AMERICAN UNIVERSITY OF BEIRUT

Data Science Approach for Forecasting the Demand on Lebanese Red Cross and Civil Defense in Case of Accidents

by
ELHAM MOHAMAD FARHAT

A thesis
submitted in partial fulfillment of the requirements
for the degree of Master of Science
to the Department of Computer Science
of the Faculty of Arts and Sciences
at the American University of Beirut

Beirut, Lebanon
April 2019

AMERICAN UNIVERSITY OF BEIRUT

Data Science Approach for Forecasting the Demand on Lebanese Red Cross and Civil Defense in Case of Accidents

by

ELHAM MOHAMAD FARHAT

Approved by:

[illegible]

Dr. Fatima Abu Salem — Advisor

Computer Science

[illegible]

Dr. Wassim El Hajj — Member of Committee

Computer Science

[illegible]

Dr. Shady El.Bassiouny — Member of Committee

Computer Science

Date of thesis defense: April, 2019

AMERICAN UNIVERSITY OF BEIRUT

THESIS, DISSERTATION, PROJECT RELEASE FORM

Student Name:	Farhat	Elham	Mohamad
	Last	First	Middle

[✓ Master's Thesis] ◯ Master's Project ◯ Doctoral Dissertation

[✓] I authorize the American University of Beirut to: (a) reproduce hard or electronic copies of my thesis, dissertation, or project; (b) include such copies in the archives and digital repositories of the University; and (c) make freely available such copies to third parties for research or educational purposes.

☐ I authorize the American University of Beirut, to: (a) reproduce hard or electronic copies of it; (b) include such copies in the archives and digital repositories of the University; and (c) make freely available such copies to third parties for research or educational purposes after: **One ___ year from the date of submission of my thesis, dissertation or project.**
Two ___ years from the date of submission of my thesis , dissertation or project.
Three ___ years from the date of submission of my thesis , dissertation or project.

[illegible] May 7 - 2019
Signature Date

This form is signed when submitting the thesis, dissertation, or project to the University Libraries

Acknowledgements

I would like to express my immerse gratitude to my advisors, colleagues, friends and families who contributed to this thesis. First of all, I would like to thank my thesis advisor, Dr. Fatima Abu Salem, for her excellent guidance and unwavering support all the time. Her insightful advices greatly helped me during the research and learning process of this thesis.

I am indebted to my academic advisor, Professor Wassim El Hajj. His patience, enthusiasm and caring was the main support for me to accomplish this piece of work. Besides valuable guidance on my academy and research, more importantly, his understanding and support during my hardest time helped me overcome the difficulties and go through the rewarding journey.

Thanks to numerous people who indirectly contributed to this thesis. My deepest gratitude and appreciation go to my families. I am lucky to have my parents and my parents in law, who supported me during the whole journey and helped in babysitting my daughter during meetings and stressful work. I also want to thank my amazing husband for standing beside me throughout the years.

An Abstract of the Thesis of

Elham Mohamad Farhat for Master of Science
Major: Computer Science

Title: Data Science Approach for Forecasting the Demand on Red Cross and Civil Defense in Case of Accidents

Being able to predict demand by the victims on Red Cross and Civil Defense services plays a vital role in Lebanon which has been reeling under the effect of regional struggles. This thesis attempts to explore data from tweets of the Lebanese Red Cross and Civil defense services to gain insight into the demand following car accidents and fire outbreaks. These tweets published by the Red Cross and Civil defense services Twitter accounts correspond to the SOS calls issued by the victims incurred during the years 2015-2016. We collect the data by scraping Twitter and generate from the Tweets a univariate time series representing demand on a daily basis for the named two years. Each tweet includes the time when the response was recorded, the location where the service took place, and the type of accident (car accident or fire outbreak).

Before attempting the multivariate predictive modeling employed by several machine learning models, it is essential to explore the classical ARIMA and SARIMA models in order to provide a baseline performance against which other machine learning models can aim to beat. The first part of the thesis provides a comprehensive ARIMA and SARIMA study for forecasting the weekly, biweekly and monthly demand for Red Cross and Civil Defense services in Lebanon using data provided by their respective tweets. Forecasting is trained on the three data sets, and delivers a tool that can be employed by the Lebanese Red Cross and Civil Defense Services to be able to predict the expected demand on a weekly, biweekly and monthly basis. Beginning with an exploratory analysis of the data, we process the time series with regards to its seasonality and stationarity leading up to the actual modeling. We conclude with a comparative assessment between the ARIMA and SARIMA models on new data pertaining to the year 2017.

The fact that demand cannot be simplified as a simple, univariate time series phenomenon but is rather actually dependent on weather conditions and public holidays or events, makes it compelling to consider machine learning predictive modeling techniques where the second part of the thesis moves the problem to another dimension. We follow two approaches,a regression problem to predict numeric demand based on machine learning models and a classification problem to classify whether the demand is low or high according to a set of selected features based on the interpretation of accuracy measures produced by several machine learning models. Finally, significant results are reported and the best models among the two approaches are selected to make predictions on unseen data.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 The Need for Demand Forecasting in Lebanon

Being able to predict demand by the victims on Red Cross and Civil Defense services plays a vital role in Lebanon which has been reeling under the effect of regional struggles. Not only does the underestimation of the demand bear impact on both the host as well as large refugee communities residing in Lebanon. Failing to understand how demand fluctuates due to other variables than time, such as weather conditions, geographical location, and public holiday or events, also affects the prediction accuracy.

Several works have appeared that address Red Cross operations and the ability to help them predict hydro-meteorological disasters in West Africa, the Philippines, and other countries that are constantly at high risk of floods and climate-related disasters [1, 2, 3]. On the other hand, a lot has been done in the field of demand forecasting for the public good in general, for forecasting electricity and primary health care demand following refugee crises in Europe, Canada and Turkey [4, 5], to name a few examples. Our work is the first of its kind for Lebanon, which in turn suffers from a very high rate of traffic and fire accidents.

According to the latest WHO data published in 2017 Road Traffic Accidents Deaths in Lebanon reached 1,129 or 3.34% of total deaths. The age adjusted Death Rate is 19.22 per 100,000 of population ranks Lebanon 86 in the world. In addition to that Fires Deaths in Lebanon reached 81 or 0.24% of total deaths. The age adjusted Death Rate is 1.37 per 100,000 of population ranks Lebanon 104 in the world [6]. These statistics show that we are against a serious problem, especially that Lebanon's population is increasing, so more traffic and fire accidents are expected to happen in the coming years. Predicting the demand rate for car or fire accidents helps in avoiding this increase in death rates, since the red cross or civil defense help would be ready and prepared beforehand and hence avoid the death of many.

1.2 Problem Statement and Thesis Motivations

The main problem handled in this piece of work is predicting the severity of demand for red cross and civil defense in case of car or fire accidents in Lebanon.

This report attempts to explore data from tweets of the Lebanese Red Cross and Civil defense services to gain insight into the demand following car accidents and fire outbreaks. These tweets published by the Red Cross and Civil defense services Twitter accounts correspond to the SOS calls issued by the victims incurred during the years 2015-2016. We collect the data by scraping Twitter and generate from the Tweets a univariate time series representing demand on a daily basis for the named two years. Each tweet includes the time when the response was recorded, the location where the service took place, and the type of accident (car accident or fire outbreak). The resulting time series data sets are treated as a univariate, single-step time series with a time step equal to one week, two weeks and one month.

Classical linear methods such as ARIMA and SARIMA will be used for the univariate time series forecasting part of this project where these will provide us with a baseline performance for other models to beat. In addition to that, and since we know that features other than time are also significant and influence the demand rate, in the other part of this thesis we will handle the multivariate forecasting of demand based on weather and events features. The problem in this part changes to classification problem where our aim becomes to predict the severity of weekly demand whether High or Low using classification machine Learning models.[7].

1.3 Thesis Outline

This thesis is organized as follows:

Chapter 2 overviews the theory of univariate and multivariate time series and their applications on demand forecasting problem. Classical linear models such as SARIMA and ARIMA introduced in section 2.2 as well as machine learning and feature selection techniques in section 2.3. The importance and need to an exploratory data analysis in any data science project is shown and discussed in section 2.4. Moreover, previous predictive models for demand on red cross or civil defense are described in section 2.5.

Chapter 3 shows our exploratory data analysis done for this project. It includes the basic data preparations, how data was collected and preprocessed in sec. 3.1 Besides, it studies the evaluation techniques and train test split methods and the data sets by decomposing it into its main components and studying the data stationarity, statistics and visualizations in sections 3.2, 3.3, and 3.4 .

In Chapter 4 , ARIMA modeling is studied and applied on our data where in section 4.1 The ARIMA parameters are configured and chosen. In section 4.2 ARIMA parameters are grid searched and the ones with best RMSE results were chosen. Residula errors are studied and corrected in section 4.3. Finally, improving results is done using box cox transform in section 4.4.

In Chapter 5, SARIMA modeling is examined and again tuning of parameters is done in section 5.1. After that non-seasonal parameters are studied in section 5.2, and seasonal parameters in section 5.3. Then an intuitive SARIMA approach is done in section 5.4, and BIC-based approach is done in section 5.5. At the end a comparison between both approaches is done to know which did a better job in forecasting demand in section 5.6.

In Chapter 6 the chosen ARIMA and SARIMA models are applied on out of sample data and error measures are reported in each of sections 6.1 and 6.2.

In Chapter 7 a different approach is proposed that states using multivariate data to make predictions. Two ways are proposed: predicting numeric value of demand using regression models (sec. 7.4) and making the problem a classification problem in which an observation is classified to either high or low demand (sec. 7.5). First, an explanation is given on why we went for this approach and how our problem was restated. Then we describe the multivariate data set being used in this approach in section 7.1. Section 7.2 explains how did we label our data, and then feature engineering and selection is done to understand what features are more significant than the others in section 7.3. Finally metric evaluation is done on regreesion models and accuracy measures of classifiers are reported, the model with the best performance on training data is chosen to be tested on test data.

Chapter 8 talks about LIME test, a new approach that tests how much can we trust the chosen model by interpreting the relation between features and classes. Finally, summary of the whole thesis is given in chapter 9.

Chapter 2

Literature Review

The objective of this literature survey is to give a general overview about some important concepts related to the thesis work. We start by giving an overview about demand forecasting done on univariate and multivariate time series data (sec.2.1). Then we introduce linear models such as ARIMA and SARIMA and their application on univariate time series forecasting (sec.2.2). After that a general overview on the use of machine learning approaches in the field of demand forecasting is given in sec.2.3 . The importance and need of exploratory data analysis in any data science project is over-viewed in sec.2.4. Finally, we will show some examples of data science projects done for the sake of social good(sec.2.5).

2.1 Overview on Demand Forecasting using Time Series Data

"Demand forecasting is a field of predictive analytics which tries to understand and predict customer demand to optimize supply decisions.."[8] The need for demand forecasting first appeared in business where it helped in production planning, inventory management or in making decisions whether to enter a new market. The main goal behind its use is to increase the profit and make more money. This process involved using historical sales data and statistical techniques to make predictions.

Data science for the social good is a newly developed approach that data scientists are adopting in order to make use of data available for a good reason that may influence the human life other than increasing profit or winning the elections. As it is mentioned in the definition above demand forecasting is used to optimize supply decisions, that is to know beforehand how much service will a customer ask for and hence provide this service in a better way.

A very related issue in this domain is the use of time series data for applying demand forecasting. A time series is a set of observations generated sequentially in time [9], time series forecasting is the process of predicting future events by estimating what has already happened. Time series data can be either univariate that depends only on one feature that is time, or multivariate in which it has more than one feature in addition to time. In both cases, The measurements taken during an event in a time series should be arranged in a proper chronological order. The procedure of fitting a time series to a proper model is termed as Time Series Analysis. A time series in general is assumed to be affected by four main components: trend, cyclical, seasonal and irregular components [10]. Most of the time these components are studied and considered useful in the forecasting process, in some other cases the trend and seasonal components are removed ad models are applied on irregular component to get better forecasts. Adding the trend or seasonality component to the data provides a very useful information to the model and helps in forecasting. We will be using such techniques in the multivariate part of this thesis.

