information that was received from the ACF and PACF plots, as well as differentiation that was conducted due to the present trend in the time series data. ARIMA model requires three input values besides the data itself; order of the autoregressive part, degree of first differencing involved, and order of the moving average part.

It was already determined that one degree of differentiation was enough to take the trend into account, and this leaves the AR and MA order values left. For this the ACF and PACF plots from Figure 8 and Figure 9 came useful. Looking at the ACF plot in Figure 8, it could be seen how only the second lag has a significant spike over the threshold, with the rest of the plot following a sinusoidal pattern with all the values well under the significance threshold. While looking at the PACF plot in Figure 9, it could be seen how none of the observations were above the significance threshold but included several larger than normal spikes at multiple different lags. From this the first list of ARIMA plots could be created that were tested for greatest accuracy; (2, 1, 2), (2, 0, 2), (1, 1, 2), (2, 1, 1). Figure 12 shows an example output for an ARIMA model which provides information regarding the generated model.



```
Series: ts
ARIMA(2,1,2)

Coefficients:
         ar1      ar2      ma1     ma2
      0.2215  -0.4960  -0.9887  0.8464
s.e.  0.1812   0.1734   0.1656  0.1237

sigma^2 estimated as 0.7295:  log likelihood=-59.58
AIC=129.16   AICc=130.59   BIC=138.51
```

Figure 10. Example of an output from creating an ARIMA model using R.

Figure 12 showcases input and output for a line of code that was used to create the ARIMA models. The output provided a large set of information regarding the created model. For this information the focus will be on the Akaike information criterion (AIC) value. AIC provides a prediction of the forecasting error in a model, but this is unable to provide an absolute answer on the quality of the model. This means that AIC and AICc can be used to compare models using the same data but cannot be used to receive the actual accuracy of the models. AICc was created due to concerns for overfitting in AIC and AICc is AIC with a correction done for smaller datasets, providing higher accuracy. Due to this AICc specifically was used to compare the different ARIMA models. Lower AICc score is better. AICc scores for the different demand data ARIMA models were looked at in Table

2 which shows the model type on the left column and the received score on the right column. (Cavanaugh 1997)

Table 2. AICc score of the generated ARIMA models.

ARIMA model	AICc score
(2, 1, 2)	130.59
(2, 0, 2)	146.47
(1, 1, 2)	133.8
(2, 1, 1)	137.04

From Table 2 the different generated ARIMA models could be compared between each other. Based on these results the model ARIMA(2, 1, 2) performed the best with the AICc score of 130.59, while ARIMA(2, 0, 2) performed the worst with the AICc score of 146.47. With these results, residuals was checked next from the ARIMA(2, 1, 2) model. Figure 13 showcases these residuals produced from the ARIMA(2, 1, 2) model and two plots analyzing the residuals; ACF and histogram plots to confirm stationarity.

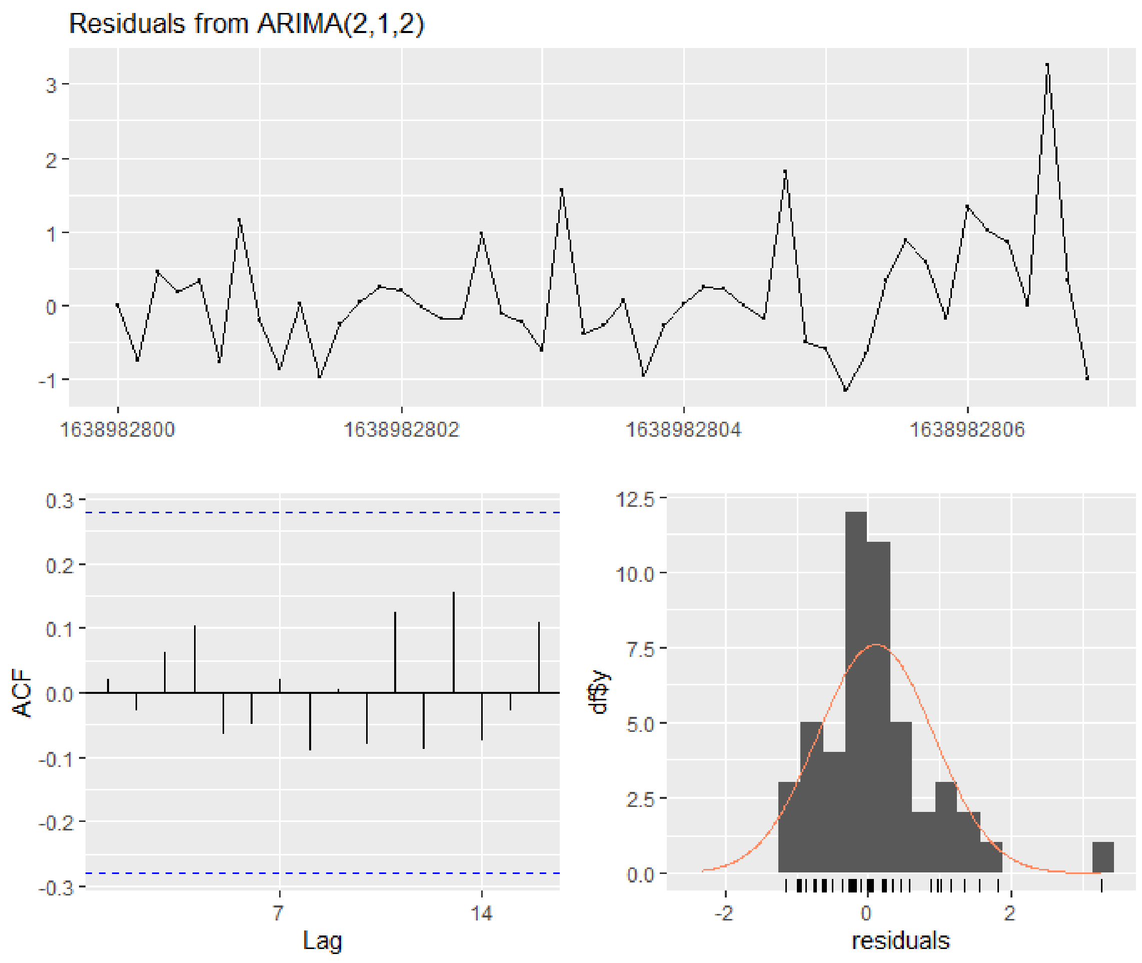


Figure 11. Residuals from the ARIMA(2, 1, 2) model.

With Figure 13 residuals were checked after creating the model to see if they resembled white noise, making sure that no unaccounted-for characteristics were left in the data after running the model. The plot of the residuals did not indicate any clear characteristics, ACF plot had all the lags well under the significance limit, and the histogram plot of the residuals resembled normal distribution. As such it was assumed that the residuals did resemble white noise. With this confirmation, next forecasting could be done for the ARIMA model. Figure 14 shows plot of the forecast received from the ARIMA model.