In addition to time series components, we are often interested in features that are observations made at prior times, called lag times or lags [11]. These give us more knowledge about the future, especially when we are dealing with a demand forecasting problem. For instance, lets say the problem is to predict the demand on health care by citizens of Beirut for the next year, in this case it is extremely important to know the demand on health care for the past two or more years.

Again knowing the history of demand gives the model more information (lags) that can be used to give better forecasts.

2.2 ARIMA and SARIMA for Univariate Time Series Forecasting

Selecting a good forecasting model is extremely important as it reflects the underlying structure of the series, and more importantly, how much the fitted model is useful for future forecasting [12]. There are two widely used linear time series models: Autoregressive (AR) and Moving Average (MA) models [9, 13]. These models are widely analyzed and used so we will not handle them in details here. An ARMA(p, q) model is a combination of AR(p) and MA(q) models and is particularly suitable for univariate time series modeling [14]. However, the ARMA models, described above can only be used for stationary time series data.

In practice, many time series show non-stationary behavior, such as those related to business and socio-economic as well as those contain trend and seasonal patterns [15, 16]. Thus from application point of view, ARMA models are inadequate to properly describe non-stationary time series, which are frequently encountered in practice. For this reason the ARIMA model [17] is proposed, which is a generalization of ARMA model to include the case of non-stationarity as well [13]. In order to fit a stationary model, it is necessary to remove non-stationary sources of variation. This can be done by differencing which is the main difference between the two models ARMA and ARIMA. The ARIMA model was found by Integrating ARMA(p, q) to the dth order [4, 18] and can be shown as ARIMA(p, d, q), where d is the number of nonseasonal differences needed for stationarity.

Knowing whether a time series is stationary or not is yet another issue. A stationary data set is one that has no trend or seasonal factors. Besides the typical models above, more specific and varied models are proposed in literature. For instance, for seasonal time series forecasting, a variation of ARIMA, the Seasonal Autoregressive Integrated Moving Average (SARIMA)[13] model is used. ARIMA model and its different variations are also broadly known as the Box-Jenkins models for the reason that they are based on the famous Box-Jenkins principle [19]. Box Jenkins method was initiated by the authors Box and Jenkins who suggested a process for identifying, estimating, and checking models for a specific time series dataset [11]. Box-Jenkins method suggests that a stationary time series can be modeled using ARMA model (autoregressive moving average model) and a non stationary time series can be modeled using ARIMA model(autoregressive integrated moving average model).

These models are considered parametric due to the fact that each model has a set of parameters that need to be configured before doing forecasts. All these methods employ a four step approach to create a model. First, the model is formulated as a hypothesis. Then, a specific model is formed by selected variables based on observations. In the third step, model parameters are estimated by least-squares or maximum likelihood. At the final step, performance of the model is tested with selected variables and parameters [4]. If the performance of the model meets predefined criteria, then the forecasting model is accepted. Otherwise, new parameters for model are estimated. We repeat this procedure until a set of parameters is found that shows the best performance according to the set criteria.

On the other hand SARIMA model is used when a time series may possess seasonal patterns such as daily, weekly, monthly, etc. A SARIMA model is an extended version of ARIMA model with additional seasonal terms and can be shown as ARIMA(p, d, q) $\times$ (P, D, Q)s, where P is the degree of seasonal AR model, Q is the degree of seasonal MA model, D is degree of seasonal integration, and s is the span of repeating seasonal pattern [4].

We mentioned earlier that it is important to study the choices of these models parameters based on some criteria. The literature reveals that most of the time the root mean squared error (RMSE) is used to evaluate the performance of a model given a set of parameters. The reason behind this is that RMSE punishes large errors and results in a score that is in the same units as the forecast data. In addition to that mean absolute percentage error (MAPE) is also used in many cases where there is a need to compare different models based on percentage and not actual values.

It is important to emphasize that ARIMA and SARIMA models are only used on univariate time series data; however, there are other stochastic models that were invented from AR and MA models to work on multivariate time series such as vector auto-regression VAR or vector moving average VMA models which use vectors to include the set of variables in the regression equation. We will not focus on these models in this review but it was significant to mention that these models exist and may perform very well in some cases.

2.3 Introduction to the Use of Machine Learning in demand forecasting

Machine learning models have been established as serious contenders to classical statistical models in the area of forecasting in the last decade [12]. Subsequently, the concept is extended to other models, such as support vector machines, decision trees, and others, that are collectively called machine learning models [12, 20]. Some of these models are developed based on the early statistics models [21]. Huge advances have been made in this field in the past years, both in the amount and variations of the models developed, as well as in the theoretical understanding of the models. Machine learning approaches are widely used for building accurate forecasting models [22, 23]. Literature searches have found the previous works where the machine learning techniques are used in combinations with the econometrics models. The results from different techniques are integrated to obtain better forecasting accuracy [24].

In practice, machine learning models are most of the time used for classification problems such as support vector machines (SVM), multilayer perceptron (MLP), naive bayes, decision trees and more and more.. It is important to understand the main purpose behind any model in order to know when to use it.

The initial aim of SVM was to solve pattern classification problems but afterwards they have been widely applied in many other fields such as function estimation, regression, and time series prediction problems [25]. The impressive characteristic of SVM is that it is intended for a better generalization of the training data. In SVM, instead of depending on whole data set, the solution usually only depends on a subset of the training data points, called the support vectors [26]. Furthermore, with the help of support vector kernels, the input points in SVM applications are usually mapped to a high dimensional feature space, which often generates good generalization outcomes.

Another significant models used in this domain are the deep learning models especially LSTM, MLP and ANN models. These models can be used to automatically learn the temporal dependence structures for challenging time series forecasting problems. Neural networks may not be the best solution for all time series forecasting problems, but for those problems where classical methods fail and machine learning methods require elaborate feature engineering, deep learning methods can be used with great success. One main constraint that might not allow these models to perform well on time series data is when the data size is small, because these complex models are able to automatically learn arbitrary complex mappings from inputs to outputs and support multiple inputs and outputs. These are powerful features that offer a lot of promise for time series forecasting, particularly on problems with complex-nonlinear dependencies, multivalent inputs, and multi-step forecasting. In this thesis, deep learning models did not perform well on time series data because our data was not large enough for such models to handle.

On the other hand, it is very essential to mention that machine learning models give a great significance to features in any prediction case. Demand is highly affected by variables other than time, it might be weather, events or any other related variable that varies with time and has influence on the demand. For that reason multivariate data shows better results when modeled by machine learning models either for classification problems or regression, in both cases the features that are highly correlated to the demand improves the forecast results.

Therefore, it is highly recommended for any machine learning problem using multivariate data to study feature importance and do feature selection thus avoiding those features that are of small influence on demand and emphasizing on features that are of great influence on demand. In the literature several techniques are mentioned for feature selection such as filter methods, wrapper methods and embedded methods.

Feature importance is used to help estimate the relative importance of contrived input features for time series forecasting.[4] Feature importance is one method to help sort out what might be more useful in when modeling. The method involves fitting a random forest model (RandomForestRegressor), and summarizing the relative feature importance scores.[27]

A popular method for feature selection that is an example of wrpper methods is called Recursive Feature Selection (RFE). We will be using this method in the thesis. RFE works by creating predictive models, weighting features, and pruning those with the smallest weights, then repeating the process until a desired number of features are left.[27]

2.4 Exploratory Data Analysis

The most important part in any data science project is the exploratory data analysis (EDA). EDA is an approach in which a data set is analyzed by the use of visualizations such as line plots, distribution plots, autocorrelation plots etc.. These visualizations describe the basic characteristics of any data set and especially time series data sets.

"Exploratory Data Analysis refers to the critical process of performing initial investigations on data so as to discover patterns,to spot anomalies,to test hypothesis and to check assumptions with the help of summary statistics and graphical representations", as defined by data scientists [28].

Visualization plays an important role in time series analysis and forecasting. Plots of the raw sample data can provide valuable diagnostics to identify temporal structures like trends, cycles, and seasonality that can influence the choice of model. A problem is that many novices in the field of time series forecasting stop with line plots. In this overview, we will take a look at 6 different types of visualizations that you can be used on time series data [11]. They are:

- Describe data Statistics: This includes all important statistics of any data set such as the number of instances, mean, std, max , min etc.. It is important to study the statistics of any time series in order to know how values are distributed around the mean and how much variance we have which may affect the predictions. It also shows the level of data that is represented by the mean which helps us compare forecast errors and study the performance of any model.
- Line Plot: One of the most popular visualizations is line plot. This plot helps in identify any trend or seasonal cycles in the time series. It is plotted by having the time values on x-axis and observation values on y-axis.[11]
- Histogram and density plot: These are distribution plots and gives information about the distribution of values in the time series, that is whether a time series is Gaussian or not.. The thing here is that these plots describe the data set regardless of temporal structure. Some linear models assume the time series to have a bell curve or normal distribution, therefore through these plots we can know whether we have normal distribution or not.

  Also these plots are helpful when we do any transformation to the data where they show how the distribution changed.
- Box and Whisker plot: Different from the histogram and density plots, box plots focus on the distribution of values by time interval. This plot draws a box around the 25th and 75th percentiles of the data that captures the middle 50% of observations. A line is drawn at the 50th percentile (the median) and whiskers are drawn above and below the box to summarize the general extents of the observations. Dots are drawn for outliers outside the whiskers or extents of the data. Box and whisker plots can be created and compared for each interval in a time series, such as years, months, or days [11].
- Heat Maps: A matrix of numbers can be plotted as a surface, where the values in each cell of the matrix are assigned a unique color. This is called a heatmap, as larger values can be drawn with warmer colors (yellows and reds) and smaller values can be drawn with cooler colors (blues and greens). Like the box and whisker plots, we can compare observations between intervals using a heat map [11].
- Scatter plot : Time series modeling assumes a relationship between an observation and the previous observation. Previous observations in a time series are called lags, with the observation at the previous time step called $lag1$, the observation at two time steps ago $lag{=}2$, and so on. A useful type of plot to explore the relationship between each observation and its lag is called the scatter plot. It plots the observation at time t on the x-axis and the $lag{=}1$ observation $(t-1)$ on the y-axis.

  - If the points cluster along a diagonal line from the bottom-left to the top-right of the plot, it suggests a positive correlation relationship.
  - If the points cluster along a diagonal line from the top-left to the bottom-right, it suggests a negative correlation relationship. Either relationship is good as they can be modeled.
  - More points tighter in to the diagonal line suggests a stronger relationship and more spread from the line suggests a weaker relationship. A ball in the middle or a spread across the plot suggests a weak or no relationship. [11]

2.5 Similar Approaches Done in the Literature

A lot has been done in the literature about demand and time series forecasting. Here is a list of projects done in this domain that were also very helpful to us in our work:

- Real-time forecasting system of spatio-temporal taxi demand based on machine learning approaches done in New York city [12]
- Artificial neural network and SARIMA based models for power load forecasting in Turkish electricity market [4]
- Demand Forecasting on primary health care for Syrian refugees in Canada [5]
- Using Seasonal Climate Forecasts to Guide Disaster Management: The Red Cross Experience during the 2008 West Africa Floods [1]
- Climate forecasts in disaster management: Red Cross flood operations in West Africa, 2008 [2]
- Red Cross Develops Innovative Mechanism To Predict and Prepare For Flood Risks [3]

Below is a summary of a case study that was done for predicting the demand on electricity in Turkey where it followed a similar process as ours and was used as a reference to a big part of our project:

The authors in their paper "Artificial neural network and SARIMA based models for power load forecasting in Turkish electricity market." proposed two separate models, one based on seasonal autoregressive integrated moving average (SARIMA) and the other based on artificial neural networks(ANN). They believed that several factors influence the load such as calendar data (D), previous load estimation plan (L), electricity price (P), weather data (W), and currency (C). The article is then divided into three main parts. The first part handles the details of methods used in models. This part shows an introduction to parametric (like SARIMA) and non-parametric methods (ANN). It carries out the structure and mathematical formulations of each model. The second part, explains the experimental results of models. It first describes the dataset including the features that affect on electrical load. The authors used Absolute Percentage Error (APE) and Mean Absolute Percentage Error (MAPE) to measure the performances of proposed approaches. Also, they showed that selecting the correct combination of input parameters is the key to an effective electrical load forecasting system. In addition, they compared the effect of these features on forecast performance, and they showed through figures and tables the details of the feature sets and the variation in their importance on load.

Having done all this, they started building SARIMA model which can be shown as ARIMA(p,d,q) (P;D;Q)s. They used the econometric toolbox of Matlab to estimate SARIMA parameters. Moreover, they did a comparison between two SARIMA models, one based on intuitive selection of parameters and the other on Bayesian Information Criteria (BIC). They discovered that BIC-based model outer-performs the intuitive model.

Concerning ANN model, the first challenge was to know the number of hidden neurons. The results showed that 2n+1 hidden neurons (where n is the number of input neurons) had the best results in short training data sets, and that 20 hidden neurons showed better results in larger training data sets. The second challenge was to find the best learning algorithm in ANN (Scaled Conjugate Gradient (SCG), Levenberg Marquardt (LM), or Bayesian Regulation (BR)) in association with training size and combination of feature sets. The results showed that LM method was the most convenient and reasonably works good for every training data set size. As a conclusion for this part, the authors discovered that an ANN with 20 hidden neurons, trained with DL feature set of previous 12 months using LM learning algorithm produced lowest MAPE. They also concluded that a larger training data set and simpler feature sets work better on load forecasting with ANN on Turkish Market.

After they're done with each model separately, they did a comparative discussion to know which model performs better. The empirical cumulative distribution functions of MAPEs of SARIMA and ANN showed that the overall performance of ANN with calender data (D) and previous load (L) outer-performs SARIMA although the error trend seems to be the same. SARIMA's main weakness is that there is no way to distinguish between the working days and holidays. However, its benefit is that it has quick recovery from the effect of national holidays. On the other hand, ANN is the method which provides the distinction required by the nature of electric consumption.

To sum it up, in the last part the authors showed that ANN model fits better than SARIMA to Turkish Market. However, in some cases SARIMA performs better than ANN, especially on the forecasts after holidays.

2.6 Summary

In this chapter, we have reviewed stochastic and machine learning models, especially for the fields of time series modeling and forecasting. Stochastic models have been widely used and optimized for the finance, engineering and social science problems. Machine learning approaches are considered as superior methods for the forecasting problems. By applying these mathematical forecasting techniques into the real-world problems in case of accidents and if we are able to predict demand on red cross or civil defense accurately this would reduce the death rates and improve the red cross and civil defense service and make it faster. However, beyond these promising researches, there are some remaining challenges. What models or algorithms should we use to forecast the demand in real time? What if the chosen model doesn't work? How to build a robust and reliable system that can always find or even build the most appropriate models? In this thesis, we are going to solve these problems.

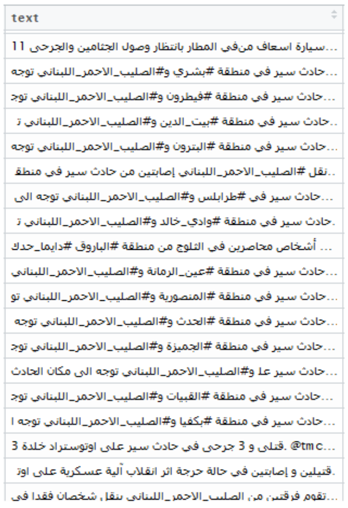
Chapter 3

Basic Data Preparation and Exploratory Data Analysis

3.1 Data Preparation

The data for this study was gathered from tweets issued by the Red Cross and Civil Defense at time junctures when accidents get reported. Our code was developed in R to scrape for the Red Cross tweets and in Java to scrape for the Civil defense tweets. Tweets tend to be messy as not all of them indicate an occurrence of an accident. As such, tweets had to be cleaned so that only significant reports are maintained.

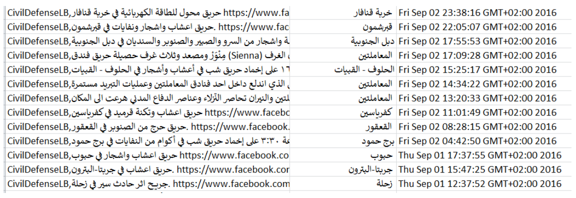
The red cross tweets have a special format that can be parsed easily and hence allowed us to retrieve our data almost immediately (see for example Fig. 3.1). The tweets are collected, cleaned, filtered, and divided into location, date-time, and text. For other advanced purposes of this project the code also gets the latitude and longitude of the location of an accident using Google maps API. The result of this code is a csv file that contains a list of places associated with a specific date-time.



text
...سيارة اسعاف منفي المطار بانتظار وصول الجثامين والجرحى 11
...حادث سير في منطقة #بشري و#الصليب_الاحمر_اللبناني توجه
...حادث سير في منطقة #فيطرون و#الصليب_الاحمر_اللبناني توج
..حادث سير في منطقة #بيت_الدين و#الصليب_الاحمر_اللبناني ت
...حادث سير في منطقة #البترون و#الصليب_الاحمر_اللبناني توجه
..نقل #الصليب_الاحمر_اللبناني إصابتين من حادث سير في منطق
...حادث سير في #طرابلس و#الصليب_الاحمر_اللبناني توجه الى
.حادث سير في منطقة #وادي_خالد و#الصليب_الاحمر_اللبناني ت
... أشخاص محاصرين في الثلوج من منطقة #الباروق #دايما_حدك
...حادث سير في منطقة #عين_الرمانة و#الصليب_الاحمر_اللبناني
...حادث سير في منطقة #المنصورية و#الصليب_الاحمر_اللبناني تو
...حادث سير في منطقة #الحدث و#الصليب_الاحمر_اللبناني توجه
...حادث سير في منطقة #الجميزة و#الصليب_الاحمر_اللبناني توج
...حادث سير عا و#الصليب_الاحمر_اللبناني توجه الى مكان الحادث
...حادث سير في منطقة #القبيات و#الصليب_الاحمر_اللبناني توج
...حادث سير في منطقة #بكفيا و#الصليب_الاحمر_اللبناني توجه ا
@tmc... .قتلى و 3 جرحى في حادث سير على اوتوستراد خلدة 3
.قتيلين و إصابتين في حالة حرجة اثر انقلاب آلية عسكرية على اوت
...تقوم فرقتين من الصليب_الاحمر_اللبناني بنقل شخصان فقدا في

Figure 3.1: Sample Red Cross Tweets

Using the Java code we were able to retrieve all tweets of Civil Defense, after which we used Excel to clean and divide the data. For example, we know that the date and time are always placed at the end of a tweet so it can easily be taken out through a simple formula (see for example Fig. 3.2).



CivilDefenseLB,حريق محول للطاقة الكهربائية في خربة قنافار https://www.fa	خربة قنافار	Fri Sep 02 23:38:16 GMT+02:00 2016
CivilDefenseLB,حريق اعشاب واشجار ونفايات في قبرشمون. https://www.fac	قبرشمون	Fri Sep 02 22:05:07 GMT+02:00 2016
CivilDefenseLB,ـة واشجار من السرو والصبير والصنوبر والسنديان في دبل الجنوبية	دبل الجنوبية	Fri Sep 02 17:55:53 GMT+02:00 2016
CivilDefenseLB,ـ الغرف (Sienna) مِنْوَرْ ومصعد وثلاث غرف حصيلة حريق فندق	المعاملتين	Fri Sep 02 17:09:28 GMT+02:00 2016
CivilDefenseLB,١٦ على إخماد حريق شب في أعشاب وأشجار في الحلوف - القبيات	الحلوف - القبيات	Fri Sep 02 15:25:17 GMT+02:00 2016
CivilDefenseLB,ـ الذي اندلع داخل احد فنادق المعاملتين وعمليات التبريد مستمرة	المعاملتين	Fri Sep 02 14:34:22 GMT+02:00 2016
CivilDefenseLB,ـلتين والنيران تحاصر النّزلاء وعناصر الدفاع المدني هرعت الى المكان	المعاملتين	Fri Sep 02 13:20:33 GMT+02:00 2016
CivilDefenseLB,حريق اعشاب وتكنة قرميد في كفرياسين https://www.facebo	كفرياسين	Fri Sep 02 11:01:49 GMT+02:00 2016
CivilDefenseLB,حريق حرج من الصنوبر في القعقور. https://www.facebook.	القعقور	Fri Sep 02 08:28:15 GMT+02:00 2016
CivilDefenseLB,ـعة ٣:٣٠ على إخماد حريق شب في أكوام من النفايات في برج حمود	برج حمود	Fri Sep 02 04:42:50 GMT+02:00 2016
CivilDefenseLB,حريق اعشاب واشجار في حبوب https://www.facebook.co	حبوب	Thu Sep 01 17:37:55 GMT+02:00 2016
CivilDefenseLB,حريق اعشاب في جربتا-البترون. https://www.facebook.con	جربتا-البترون	Thu Sep 01 15:47:25 GMT+02:00 2016
CivilDefenseLB,جريح اثر حادث سير في زحلة. https://www.facebook.com	زحلة	Thu Sep 01 12:37:52 GMT+02:00 2016

Figure 3.2: Sample Civil Defense Tweets

We used Buckwalter Arabic transliteration tool, an ASCII transliteration tool only, from which we were able to extract the dates comprising the time series data. In order to produce weekly, biweekly and monthly time series, and after removing about 3% of the data that was deemed outlying, we performed down-sampling of daily.

In order to change the data into a time series format, that is a sequence of unique timestamps, what we did was that we counted the repeated dates, and the number of repetitions is set as the demand at this date, we ended up having a sequence of unique dates from years 2015 and 2016 and for each date a demand value is assigned. This was first done on daily data and then down-sampled using a python code into weekly, bi-weekly and monthly time series data to be used in modeling.

3.2 Choosing test data

Our data set is live-streamed; this means that we can easily collect updated data to validate our model. The data we operated on at the time of writing is for the years 2015 and 2016 (the year 2017 had many missing observations which could have affected our modeling). We decided to use the data collected for the first few months of 2017 as out of sample data. Considering that the test data set is around 10% of the studied data, we will consider the first 66 observations of 2017 (given that daily observations are about 663 observations for years 2015 and 2016) and down-sample it to get weekly, biweekly and monthly unseen data sets . We now end up with two data files for each case:

- Data.csv: that has the original data of the two years 2015 and 2016 to be studied (104 observations for weekly, 53 observations for biweekly and 24 observations for monthly).
- Test.csv: that contains the data taken from the first 66 observations of 2017 to validate the final model (12 weeks for weekly validation, 6 bi-weeks for biweekly validation and 3 months for monthly validation).

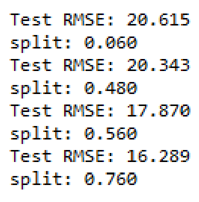
3.3 Methods for Evaluating Models

We proceed as follows:

- Do a train-test split on data from 2015-2016 that respects the dependencies between observations.
- Do multiple train-test splits on data from 2015-2016 that respect dependencies between observations and choose the one that result in best performance.
- Use the walk forward validation technique where the model is updated each time a new data is received.