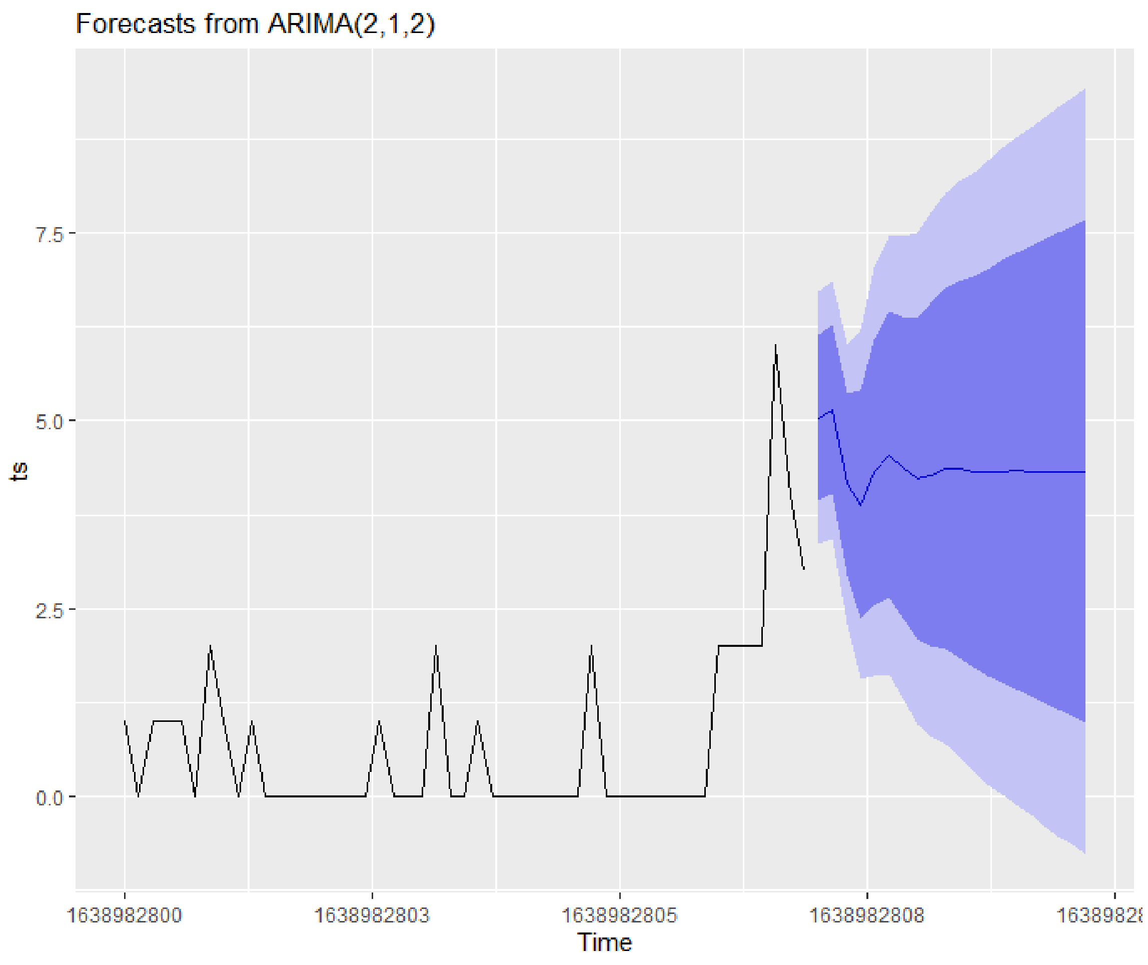


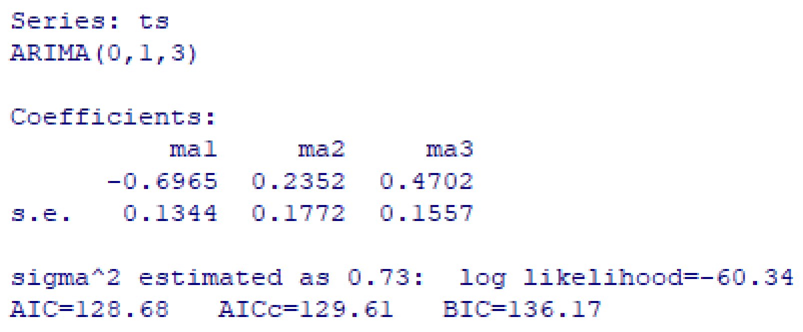
Figure 12. Forecast plot of the ARIMA(2, 1, 2) model.

Figure 14 showed another unique forecasting plot from the previously created forecasting plots. The forecast was more sporadic towards the start of the forecast, evening out more to a straight line quickly towards the end of the forecasted values.

3.3.1.4 Automatic ARIMA model

Automatic ARIMA model is an existing function that attempts to automatically find the best fitting ARIMA model to a dataset, without having the user manually input the model parameters; order of the autoregressive part, degree of first differencing involved, and order of the moving average part. The way that the automatic ARIMA function works is very similar to the workflow that would be followed to find the best fitting ARIMA model manually. First the function determines the number of differences according to Kwiatkowski–Phillips–

Schmidt–Shin tests, which provides a value between zero and two for the model to use. The second step is for the function to minimize AICc score by altering the values that are provided for the order of the autoregressive and moving average parts. The function provides multiple varying possibilities for each value, from which the lower AICc scores are determined until no further reduction can be seen in the model. The function does not have to calculate unnecessary models since the function only considers the autoregressive and moving average orders from neighboring values. This leads to possible cases where the automatic function is not able to find the best possible model, which is why manual analytics should not be discarded. Figure 15 shows the output from the model that the automatic ARIMA function created from the demand data.



```
Series: ts
ARIMA(0,1,3)

Coefficients:
          ma1     ma2     ma3
      -0.6965  0.2352  0.4702
s.e.   0.1344  0.1772  0.1557

sigma^2 estimated as 0.73:  log likelihood=-60.34
AIC=128.68   AICc=129.61   BIC=136.17
```

Figure 13. Output from the automatic ARIMA model.

From the output in Figure 15 it could be observed that the function came to the best conclusion, which was ARIMA(0, 1, 3) model with the AICc score of 129.61. Comparing this value to what was manually created, the manually created ARIMA(2, 1, 2) model performed marginally worse with the AICc score of 130.59. Due to this the model created by the automatic ARIMA function could be included and continued with into the accuracy comparison.

3.3.2 Wait Time Forecasting

Wait time, or lead time, was calculated by how long it has taken for an individual taxi to exit the location area after entering it. This gave the length of time that a taxi spent inside the area before leaving. Although the time spent outside the pick-up area could be calculated from the captured observations, it was not utilized within the scope of this project. Due to the nature of the data and the purpose for why the wait time was being forecasted, it was chosen to split it into equal, hour-long segments. In cases where multiple observations were recorded within an hour, the average of the wait time observations was taken, which was then shown as the observation point for the given hour. Before starting forecasting, a closer look into the data was taken similarly as to the demand data. Trend, seasonality, and autocorrelation was all investigated to obtain a better understanding of the data that was being used. Figure 17 allows for closer observation of the different characteristics in the wait time data. As with the demand data seasonality, trend, and remainder was analyzed.