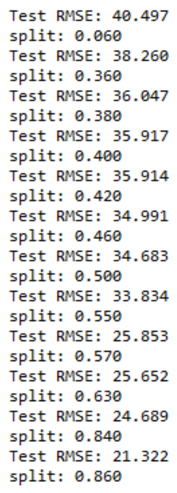
3.3.1 Train-Test Split

We will split the data set into two parts: a training set and a test set. Our data exhibits a temporal structure to be learned by the model, and so, in order to determine the best split we considered all the possible splits beginning with 5% train, 95% test all the way to 95% train, 5% test. We then apply the persistence model at each split and record the RMSE. The persistence model provides the simplest baseline in forecast performance. It is a point of reference against which to compare all other modeling techniques. The persistence algorithm uses the value at the current time step t to predict the expected outcome at the next time step $t + 1$. Finally, we choose the split that



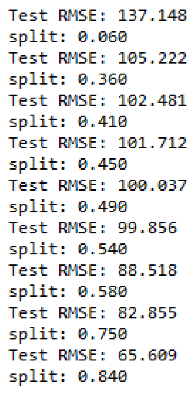
```
Test RMSE: 20.615
split: 0.060
Test RMSE: 20.343
split: 0.480
Test RMSE: 17.870
split: 0.560
Test RMSE: 16.289
split: 0.760
```

(a) Best split on weekly data



```
Test RMSE: 40.497
split: 0.060
Test RMSE: 38.260
split: 0.360
Test RMSE: 36.047
split: 0.380
Test RMSE: 35.917
split: 0.400
Test RMSE: 35.914
split: 0.420
Test RMSE: 34.991
split: 0.460
Test RMSE: 34.683
split: 0.500
Test RMSE: 33.834
split: 0.550
Test RMSE: 25.853
split: 0.570
Test RMSE: 25.652
split: 0.630
Test RMSE: 24.689
split: 0.840
Test RMSE: 21.322
split: 0.860
```

(b) Best split on biweekly data



```
Test RMSE: 137.148
split: 0.060
Test RMSE: 105.222
split: 0.360
Test RMSE: 102.481
split: 0.410
Test RMSE: 101.712
split: 0.450
Test RMSE: 100.037
split: 0.490
Test RMSE: 99.856
split: 0.540
Test RMSE: 88.518
split: 0.580
Test RMSE: 82.855
split: 0.750
Test RMSE: 65.609
split: 0.840
```

(c) Best split on monthly data

Figure 3.3: Best Splits Using Persistence Model

shows the least RMSE. Below are some of the results we obtained, leading to the best split at 76% for training and 24% for testing weekly data, 86% for training and 14% for testing biweekly data and 84% for training and 16% for testing monthly data.

The following figure shows a sample of the results obtained from running a loop to get the best split. The loop stops on the best split and shows the corresponding RMSE value above it.

The table below summarizes the RMSE and MAPE values of persistence model which will later be used to compare the performance of ARIMA, SARIMA and regression machine learning models :

Case	RMSE	MAPE
Weekly Data	16.289	37.157
Biweekly Data	21.322	32.102
Monthly Data	65.609	58.607

Table 3.1: RMSE and MAPE results of persistence model

We will apply models on the training data set and make predictions on the test data set. Moreover, we will be using the rolling forecast method , a.k.a. the walk forward validation technique in which we move one time step at a time and at each time step a model is used to produce a forecast. This predicted value is then compared with the actual expected value from the test set and added to the model for the next time step.

Finally, all forecasts on the test dataset will be collected and an error score calculated to summarize the skill of the model. The root mean squared error (RMSE) and mean absolute percentage error (MAPE) will be used as loss functions. In all that follows, n denotes the total number of predicted values in the series.

Root Mean Squared Error In this study we will use the root mean squared error as our basic criteria to study the performance of any forecasting model. RMSE is used because it punishes large errors and results in a score that is in the same units as the forecast data. In addition, Since the errors are squared before they are averaged, the RMSE gives a relatively high weight to large errors. This means the RMSE should be more useful when large errors are particularly undesirable.

The RMSE is calculated as such:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \overline{x}_i)^2}$$

Mean Absolute Percentage Error In addition to RMSE we will use in some cases MAPE loss function because it helps us compare between different models due to the fact that it measures the size of the error in percentage terms. Just as MAE is the average magnitude of error produced by your model, the MAPE is how far the model's predictions are off from their corresponding outputs on average. Like MAE, MAPE also has a clear interpretation since percentages are easier for people to conceptualize. Both MAPE and MAE are robust to the effects of outliers thanks to the use of absolute value:

$$MAPE = \frac{100\%}{n}\sum_{i=1}^{n}\left|\frac{x_i - \overline{x}_i}{x_i}\right|$$

3.3.2 Walk Forward Validation

In this study we will use the walk forward validation technique. The reason behind this is that we want to do one step forecasts which involves moving along the time series one-time step at a time. In addition we will be using the rolling forecast type model because a sliding window is used to train a model. The steps of walk forward validation are summarized below:

- We first divide the data set into two parts. One for training and the other for testing. In our case, we will use the best split studied earlier.
- Then at each time step, a new model is trained, iterated, and then a prediction is made and stored to be validated.
- For the next iteration the actual observation from the test dataset will be added to the training data set.
- Finally, we will evaluate the predictions made at each iteration and report the error.

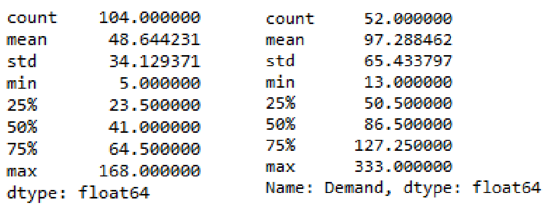
Once a prediction is made, it is appended to predictions list. The actual observation taken from test data is appended to the history i.e. training data to be used in next iterations.

Next, we will demonstrate the visual exploratory analysis conducted prior to the forecasting application.

3.4 Data Visualizations

3.4.1 Data Statistics

Below are the summary statistics of the three time series data sets:

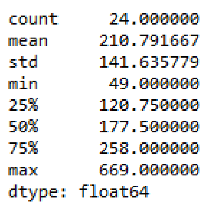


```
count    104.000000
mean      48.644231
std       34.129371
min        5.000000
25%       23.500000
50%       41.000000
75%       64.500000
max      168.000000
dtype: float64
```

(a) weekly data

```
count     52.000000
mean      97.288462
std       65.433797
min       13.000000
25%       50.500000
50%       86.500000
75%      127.250000
max      333.000000
Name: Demand, dtype: float64
```

(b) biweekly data



```
count     24.000000
mean     210.791667
std      141.635779
min       49.000000
25%      120.750000
50%      177.500000
75%      258.000000
max      669.000000
dtype: float64
```

(c) monthly data

Figure 3.4: Summary statistics

Studying the statistics of a time series is very important since it allows the data scientist to understand better the distribution of values in the time series and the mean value which shows us a level to which we compare the error measures obtained when modeling. It is better for an RMSE value of a certain model to have a noticeably lower value than the mean.

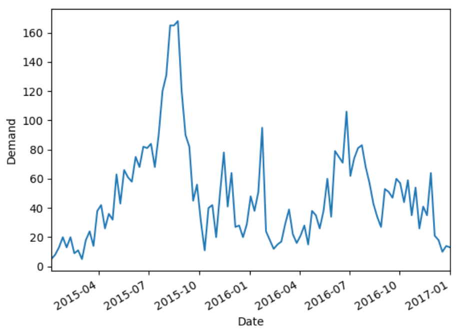
The following table summarizes the persistence RMSE values in correspondence with the mean values. These two will be used later to study the performance of our models:

Case	Persistence RMSE	Mean
Weekly Data	16.289	48.644
Biweekly Data	21.322	97.288
Monthly Data	65.609	210.791

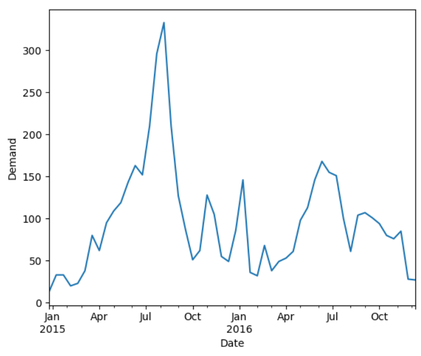
Table 3.2: RMSE results of persistence model and mean

3.4.2 Line Plot

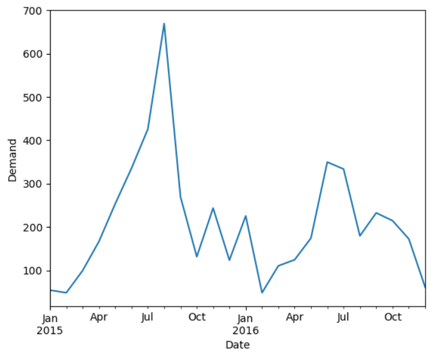
Line plots are the most common way to visualize a time series since it shows how the data varies through time and can detect the presence of a seasonality or trend structures.



(a) weekly data



(b) biweekly data



(c) monthly data

Figure 3.5: Line Plots

- Weekly Data:
  The line plot of the weekly data clarifies the seasonality issue. Notice the sharp increase and decrease in demand between July and October 2015. Similarly, we notice an increase in demand followed by a decrease between July and October 2016. Concerning the trend, we notice several fluctuations in the demand: there is no clear increasing or decreasing trend. However we may be able to catch partial trends in the data for example the increasing trend from January 2015 until August 2015.

- Biweekly Data:

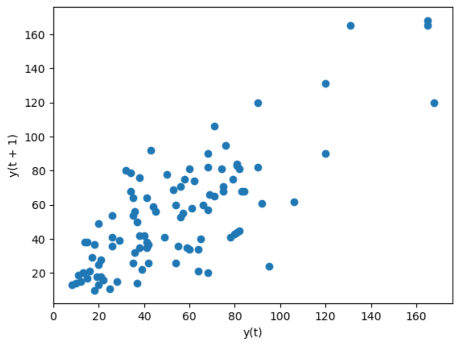
  The overall trend in the biweekly plot looks almost the same as the weekly plot. The variations looks similar as well with a smoother view (some fine grained variations are not shown), no clear seasonality appears in the plot. However we notice that the months 6, 7, 8 and 9 of 2015 show the maximum growth in the rate of demands. Fluctuations in the rate occur during the year of 2016 with again a noticeable growth in demand during months 5, 6, 8, and 9.

- Monthly Data:

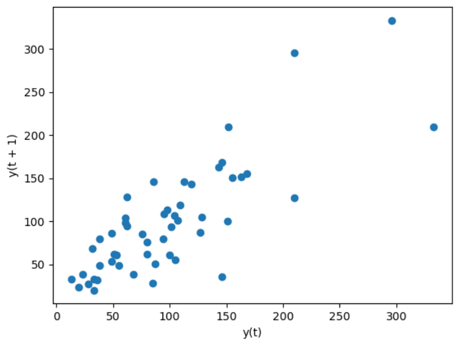
  Again the same interpretation is shown here for monthly case.

3.4.3 Lag Plot

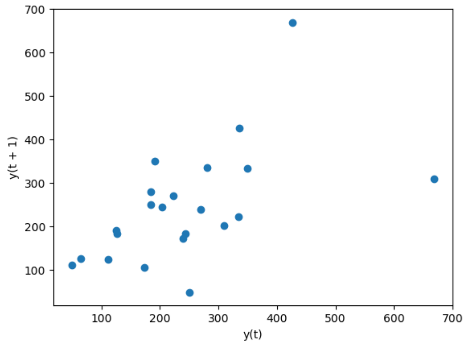
A lag or scatter plot shows how much the observations are related to previous observations. In this plot we will show the relation with lag=1 that is the observation at time $t+1$ and its lag at time t.



(a) weekly data



(b) biweekly data



(c) monthly data

Figure 3.6: Scatter plots (lag=1)

- Weekly Data:

  We notice from the scatter plot of the weekly data that there exists a positive correlation between observations and their lags. This is indicated by the elongation of dots around the diagonal in an increasing manner, and their tightness except for few outliers.

- Biweekly Data:

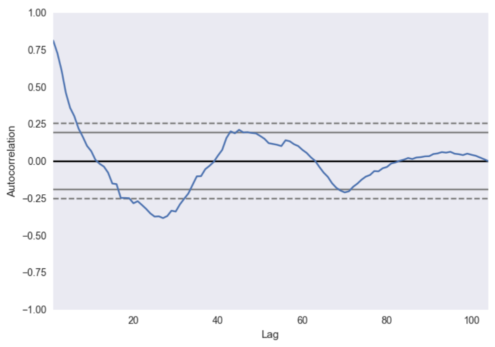
  Similarly the scatter plot for $lag=1$ regarding this dataset shows in general a strong positive correlation when the demand is below 200, above this value points appear not so tight to each other which shows a weak correlation.

- Monthly Data:

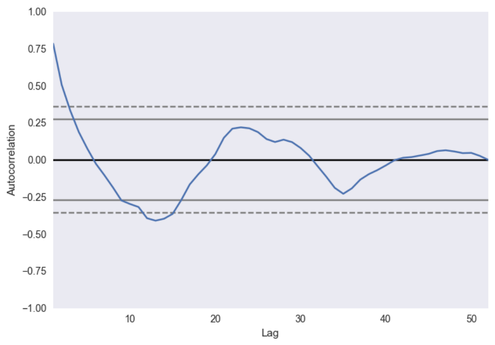
  On the other hand the scatter plot for *lag*=1 regarding monthly dataset shows in general a weak positive correlation. The values are not so tight to each other which shows the weak correlation.

3.4.4 Autocorrelation Plot

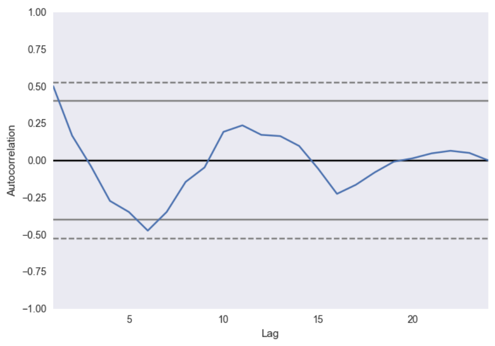
This plot helps quantify the strength and type of relationship between observations and their lags. We can see from the plots that there are positive and negative correlations (the sign identifies whether positive or negative correlation).



(a) weekly data



(b) biweekly data



(c) monthly data

Figure 3.7: Autocorrelation plots

- Weekly Data:

There are two significant points in this plot: the point at which the value of autocorrelation became insignificant at lag 8, and the point where the curve shows the greatest negative correlation at lags 25 or 30. Other values appear inside the two dotted lines, and thus are of no significant correlation.

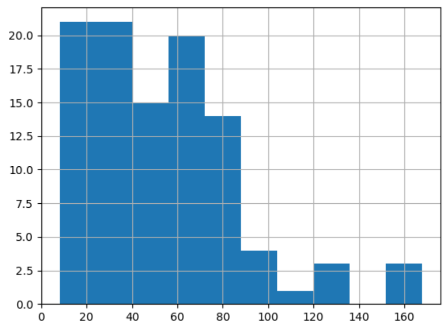
- Biweekly Data:

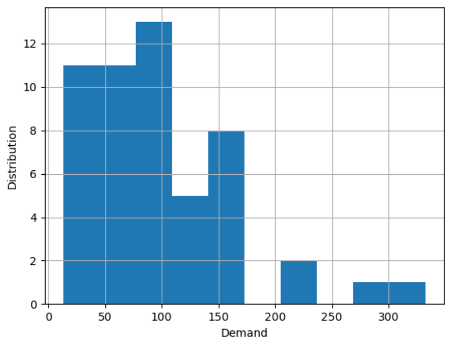
  The autocorrelation plot for this dataset shows two significant results: *lags* 0 to 3 and *lag* 13 where the curve appears to be outside the dotted lines thus indicating a significant correlation, positive one at *lag* 3 and negative at *lag* 13. The rest of the curve appears inside the dotted lines showing no significant correlations. (see Fig **??**)

- Monthly Data: The autocorrelation plot for this dataset shows that almost no significant results which ensures our significance concerning the scatter plot (Fig 3.6(c)), the curve appears to be inside the dotted lines. It might be that the very small volume of data is causing this insignificant autocorrelation behavior.

3.4.5 Histogram and Density Plots

Histograms and Density plots help us understand the distribution of observations in a time series.


(a) weekly data



(b) biweekly data

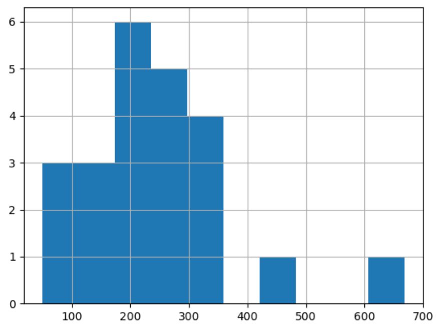

(c) monthly data

Figure 3.8: Histogram Plots

- Weekly Data:

  The histogram (Fig 3.8(a)) shows a right shift where most of the values lie between 0 and 100 demands per week. There does not appear to be a near normal distribution and the histogram shows a small tail to the right indicating that we have few large demand values. Since linear models like SARIMA and ARIMA assume normality of time series, we will need to transform the data as such. Our observations are confirmed by the associated density plot (figure 3.9(a)).

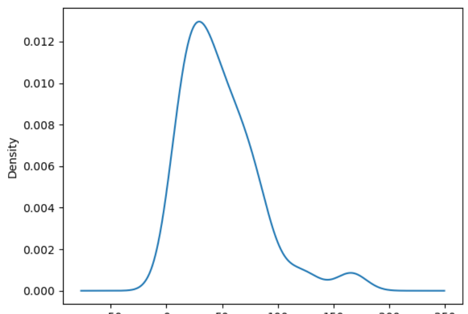
- Biweekly Data:

  We notice here (figure 3.8(b)) that the distribution is right shifted with a small tail to the right indicating that only few demand values are high (above 150).

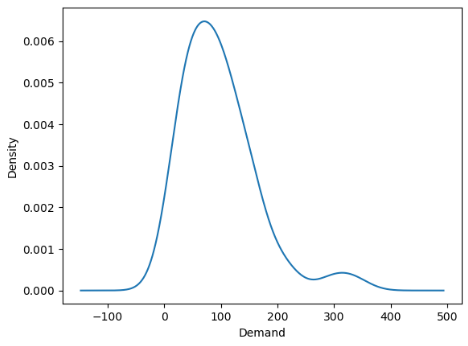
- Monthly Data:

  We notice that the distribution is skewed left where it has a large number of occurrences in the lower value cells (left side) and few in the upper value cells (right side) thus showing this short tail to the right (figure 3.8(c)). Again notice that the distribution is not normal, this might affect our results when we apply SARIMA model.

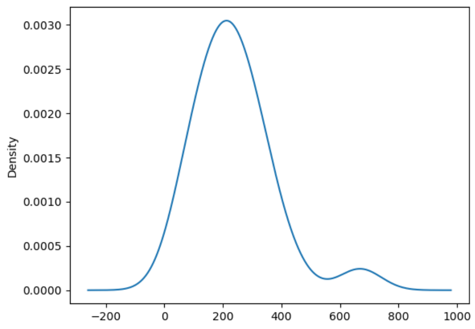
Next we will check the density plots for confirming our assumptions on the distribution of observations.



(a) weekly data



(b) biweekly data



(c) monthly data

Figure 3.9: Density Plots

- Weekly Data:

  Another smooth way to study distribution is the density plot. We can emphasize from the plots(figure 3.9(a)) that the distribution is not normal, demand rates between 0 and 100 shows the greatest density with again a small tail to the right.

- BiWeekly Case:

  We can see from the plot (figure 3.9(b)) that the distribution is close to normal around 100 with a short tail to the right. Note that the negative values in the density plot are due to the use of a function (kernel density estimate (kde)) that fits the distribution of observations and summarizes it with a nice, smooth line.

- Monthly Data:

  We can see from the plot (figure 3.9(c)) that the distribution is almost normal around 200 with a small tail to the right, demand rates between 100 and 300 shows the greatest density.

Summary To sum it up we conclude that our time series data sets show in general no clear global trend but some partial trends (increasing and decreasing), existence of weak seasonal signals, distribution of values is not uniform, and the mean is not zero in any case. Next, we will examine the seasonal decomposition and stationarity of each data set to confirm these conclusions and discover more about the structure of each time series and whether it is good for modeling.

3.5 Seasonal Decomposition

Another significant method to visualize and study data is seasonal decomposition. Seasonal decomposition illustrates how the time series is decomposed, by revealing the four important components of a time series: trend, seasonality, level and residuals. This, in turn, sheds light on the stationarity of the data set. A stationary time series is free of trend and seasonality components. Our aim here is to check whether our time series data set have a clear trend or seasonality that affect stationarity, since a stationary time series is assumed to be free of any temporal structure before modeling. We briefly recall below some relevant definitions:

- level: the mean average value in the series.

- seasonality: the repeated cycles in a series.

- trend: the increasing or decreasing pattern of a series.

- residual: the random variation in a series [11, p. 105].

From the line plot of weekly, biweekly and monthly data sets in Figs 3.5(a),3.5(b),3.5(c) respectively one can clearly see that the time series has an increasing followed by decreasing trends and thus is non-linear. Seasonal decomposition can be done using two models: either additive or multiplicative, depending on how the components of the time series are related. An additive model is linear where changes over time are consistently made by the same amount. A linear seasonality has the same frequency (width of cycles) and amplitude (height of cycles), which is not suitable for our dataset. In a multiplicative model the components are multiplied together as such:

$$y(t) = Level \times Trend \times Seasonality \times Noise$$

A multiplicative model is nonlinear, typically quadratic or exponential. A nonlinear seasonality has an increasing or decreasing frequency and/or amplitude over time [11], which in contrast, is suitable to seasonally decompose our time series.

Below is the seasonal decomposition plots for our data sets. Note that we need to specify the frequency used for the seasonal decomposition. Frequency indicates the periodic number of demands per unit time. For example, if we want to check the seasonality per month using weekly data, we take frequency to be equal to four.