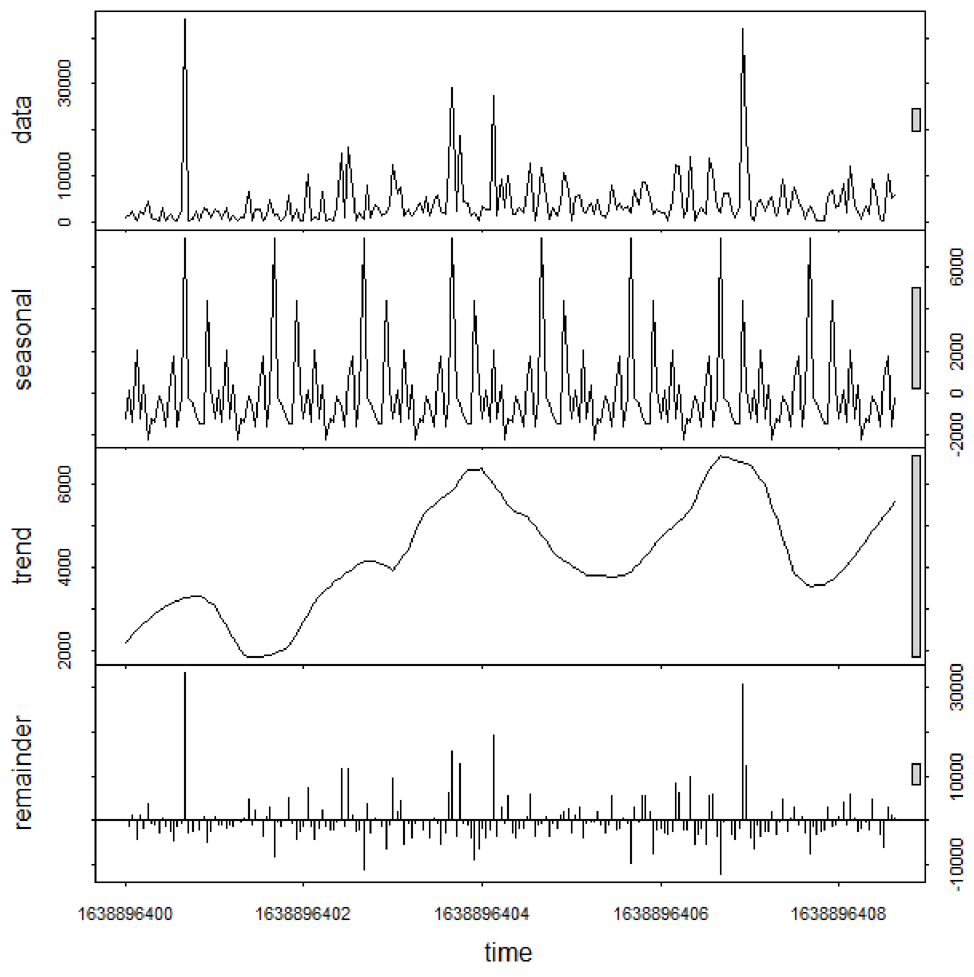


Figure 14. Seasonality, trend, and remainder of the wait time data.

As seen in Figure 17, seasonality showed regular spikes, while there was no clear trend. The calculated trend line having relatively low but volatile variance could be difficult to forecast for, if longer period forecasts are wanted in the future. Due to the nature of the data, there was more observations to be considered in the forecast, and with the high variance in the observations, it could be difficult to receive accurate forecasts. From these graphs the data was assumed to have certain amount of seasonality to it that will be seen in the provided forecasts. With Figure 18 seasonality and trend could be further analyzed with the help of ACF plot.

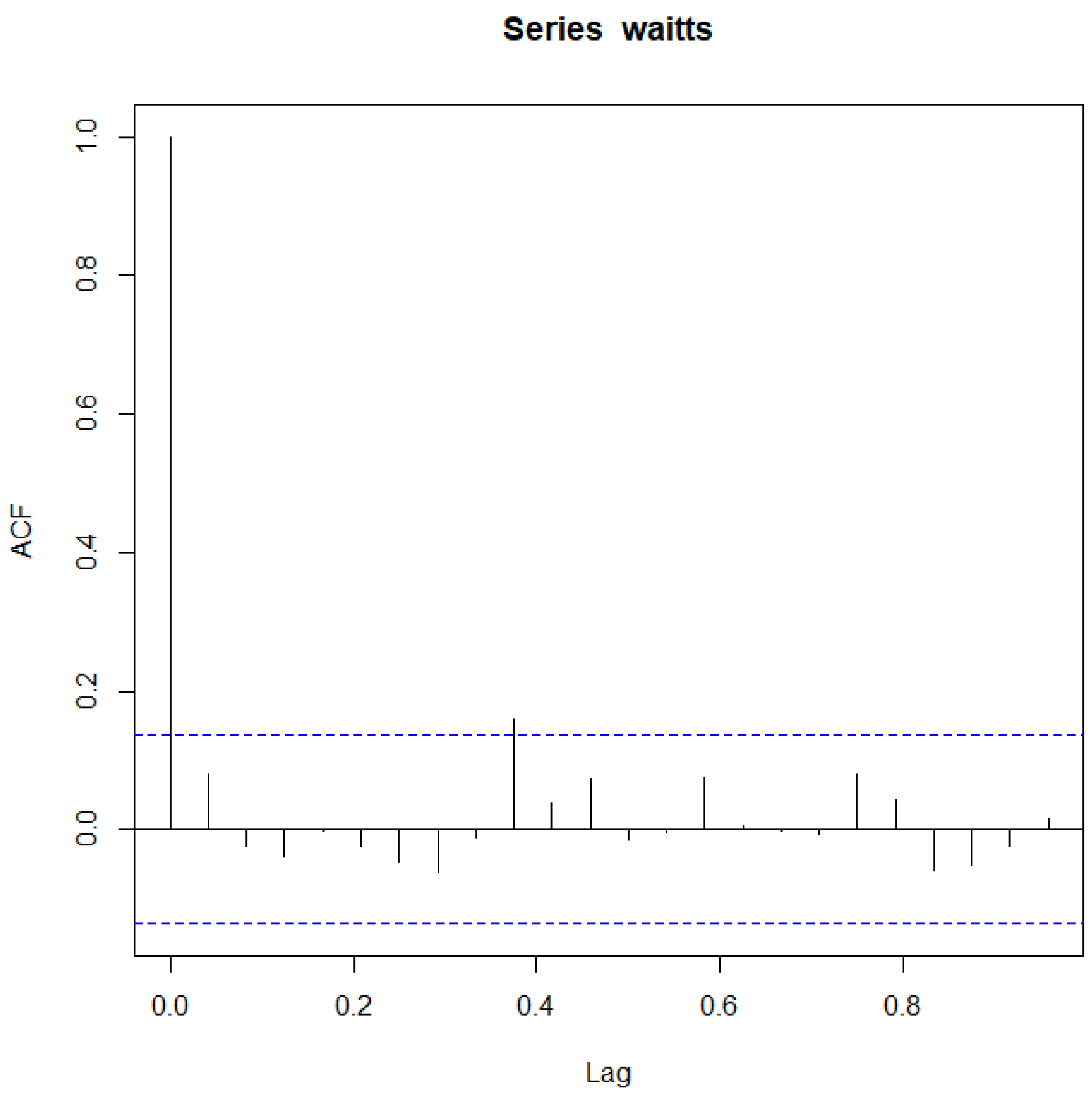


Figure 15. ACF plot of the wait time data.

From Figure 18 it was difficult to notice any seasonality, which should show in the graph as a wave like pattern, similarly to there not having been any trend. From the plot one spike at lag 10 went over the significance limit, which was required when a manual ARIMA forecasting model is created in the following section. Figure 19 goes further into analyzing the wait time data to confirm that the data was stationarity by analyzing the PACF plot.

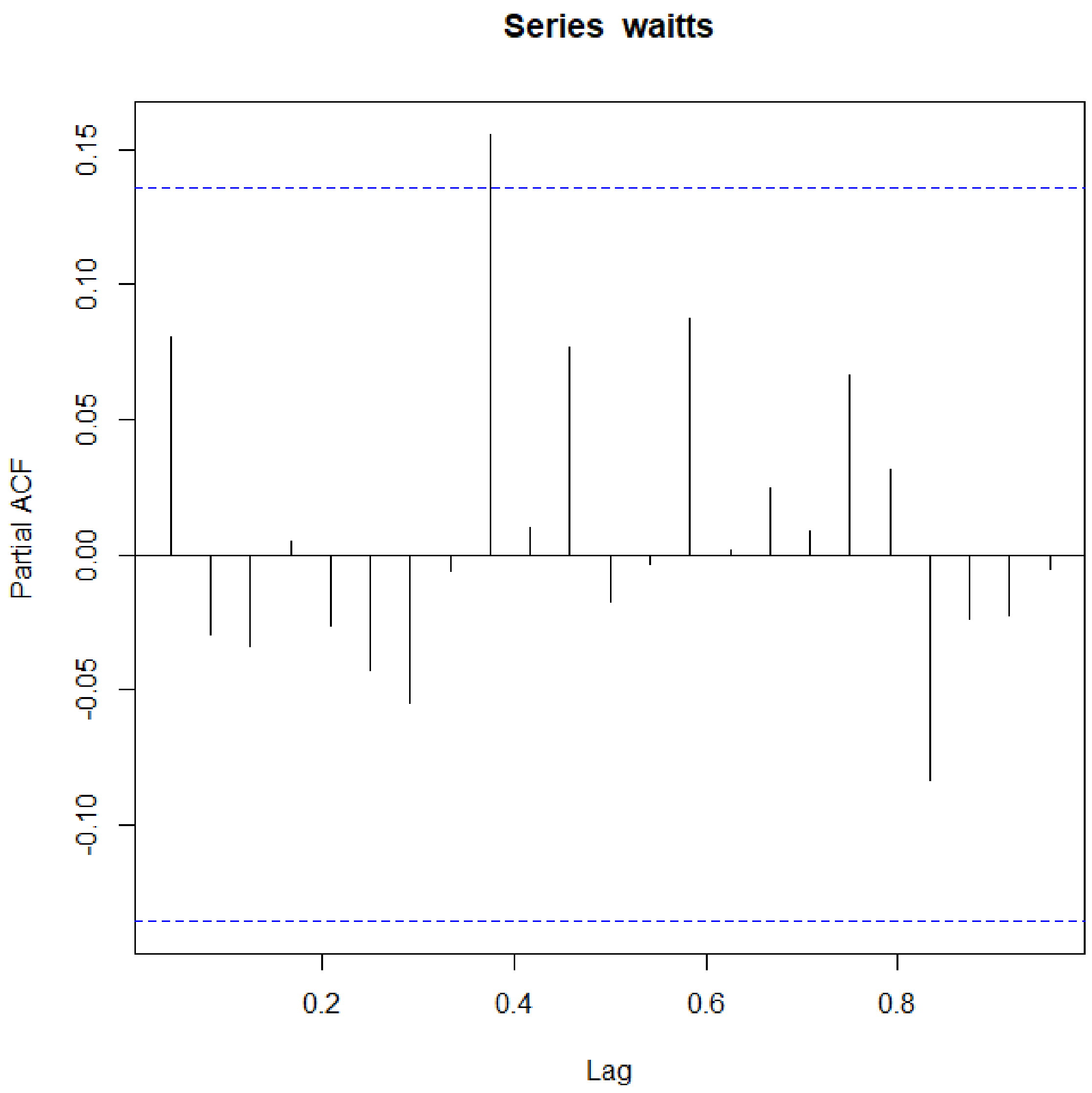


Figure 16. PACF plot of the wait time data.

As seen in Figure 19 plot, one significant spike was visible at lag 9 with no other visible characteristics to the data. This will be required when creating a manual

ARIMA forecasting model in a later section. With this analysis done, forecasting could be done for the different models that require specific parameters.

3.3.2.1 Baseline models

From the initial analysis of the time series data, the forecasting models could be calculated in the same way forecasting was carried out for the demand data. The analysis into the data showed the wait time data to be more volatile than that of the demand data. This could lead to more inaccurate forecasting models overall, even though the comparison between the different forecasting models could still be done. Figure 20 shows the generated baseline forecasting models from the wait time data.

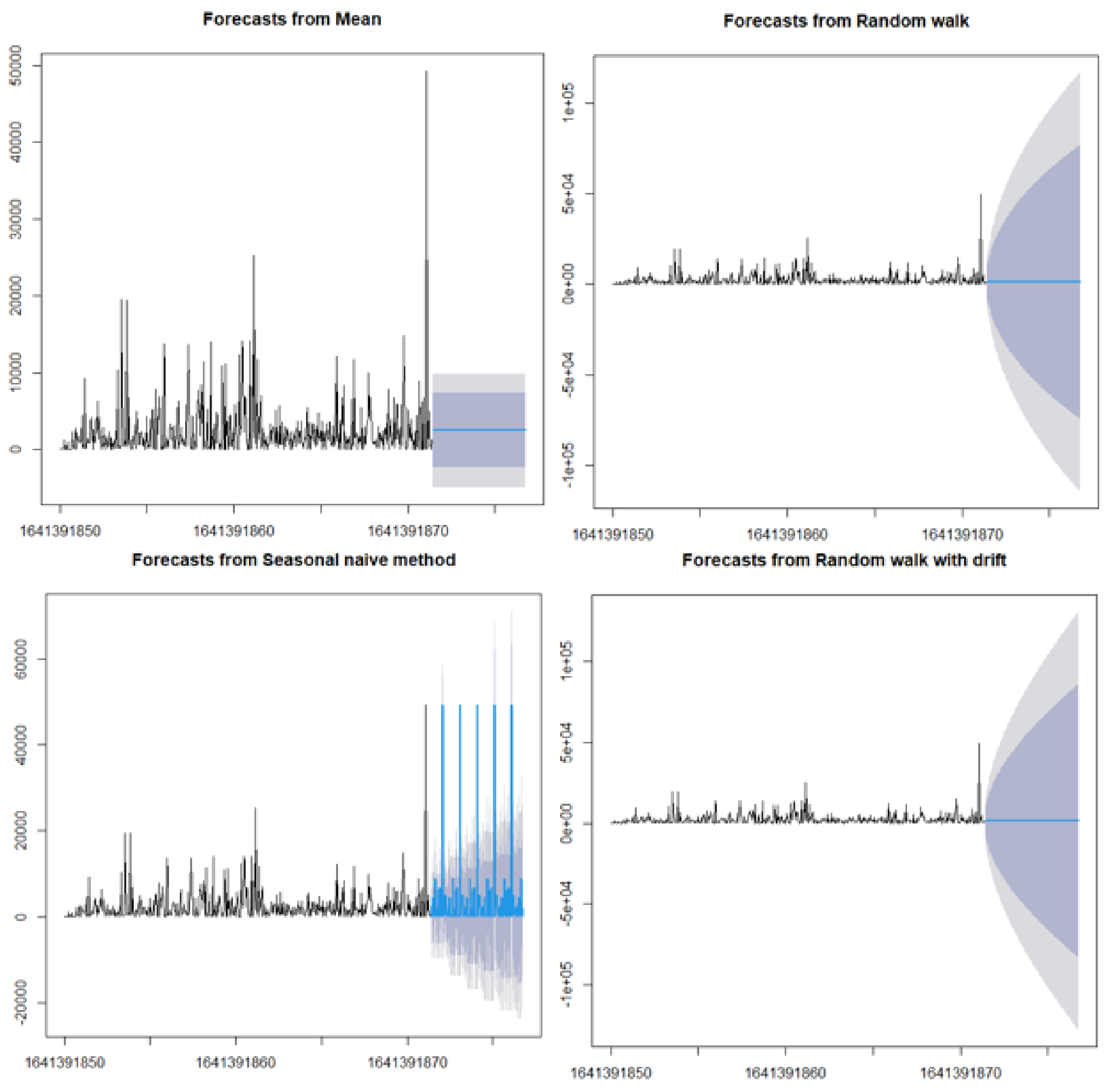


Figure 17. Plots for each of the baseline forecast methods.