- Weekly case:

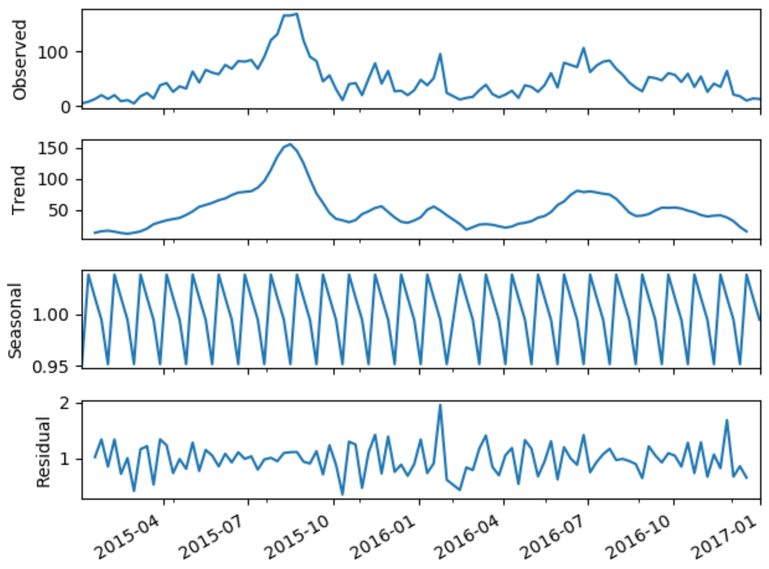


Figure 3.10: Seasonal decomposition of weekly data with frequency = 4

- Biweekly case:

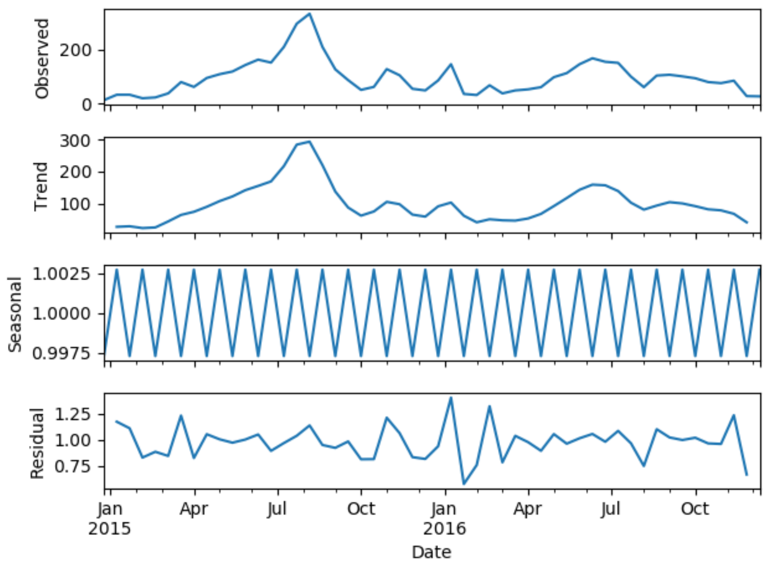


Figure 3.11: Seasonal decomposition of biweekly data with $freq$=2 (2 bi-weeks= 1 month)

- Monthly case: in this case the $freq$=1 showed a straight line at 1 for seasonality and residual plots as in figure 3.12)

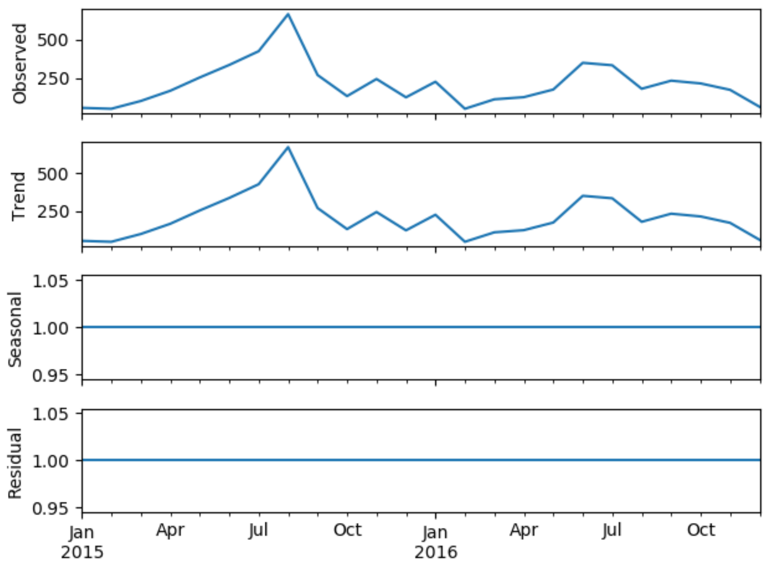


Figure 3.12: Seasonal decomposition of monthly data with $freq$=1

so we tried $freq$=3 to check seasonality of $freq$= 3 months (figure 3.13). This means we are decomposing the series using seasonality of three months in the plot below:

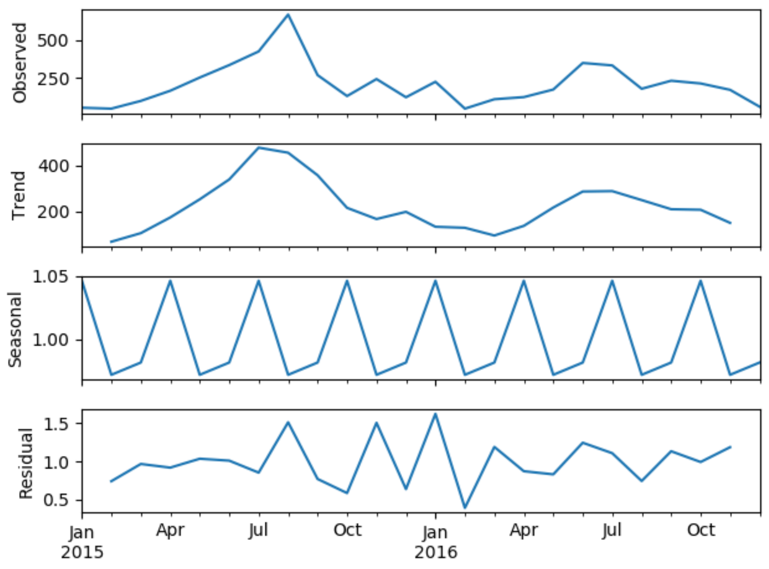


Figure 3.13: Seasonal decomposition of monthly data with $freq$=3(3 months= 1 season(fall,winter,spring,summer))

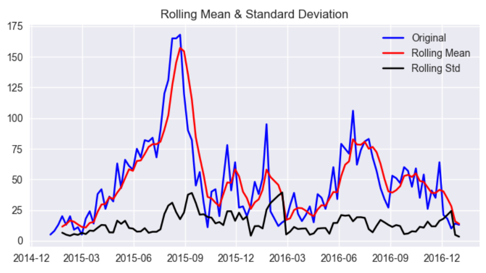
For the above plots:

- The “Observed” plot in each case shows the line plot of original data.
- Concerning trend: we can see variable trend among the two years (no monotonically increasing or decreasing trends) in all cases but the overall trend is similar and hence can be detected and taken out as a feature for later purposes (multivariate part).
- Concerning seasonality: in our case we chose to check seasonality per month or per season(3 months), we discovered that seasonality is very weak in our data, this is indicated by the very tight range of seasonal signals appearing in the figures.

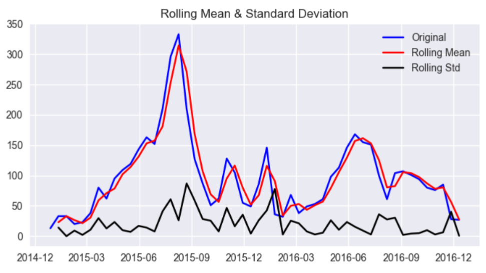
- Concerning residuals: The residuals (which are the time series after the trend and seasonal components are removed) seem reasonable since they show periods variability in the early and later unit times of the series. It is important to see that residuals show random variations with no clear temporal structure since these also will be used later in correcting forecasts of ARIMA model.

3.6 Stationarity of Time Series

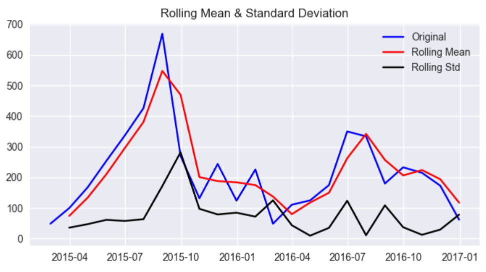
In the previous section we observed seasonality in our time series. As a result, we will need to ensure whether our time series is stationary and thus amenable to predictive modeling. Moving (rolling) averages are a common type of smoothing applied to time series to remove the fine-grained variation between time steps. Visualizing the rolling mean and standard deviation of a data set is important to know if it's stationary by revealing whether the mean is changing over time and how far other values are away from the mean. A Stationary time series will have almost constant rolling average and standard deviation [29]. Below are the plots of the rolling mean and standard deviation in addition to original line plot of each time series. The rolling mean shows high variations across the two years indicating non-stationarity in the series in all cases.



(a) weekly data



(b) biweekly data



(c) monthly data

Figure 3.14: Rolling Mean and Standard deviation

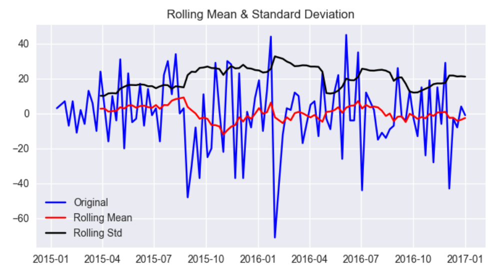
Several steps can be applied to make a time series completely stationary [29]:

- Apply first differencing to remove trend.

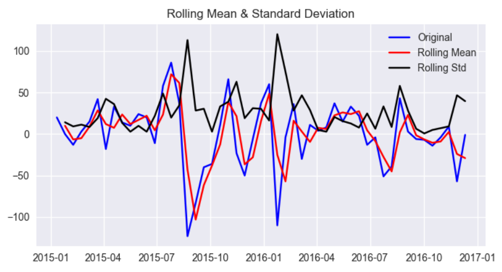
- Apply seasonal differencing to remove seasonality

3.6.1 First Differencing and the Non-Seasonal Integration Parameter d

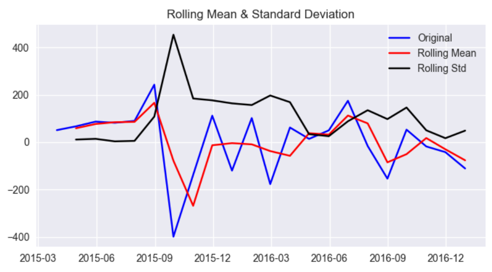
Taking first difference is important since it helps eliminate the overall trend in the time series which is one main cause of non-stationarity. As we go through this we would be checking whether we need to set the non-seasonal integration parameter d of ARIMA and SARIMA to 1 (indicating a one step of differencing) or zero (none), before the time series approaches being stationary. The graphical results following first differencing are shown in Figs. 3.15(a),3.15(b),3.15(c) of weekly, biweekly and monthly cases respectively.



(a) weekly data



(b) biweekly data

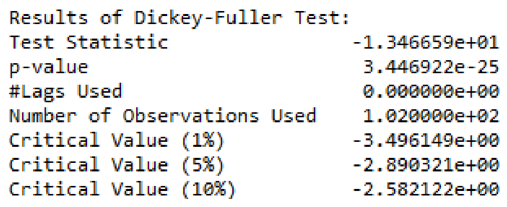


(c) monthly data

Figure 3.15: Rolling Mean and Standard deviation after doing first difference

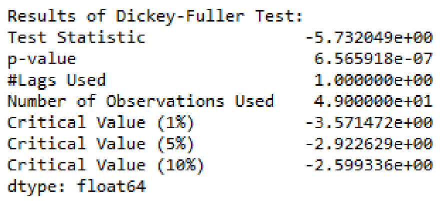
Notice for weekly data set the series looks stationary, given that the rolling mean becomes almost constant. On the contrary, for the biweekly and monthly cases we're not sure how did the difference perform on the stationarity of the series, we can still see some variations in the rolling mean. Therefore we will look at another method that tests stationarity of a time series that is the Augmented Dickey-Fuller (ADF) hypothesis test. This is one of the statistical tests for checking stationarity. First we consider the null hypothesis: the time series is non-stationary. The results of the test will contain the test statistic and critical values for different confidence levels. The idea is to have test statistics less than critical values, in this case we can reject the null hypothesis and say that this time series is indeed stationary.