Just as with the demand data forecasts, the baseline models shown in the Figure 20 for the wait time data were mostly straight lines, save for the seasonal naïve model. The dark grey and the light grey shaded areas in the plots refer to 80% and 95% prediction intervals, respectively. The naïve and drift models seemed especially inaccurate, even with just visual observation. The seasonal naïve plot showed to continually repeat one of the large spikes in the observation value towards the end of the data period, which did likely result in high amount of error during the accuracy analysis.

3.3.2.2 Forecasting models

Linear regression, ARIMA, and automatic ARIMA models were used as the actual forecasting models for the wait time data as with the demand data. Figure 21 shows the forecast plot for the linear regression model. This model was created using the same method as in demand data and with the same code; simple linear regression was used using time variable and observation values with no multiple variables.

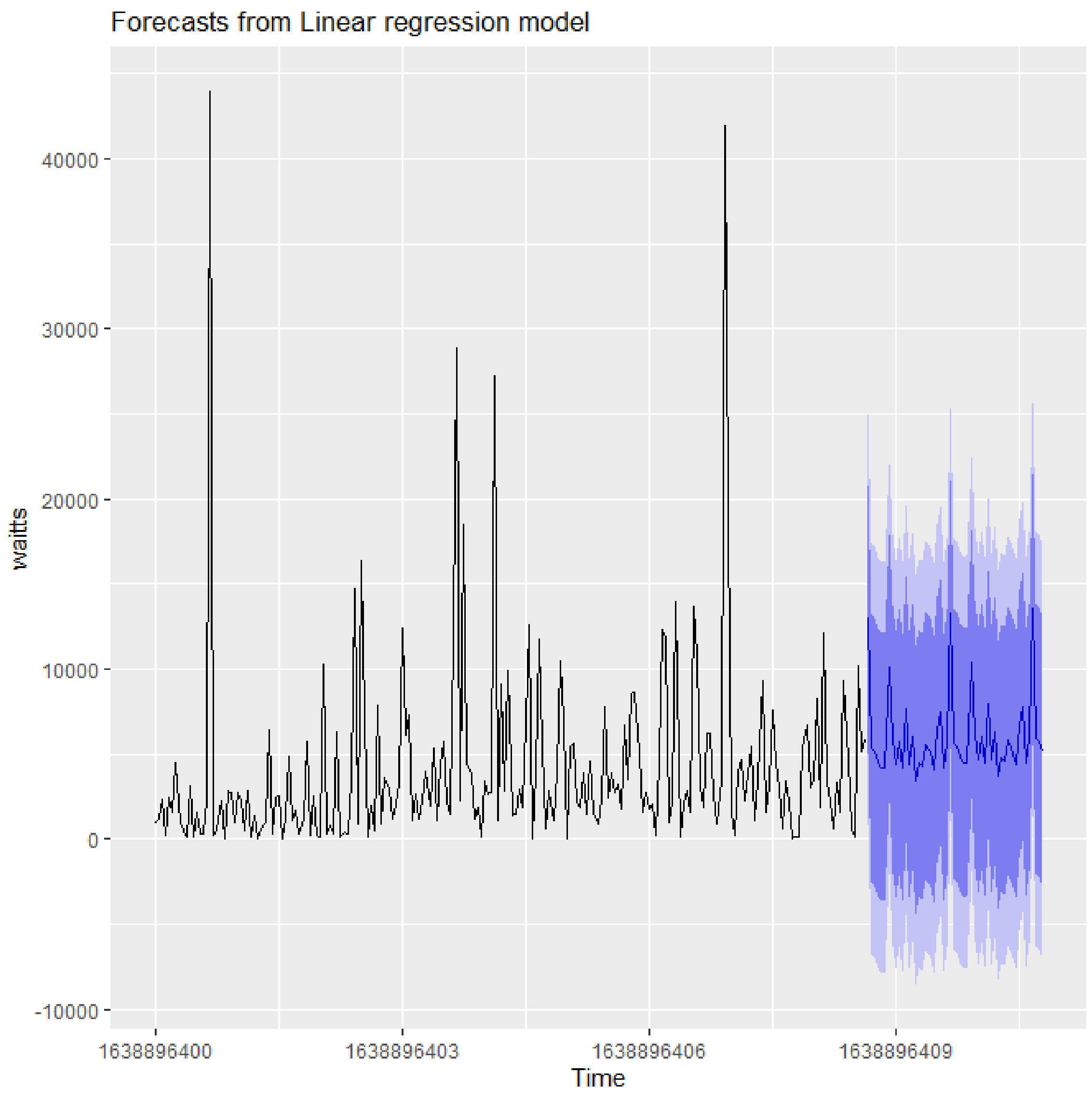


Figure 18. Forecast plot using linear regression model.

Figure 21 showed much higher amount of variance that could be observed in the forecast. The prediction intervals in the plot were closer to the average observations, instead of the high spikes that were observed with the seasonal naïve model. Due to this it was likely that the linear regression model proves to be more accurate than this base model.

Next a manual ARIMA model was calculated using the earlier ACF and PACF plot results that were observed in the earlier section. These were used for the three inputs on the ARIMA model; order of the autoregressive part, degree of first differencing involved, and order of the moving average part. With no degrees of differentiation apparent in the ACF plot, and the AR and MA parts from the first analysis being nine and ten respectively. With these points in mind the initial ARIMA model that was created is ARIMA(9, 0, 10). From this initial model multiple variations could be created using neighboring values and compared the received AICc values to determine the best model. This was continued multiple times to find the best possible fit, creating multiple batches of models to test out. Table 3 shows the AICc scores for all the different ARIMA models that were created for the wait time data to compare. Left column shows the specific ARIMA model and right column shows the AICc score received for the model.

Table 3. AICc score of the generated ARIMA models.

ARIMA model	AICc score
(9,0,10)	4223.26
(9,2,10)	4205.5
(9,0,11)	4226.23
(9,0,9)	4220.89
(8,0,10)	4225.56
(10,0,10)	4227.13
(8,2,9)	4199.81
(8,3,9)	4203.39
(7,2,9)	4198.16
(8,2,8)	4200.42

(6,2,9)	4195.04
(5,2,9)	4194.46
(4,2,9)	4200.77

As seen in Table 3, over dozen different models were run before no more reduction to the AICc score was observed in the models, which indicated reaching most accurate model. Range in the received AICc scores was relatively low, between 4227 and 4194, which indicated minimal overall improvement in accuracy. From these models the lowest AICc score came from the ARIMA(5, 2, 9) model. Figure 22 shows the forecast which was formed from this model.