The results of Figs.3.16(a),3.16(b),3.16(c) show the ADF statistics for each case after applying first difference:



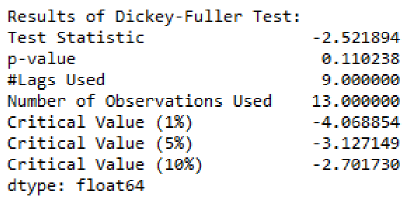
```
Results of Dickey-Fuller Test:
Test Statistic                 -1.346659e+01
p-value                         3.446922e-25
#Lags Used                      0.000000e+00
Number of Observations Used     1.020000e+02
Critical Value (1%)            -3.496149e+00
Critical Value (5%)            -2.890321e+00
Critical Value (10%)           -2.582122e+00
```

(a) weekly data



```
Results of Dickey-Fuller Test:
Test Statistic                 -5.732049e+00
p-value                         6.565918e-07
#Lags Used                      1.000000e+00
Number of Observations Used     4.900000e+01
Critical Value (1%)            -3.571472e+00
Critical Value (5%)            -2.922629e+00
Critical Value (10%)           -2.599336e+00
dtype: float64
```

(b) biweekly data



```
Results of Dickey-Fuller Test:
Test Statistic                 -2.521894
p-value                         0.110238
#Lags Used                      9.000000
Number of Observations Used    13.000000
Critical Value (1%)            -4.068854
Critical Value (5%)            -3.127149
Critical Value (10%)           -2.701730
dtype: float64
```

(c) monthly data

Figure 3.16: ADF statistics after first difference

- Weekly case: Here the test statistic is less than all critical values (more negative) and the p-value is negligible (very close to zero). Hence, series is confirmed as stationary.

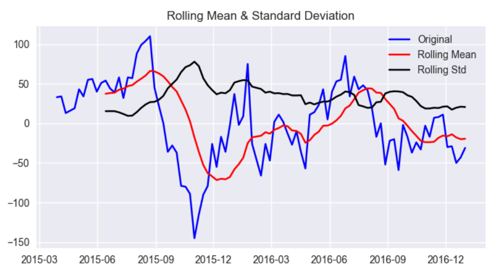
- Biweekly case: Also here we can see that the test statistic is less than all critical values and the p-value almost negligible. Therefore, in this case we achieved stationarity.

- Monthly case: On the other hand, the test statistic in the monthly case is not less then any critical value and the p-value is 0.1 not negligible , thus the first difference did not achieve stationarity of the monthly series.

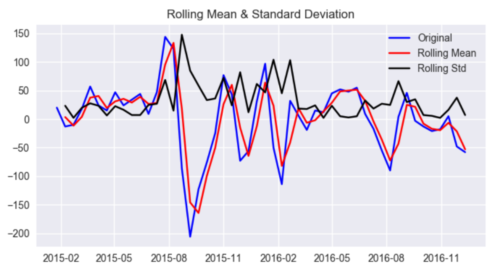
To sum it up, we can suggest a d parameter $=1$ for weekly, and bi-weekly data sets but d=0 for monthly data for both ARIMA and SARIMA models.

3.6.2 Seasonal Differencing and the Seasonal Integration Parameter D

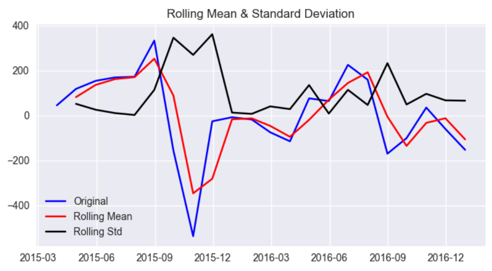
Another important method for ensuring stationarity is by applying a seasonal difference which eliminates the seasonality factor from the time series. As we go through this we would be determining the value of seasonal Integration parameter D (of SARIMA model), that is how many times a series has to be seasonally differenced. Figs. 3.17(a),3.17(b),3.17(c) reveals the results after we apply seasonal difference for several frequencies.



(a) weekly data



(b) biweekly data



(c) monthly data

Figure 3.17: Rolling mean and standard deviation after seasonal difference

```
Results of Dickey-Fuller Test:
Test Statistic                 -2.212180
p-value                         0.201855
#Lags Used                     12.000000
Number of Observations Used    87.000000
Critical Value (1%)            -3.507853
Critical Value (5%)            -2.895382
Critical Value (10%)           -2.584824
dtype: float64
```

(a) weekly data

```
Results of Dickey-Fuller Test:
Test Statistic                 -1.948631
p-value                         0.309541
#Lags Used                      8.000000
Number of Observations Used    41.000000
Critical Value (1%)            -3.600983
Critical Value (5%)            -2.935135
Critical Value (10%)           -2.605963
dtype: float64
```

(b) biweekly data

```
Results of Dickey-Fuller Test:
Test Statistic                 -2.980725
p-value                         0.036734
#Lags Used                      8.000000
Number of Observations Used    13.000000
Critical Value (1%)            -4.068854
Critical Value (5%)            -3.127149
Critical Value (10%)           -2.701730
dtype: float64
```

(c) monthly data

Figure 3.18: ADF statistics after seasonal difference

- Weekly case: In Fig. 3.17(a) the rolling mean suffers from large variations indicating at some places very far values from the mean. Fig. 3.18(a) shows the ADF statistic value which is greater than all critical values, thus failing to reject the null hypothesis that time series is non-stationary. We thus set the seasonal integration parameter for the SARIMA model to be $D = 0$.

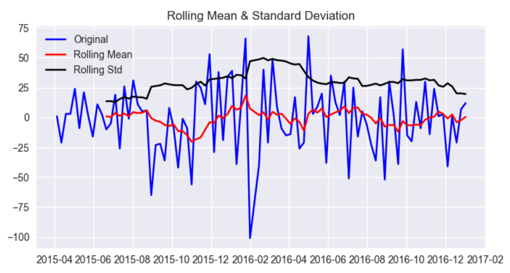
- Biweekly case: Similar to weekly case, in Fig. 3.17(b) the rolling mean suffers from large variations indicating at some places very far values from the mean. Fig. 3.18(b) shows the ADF statistic value which is greater than all critical values, thus failing to reject the null hypothesis that time series is non-stationary. We thus set the seasonal integration parameter for the SARIMA model to be $D = 0$.

- Monthly case: Regarding monthly data, things get better when we apply seasonal difference, fig.3.17(c) shows the the rolling mean and std sow less variations but still not constant. Fig.3.18(c) shows that the ADF statistic value is now more negative than before, it shows that we're only 90% confident that our series is stationary, this confirms the plot of rolling mean and rolling std which showed we still see some variations. Therefore, for the monthly case we may suggest adding D =1 to the SARIMA model.

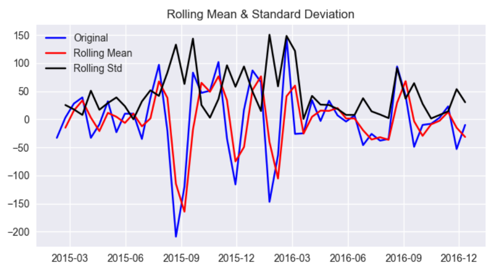
To sum it up, concerning the D parameter of SARIMA, the monthly data showed good results when applying seasonal differencing and hence we can add D=1 to the model in this case. However concerning weekly and bi-weekly data seasonal differencing showed worse results regarding stationarity of the data and hence $D = 0$ is suggested for both cases.

3.6.3 Taking Seasonal Difference of First Differenced Time Series, Assigning d=1 and D=1 in SARIMA Model

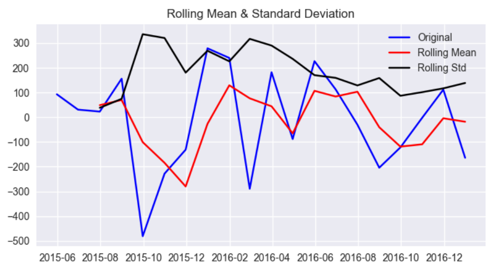
We will now explore whether a combination of non-seasonal and seasonal integration parameters (d and D respectively) shows a better performance in the SARIMA model. This is achieved by applying seasonal differencing on a first differenced series; i.e. setting both d and D parameters to 1. Our results are shown in Figures 3.19(a) and 3.20(a).



(a) weekly data

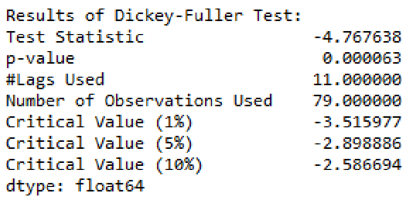


(b) biweekly data



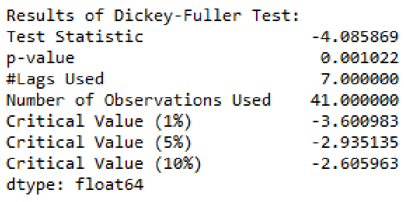
(c) monthly data

Figure 3.19: Rolling mean and std after applying seasonal difference on first differenced data



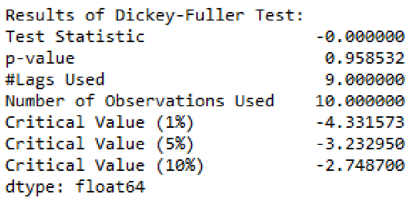
```
Results of Dickey-Fuller Test:
Test Statistic                 -4.767638
p-value                         0.000063
#Lags Used                     11.000000
Number of Observations Used    79.000000
Critical Value (1%)            -3.515977
Critical Value (5%)            -2.898886
Critical Value (10%)           -2.586694
dtype: float64
```

(a) weekly data



```
Results of Dickey-Fuller Test:
Test Statistic                 -4.085869
p-value                         0.001022
#Lags Used                      7.000000
Number of Observations Used    41.000000
Critical Value (1%)            -3.600983
Critical Value (5%)            -2.935135
Critical Value (10%)           -2.605963
dtype: float64
```

(b) biweekly data



```
Results of Dickey-Fuller Test:
Test Statistic                 -0.000000
p-value                         0.958532
#Lags Used                      9.000000
Number of Observations Used    10.000000
Critical Value (1%)            -4.331573
Critical Value (5%)            -3.232950
Critical Value (10%)           -2.748700
dtype: float64
```

(c) monthly data

Figure 3.20: ADF statistics after applying seasonal difference on first differenced data

- Weekly Data: Notice in this case figs. 3.19(a),3.20(a) that the data is stationary according to ADF statistics and the rolling mean and std show small variations and almost constant level. However, contrasting Fig. 3.16(a) against Fig. 3.20(a) we notice that the stationarity results were better when only the first differencing was applied. There, the test statistics were smaller and the *p*-value closer to zero. Again, this suggests that we need to empirically explore the following two options:
  - Assign d=1 and D=0 to the SARIMA model.
  - Assign d=1 and D=1 to the SARIMA model.
- Biweekly Data:

  Finally the biweekly data showed stationarity when both types of differencing were applied to it. The test statistic is less than all critical values and the *p*-value is relatively low. However note that the stationarity results were better when we applied only first difference (figure 3.16(a)) in section 3.6.1 where the test statistics were more negative and the *p*-value closer to zero. (figure 3.20(b)) Again this suggests that we need to try both cases when making our model:
  - Assign d=1 and D=0 to SARIMA model
  - Assign d=1 and D=1 to SARIMA model
- Monthly Data:

  In case of monthly data, applying a seasonal difference to the differenced series had no effect on the stationarity of the differenced series, the test statistic remained zero and relatively high p-value(figure 3.20(c)). Therefore, this suggests that d=0 and D=1 is a good choice for SARIMA integration parameters referring to section 3.6.2.

3.7 Summary

As a summary for this chapter, we noticed several important characteristics of our data:

- First the data was gathered from instant tweets made by red cross and civil defense so it is a real time problem.
- The data was reshaped as time series data to be used in univariate time series forecasting using stochastic models ARIMA and SARIMA.
- Walk forward validation will be used for evaluating models performance on training data.
- RMSE and MAPE measures will be reported for evaluating models general performance.
- Data visualizations showed that the data has no clear overall trend and few seasonality signals. It also showed that weekly and biweekly time series observations are autocorrelated that is they are related to their lag values.
- The seasonal decomposition of each time series confirmed the existence of seasonality.
- The stationarity tests revealed that differencing makes our data more stationary that is more ready for modeling.

Chapter 4

ARIMA Modeling

In this section we will apply the ARIMA model on our data sets, evaluate it, attempt to optimize its performance by tuning its parameters, and finally, apply necessary power transforms to the data if needed.