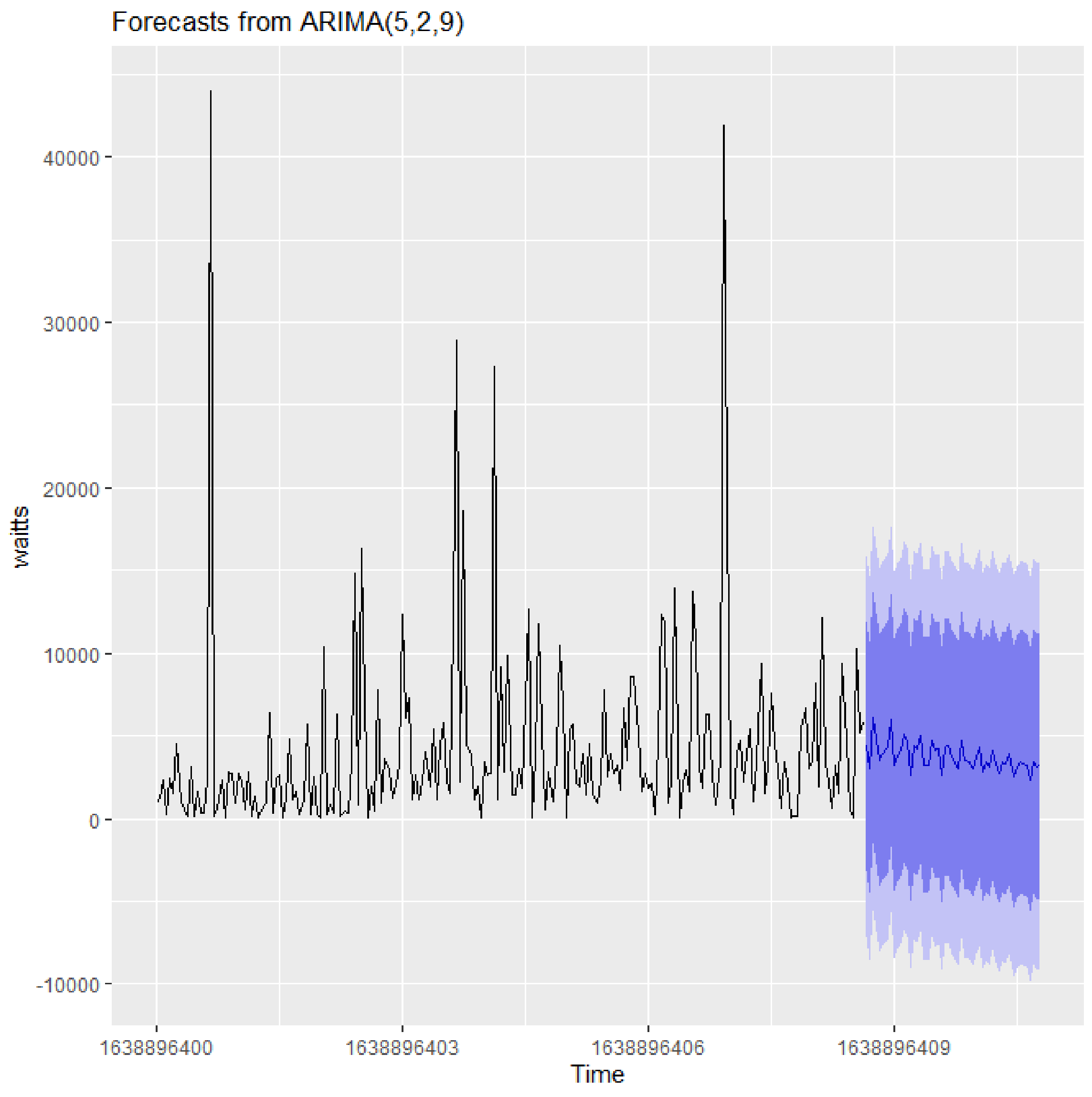


Figure 19. Forecast plot of the ARIMA(5, 2, 9) model.

Figure 22 showcases the forecast plot for the ARIMA(5, 2, 9) model. This model included sporadic but even variation in the observation values with a gradual drift downwards and smoothing towards the end of the forecast period. Figure 23 shows the analysis for the residuals that were received from the forecasted ARIMA model.

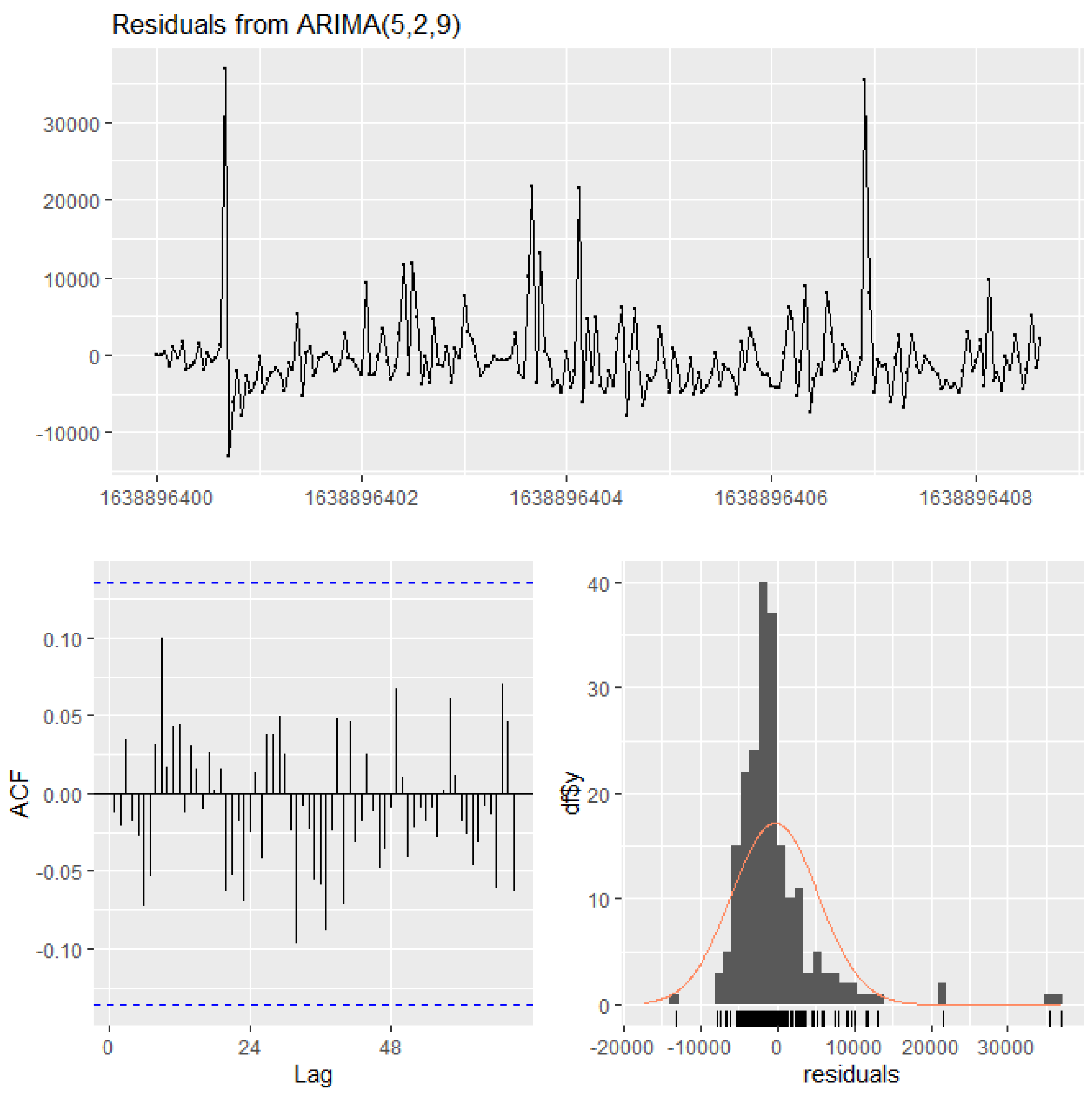


Figure 20. Residuals from the ARIMA(5, 2, 9) model.

With Figure 23 the residuals for the ARIMA(5, 2, 9) model was observed and analyzed. The shown residuals resembled white noise with no apparent characteristics to them. The same was observed from the created ACF plot, which had all the lags under the significance limit, resembling sinusoidal pattern. As such it was assumed that no more could be done to improve this ARIMA model.

After the manual process, automatic ARIMA function was run for the wait time data. Running the automatic ARIMA model wait time data produced ARIMA(0, 0, 0) model, which is shown in the Figure 24.

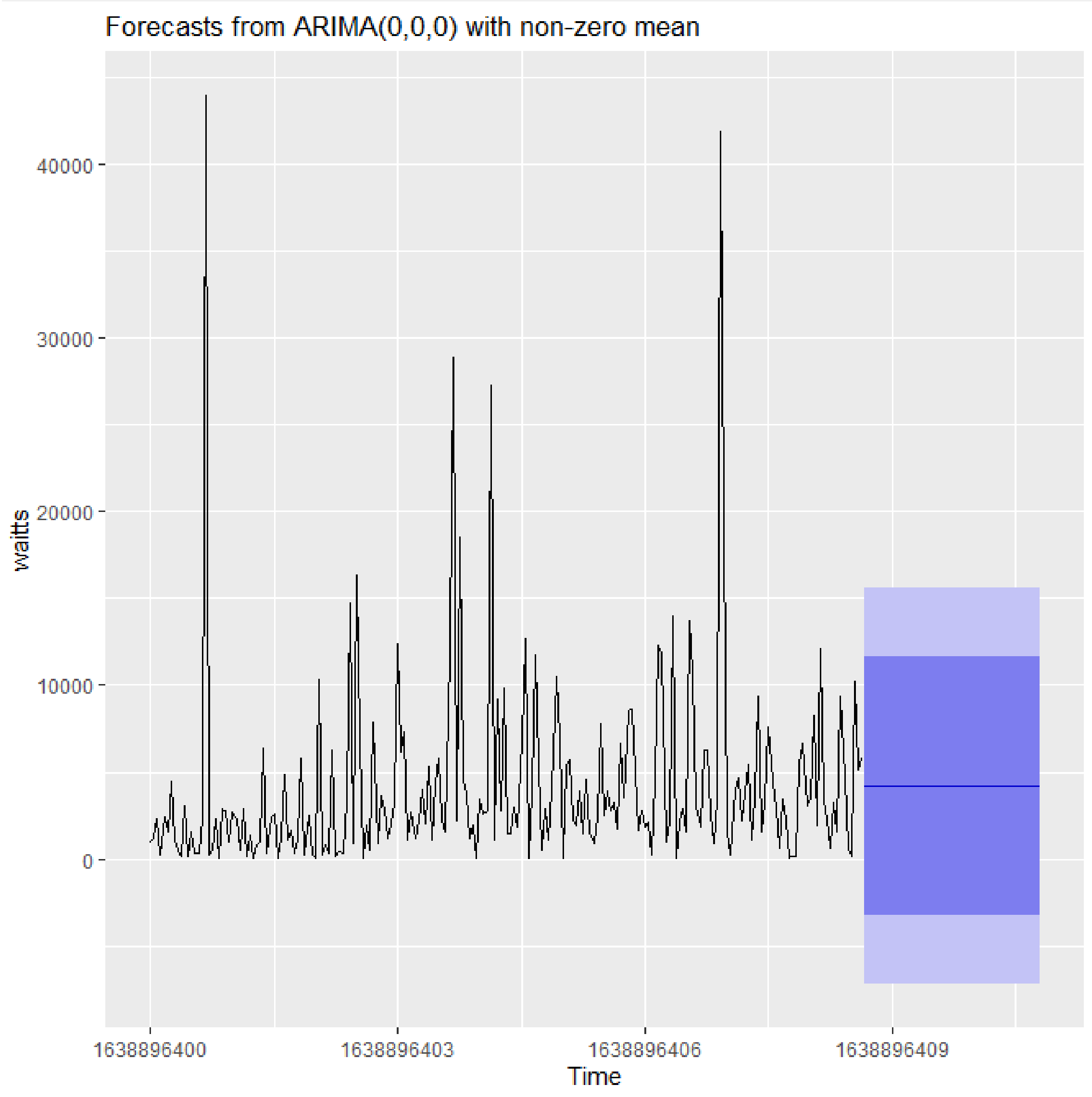


Figure 21. Forecast plot of the automatic ARIMA model.

Figure 24 showed a unique result in the sense that this was a white noise model, meaning that the function was not able to find a better model as the data was too volatile and changing. For the automatic function there was nothing that could be reliably predicted, and each predicted forecast would be random. Although the automatic ARIMA function could not find a better model than this, it does not mean that a better model did not exist, and that the previously created manual ARIMA model would not perform better. As the baseline models already took the average model into account, which was the most basic forecasting model, could this ARIMA(0, 0, 0) model be discarded.

4 Accuracy Analysis

After building all the different forecasting models for both sets of data, demand and wait time, the next step was to calculate and compare the accuracies of the different models. Before training the forecasting models, the data was split into training and testing set. Training set of the data was provided to the forecasting models, and which the forecasting models used to create their predictions. After this step the predicted values were compared against the real values from the test data set, which the forecasting models had not seen before, from which the difference between the calculated and real value could be recorded. This difference was the error in the forecasting model. Many methods exist for calculating this error, each of which can tell a different story, which is important to understand when comparing any models to each other. The best model could be chosen after comparing the different accuracy scores, which then could be used for further solution development in the future.

In the forecasting models in the previous chapter, there was a clear difference between forecasting models, some of which had much higher variance in the forecasted observations. It should be noted that although higher variance could look at first glance that the model is doing more, and at the same time providing a model with higher accuracy, in reality this is not always the case. If the data was closer to white noise, meaning that the observed values are more random, a model giving more even forecasts could prove to be more accurate in the long run. This is apparent in stock markets, and how the drift model is still widely in use, due to how unpredictable stock markets tend to be, as well as how much less computing power these simpler methods use compared to more complex models. This allows for higher volume of forecasting models to be calculated, which can be crucial for stock markets where data can change within seconds. Therefore, comparing the accuracy score becomes important when comparing forecast models to each other, as this is the only way to obtain an actual understanding of how accurate a model truly is.

Many different methods exist to calculate the accuracy and how the errors are interpreted. Each of these methods have their place when comparing forecasting models, and which error calculation model is desired depends largely on the wanted outcome and what characteristics are desired. When considering which error model should be used for the forecasting models within this project, one of the largest deciding factors was the lag of requirement to compare the forecasting accuracy between data sets. From this the possibilities could be narrowed down to mean absolute error and root mean squared error; MAE and RMSE respectively. (Hyndman 2018)

Both methods are widely used as they are easy to read and fast to calculate. From here the main difference between the two is that in RMSE the error values are squared before calculating the average with root taken from the result, which results in higher error values being represented more in the final error score. Thus, the main consideration here between the error models is whether or not large errors are especially undesirable. It is possible to also compare the values between MAE and RMSE to obtain a better understanding of the magnitude of large errors in the forecasting data.

4.1 Accuracy Score Comparison

As the project was forecasting data that will be used to assist drivers with making better decisions, large errors in the forecasts was expected to result in frustration from the users. Due to this RMSE was used as the main criteria in choosing the used forecasting model. With the RMSE score, lower the score, better the accuracy was deemed to be. Table 4 shows the accuracy results for the demand forecast; the first column having name of the forecasting model, the second column showing the accuracy score for the training data, and the third column showing the accuracy score for the test data.

Table 4. Demand forecasts RMSE scores.

Demand RMSE score (lower is better)	Training	Test
Average	1.212183	6.752475
Naïve	1.080123	4.795832
Seasonal Naïve	1.566008	5.621388
Drift	1.079319	4.713714
Linear model	1.102608	6.077056
Manual Arima	0.8093425	3.7115617
Auto-Arima	0.8187895	3.4577401

Table 4 showcased the different RMSE scores received from the created forecasting models, split between scores on the training and test data. Training data was what the forecasting models are based on, the data that the model was provided, while the test data was testing how well the model works when it was compared against data that the forecasting model had not been calculated with, data that the forecasting model had not seen. This test data was a way to simulate real world conditions where the actual data was still unknown. From Table 4 it was seen that the worst accuracy was provided by the average model, which was one of the baseline models, with the linear model as the second worst. As the linear model was worse than three of the base models, it was clear to say that this was not considered further. From the ARIMA models, the automatic ARIMA model performed slightly better, while giving the best score of all the forecasting models. It was interesting to note that the manual ARIMA model performed slightly better than the automated counterpart, which at this stage does not say much, but in the future the ARIMA model should be periodically analyzed to make sure that the best model is still in use.

Table 5 shows the accuracy results for the wait time forecast; the first column having name of the forecasting model, the second column showing the accuracy score for the training data, and the third column showing the accuracy score for the test data.

Table 5. Wait time forecasts RMSE scores.

Wait RMSE score	Training	Test
Average	3779.022	4110.954
Naïve	5189.912	4727.744
Seasonal Naïve	5016.505	9202.526
Drift	5189.911	4698.993
Linear model	5381.161	4993.670
Manual Arima	5610.427	3793.909
Auto-Arima	5806.068	3807.112

Table 5 showed the scores that were received from the accuracy of the wait time forecasting models. What was interesting to note from the received scores was that with the training data, the baseline models performed significantly better to the actual forecasting models, while with the test data the results were the other way around. As the test performance was what matters the most, was this the main consideration here. The main difference here when compared to the Table 4 was that the seasonal naïve baseline model performed worst, while the manual ARIMA model gave the best performance. Again, the linear model failed to perform better than three of the baseline models and was not considered further.

5 Summary and Conclusions

In conclusion, this project was able to successfully analyze, aggregate, and create a forecasting model for the data that was provided as the created forecasts performed better than the baseline models. With the goal being to identify a forecasting model that could be used for the data that is being collected to the customer, ARIMA model could be confidently said to fulfil this condition. In the future it will be more matter of how much time will be wanted to be dedicated to finding possibly only marginally better model, as the automated function was proven to provide close results with the manually analyzed and created models. It should be noted that other forecasting models exist that were not used within this project that could prove to provide an improved model. One of these would be neural network models which were considered to be included within this project, but due to resource constraints had to be excluded in the end.

Every project has its own set of problems emerging during it, and even if there have not been any notable problems, there should typically be something to learn and improve on. As for this thesis, several points could have been done better to make it go more smoothly, but in the end the project was able to keep up with the original timeline. Most of the problems came from the technical details of handling the data; splitting the data into train and test data too early, test data being included in the dataset that had to be identified and removed, having the data in correct format, and visualizing the graphs properly. There were also many problems related to the coding itself, having to perform extensive troubleshooting at times, which ended up requiring more time, but which was accounted for to the timeline at the beginning. Outside factors did also influence the project, mainly communication to different parties taking time to receive a response, and other responsibilities providing pressure on the schedule that was agreed for this project. Lastly the final part of the project, finalizing the report, took longer than originally planned due to more stakeholders having to review the text at different stages of the project. In the end this all was successfully managed, with only minor delay to the original schedule.

There were also several other methods tried during the project that did not make it into the final report. The main one was using ANN model as one of the forecasting models, which was tested but due to the resource limitations had to be dropped. ANN model could be tested as part of the future development. Another one was trying oversampling to increase the amount of data available for the forecasts, which was dropped as it did not add to the quality of the work nor the result.

5.1 Future Development

Although a successful forecasting model was identified, more work could still be done before reaching the most optimized model according to best practices. Future development should include model optimization where different amounts of data is provided for the same forecasting model to graph out the accuracy that is received from different amounts of data. This will be helpful in determining how much data is optimal to be used for the best accuracy while keeping the computing amounts as low as possible. As more data is provided for forecasting models, the longer it will take for computers to calculate the forecasting model from the provided data. For this sufficient amount of historical data would be required.

Another improvement could be done in how the wait time data was processed before creating the forecasting models. From the residuals that was received from the forecasting models, the possible error amount can regularly vary up to an hour, which when considering that the data was forecasting expected wait times at one-hour intervals, could this quickly prove to be a very frustrating experience for the users who would utilize the forecasts. In the data it was possible to identify multiple outliers whose wait time exceeded multiple hours, which should be investigated into what was the cause of this. As this would likely involve more than looking at the available data to figure out the cause for, it was impossible to fully take into consideration within this project. Although investigating the root causes for this would be the ideal solution, could this be labor intensive solution. Less time consuming, and easy to implement solutions exist, such as implementing a certain maximum value that if the wait time exceeds, it is considered invalid as it can be assumed that something more affects those individuals wait time than what is wanted to be forecast.

After these possible additional improvements and analytics, the code should be transferred to more automated format. To deploy the code, this original method should be recreated in another coding language such as Python which is simpler to deploy and maintain in an API format at a cloud server. The code would be created to automatically perform each task and provide a forecast of the possible future values that can be accessed through an API either internally or externally according to the needs of the customer. This would allow this service to be accessed more easily by other processes that could then further utilize data provided. This was excluded from this project due to timeline restrictions.

References

Cavanaugh, Joseph. 1997. Unifying the derivations for the Akaike and corrected Akaike information criteria. Online. Science Direct. <https://www.sciencedirect.com/science/article/abs/pii/S0167715296001289?via%3Dihub>.

fma. 2020. R package version 2.4. <https://cran.r-project.org/package=fma>.

Frost, Jim. 2017. The Difference between Linear and Nonlinear Regression Models. Online. Statistics By Jim. <https://statisticsbyjim.com/regression/difference-between-linear-nonlinear-regression-models/>. Read 22.11.2021.

Hyndman, Rob & Athanasopoulos, George. 2018. Forecasting: Principles and Practice. E-book. OTexts.

IBM. 2020. Neural Networks. Online. <https://www.ibm.com/cloud/learn/neural-networks>. Read 23.11.2021.

IBM. 2021. Linear regression. Online. <https://www.ibm.com/topics/linear-regression>. Read 22.11.2021.

Makridakis, Spyros; Wheelwright, Steven & Hyndman, Rob. 1998. Forecasting: Methods and Applications. Book. Wiley.

Palachy, Shay. 2019. Stationarity in time series analysis. Online. Towards Data Science Inc. <https://towardsdatascience.com/stationarity-in-time-series-analysis-90c94f27322>. Read 12.11.2021

Palma, Wilfredo. 2016. Time Series Analysis. E-book. John Wiley & Sons, Incorporated.

Pardoe, Iain; Simon, Laura & Young, Derek. 2021. Autoregressive Models. Online. Eberly College of Science. <https://online.stat.psu.edu/stat501/lesson/14/14.1>. Read 23.11.2021.

Radečić, Dario. 2021. Moving Averages (MA) Theory and Implementation. Online. Towards Data Science Inc. <https://towardsdatascience.com/time-series-from-scratch-moving-averages-ma-theory-and-implementation-a01b97b60a18>. Read 23.11.2021.

Routa Digital. 2021. About Us. Online. <https://routadigital.com/>. Read 20.07.2021.

Statista. 2021. Volume of data/information created, captured, copied, and consumed worldwide from 2010 to 2025. Online.

<https://www.statista.com/statistics/871513/worldwide-data-created/>. Read 21.20.2021.