4.1 Configuring ARIMA Model Manually

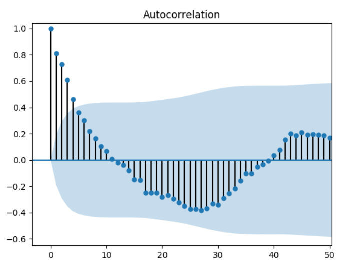
There are three parameters that ought to be configured in order to apply ARIMA model on our data.

- p: (Auto-regression) AR parameter can be configured using the autocorrelation function plot (ACF), which indicates the number of lag observations affecting the model.

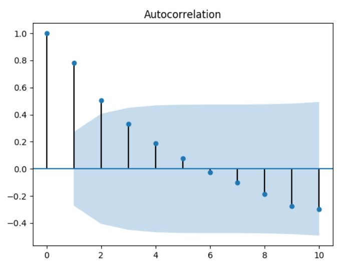
- d: Integration parameter that indicates the number of times we will difference the time series in order to make it stationary.

- q: (Moving average) MA parameter, which can be configured using the partial autocorrelation function (PACF), indicating the size of the moving average window.

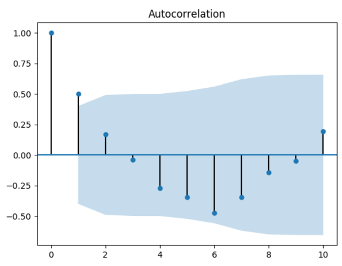
Configuring the AR parameter p As we defined earlier the p or auto-regression parameter indicates the number of lag observations required by the model, which can be inferred from the autocorrelation plot. Below is the figure zooming into the 2D plot of the lag value along the x-axis and the correlation on the y-axis. The cone designates the confidence interval, set by default at 95%, suggesting that correlation values outside of this cone are significant.



(a) weekly data



(b) biweekly data



(c) monthly data

Figure 4.1: ACF of data

- Weekly Data:

  From Fig. 4.1(a) we observe a positive correlation for the first ten lags with only the first five lags being of significant values. This indicates that a range of p between 0 and 4 would be good for testing.

- Biweekly Data:

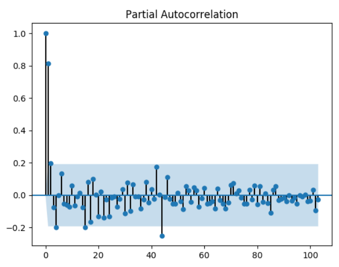
  From Fig. 4.1(b) we observe a positive correlation for the first six lags with only the first three lags being of significant values. This indicates that a range of p between 0 and 2 would be good for testing.

- Monthly Data:

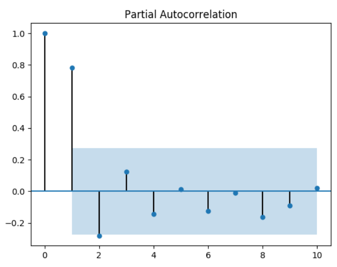
  From Fig. 4.1(c) we observe a positive correlation for the first three lags with only the first two lags being of significant values. This indicates that a range of p between 0 and 1 would be good for testing.

Configuring the Integration Parameter d From Sec. 3.6.1, the stationarity tests appearing in Figs. 3.16(a), 3.16(b), 3.16(c) after applying first difference on time series suggests that the d parameter can be assigned to 1 in case of weekly and biweekly data and 0 in case of monthly data. Therefore we will try both values in any case 0,1 for the d parameter of ARIMA.

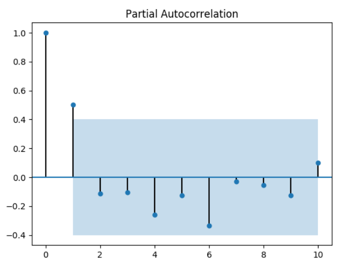
Configuring the MA parameter q Recall that q is the moving average width, which can be inferred by looking at the partial autocorrelation plots. Partial autocorrelation seeks to remove the indirect correlations between an observation and its lags showing in the ACF plot. The plots below show the PACF zoomed in for the first few observations in each case.



(a) weekly data



(b) biweekly data



(c) monthly data

Figure 4.2: PACF of data

- Weekly case:

  The PACF fig. 4.2(a) shows a significant lag for about the first two weeks, so a q value between 0 and 2 could be studied.

- Biweekly case: The PACF fig. 4.2(b) shows a significant lag for about the first two bi-weeks, so a q value between 0 and 2 could be studied.

- Monthly case: The PACF fig. 4.2(c) shows a significant lag for about the first month, so a q value between 0 and 1 could be studied.

4.2 Grid Search for ARIMA hyper-parameters

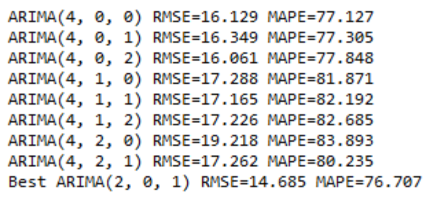
In this section we will search for the best combination of hyper parameters (p, d, q) which results in the least RMSE. The grid search for ARIMA examines all possible combinations for certain given ranges for each of p, d, and q, and returns the one with least RMSE as the best model. From the above configurations, one can consider the following range of values requiring a total of $5 * 3 * 3 = 45$ search coordinates in the grid.

- For p value: 0 to 4

- For q value: 0 to 2

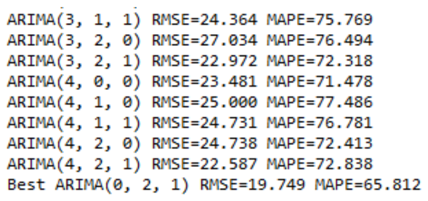
- For d value: 0 or 2

Below is a sample of the results showing the combination with the best RMSE and MAPE values following the grid search on each dataset, with 76% training rows for weekly, 86% for biweekly and 84% for monthly:



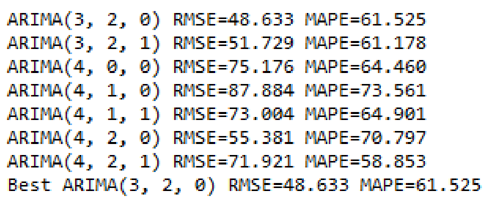
```
ARIMA(4, 0, 0) RMSE=16.129 MAPE=77.127
ARIMA(4, 0, 1) RMSE=16.349 MAPE=77.305
ARIMA(4, 0, 2) RMSE=16.061 MAPE=77.848
ARIMA(4, 1, 0) RMSE=17.288 MAPE=81.871
ARIMA(4, 1, 1) RMSE=17.165 MAPE=82.192
ARIMA(4, 1, 2) RMSE=17.226 MAPE=82.685
ARIMA(4, 2, 0) RMSE=19.218 MAPE=83.893
ARIMA(4, 2, 1) RMSE=17.262 MAPE=80.235
Best ARIMA(2, 0, 1) RMSE=14.685 MAPE=76.707
```

(a) weekly data



```
ARIMA(3, 1, 1) RMSE=24.364 MAPE=75.769
ARIMA(3, 2, 0) RMSE=27.034 MAPE=76.494
ARIMA(3, 2, 1) RMSE=22.972 MAPE=72.318
ARIMA(4, 0, 0) RMSE=23.481 MAPE=71.478
ARIMA(4, 1, 0) RMSE=25.000 MAPE=77.486
ARIMA(4, 1, 1) RMSE=24.731 MAPE=76.781
ARIMA(4, 2, 0) RMSE=24.738 MAPE=72.413
ARIMA(4, 2, 1) RMSE=22.587 MAPE=72.838
Best ARIMA(0, 2, 1) RMSE=19.749 MAPE=65.812
```

(b) biweekly data



```
ARIMA(3, 2, 0) RMSE=48.633 MAPE=61.525
ARIMA(3, 2, 1) RMSE=51.729 MAPE=61.178
ARIMA(4, 0, 0) RMSE=75.176 MAPE=64.460
ARIMA(4, 1, 0) RMSE=87.884 MAPE=73.561
ARIMA(4, 1, 1) RMSE=73.004 MAPE=64.901
ARIMA(4, 2, 0) RMSE=55.381 MAPE=70.797
ARIMA(4, 2, 1) RMSE=71.921 MAPE=58.853
Best ARIMA(3, 2, 0) RMSE=48.633 MAPE=61.525
```

(c) monthly data

Figure 4.3: Best ARIMA produced by grid search

- Weekly:

  The best ARIMA model appears to be confirmed for $p = 2$, $d = 0$, and $q = 1$ for the case of weekly data.

- Biweekly:

  The best ARIMA model appears to be confirmed for $p = 0$, $d = 2$, and $q = 1$ for the case of biweekly data.

- Monthly:

  The best ARIMA model appears to be confirmed for $p = 3$, $d = 2$, and $q = 0$ for the case of monthly data.

Notice that in all cases the values fall in the ranges we configured earlier in the previous section, although we can see that the d parameter showed interesting results. It showed that a better performance of the model was achieved when no differencing was applied to weekly data, and when biweekly and monthly data were differenced twice.

The table below summarizes the results of best ARIMA models associated with RMSE values:

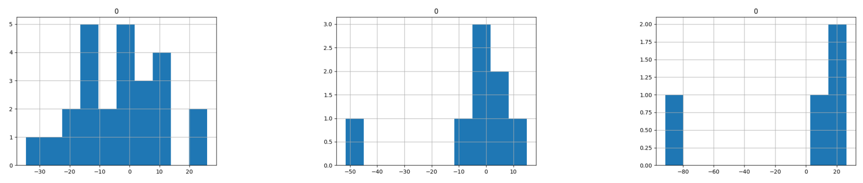
Case	ARIMA model	RMSE	MAPE
Weekly Data	(2,0,1)	14.685	76.707
Biweekly Data	(0,2,1)	19.749	65.812
Monthly Data	(3,2,0)	48.633	61.525

Table 4.1: RMSE and MAPE results of chosen ARIMA models

Next we will study residual errors, and try to use them for correcting the forecasts.

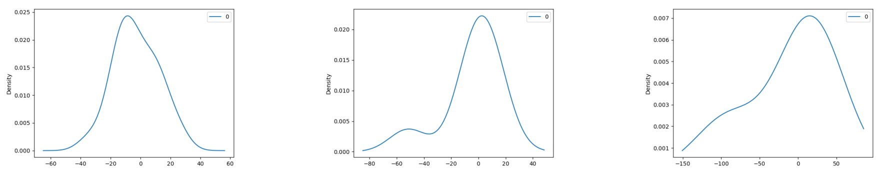
4.3 Residual Errors

In order to investigate whether the suggested model can be improved further one needs to examine its residual errors. If those errors exhibit any temporal structure, this indicates that the model can be improved, the errors should show a white noise behavior in their idealized form. Below are the distribution plots of residual errors resulting from the ARIMA models done for each case (weekly,biweekly, monthly):



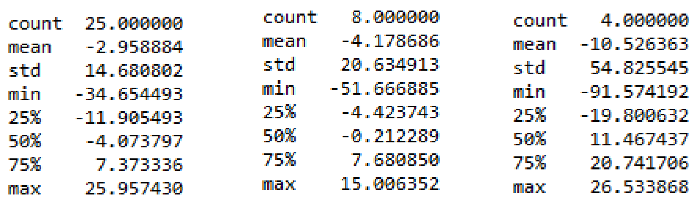
(a) weekly data (b) biweekly data (c) monthly data

Figure 4.4: Histogram plot of ARIMA residual errors



(a) weekly data (b) biweekly data (c) monthly data

Figure 4.5: Density plot of ARIMA residual errors



```
count    25.000000
mean     -2.958884
std      14.680802
min     -34.654493
25%     -11.905493
50%      -4.073797
75%       7.373336
max      25.957430
```

```
count     8.000000
mean     -4.178686
std      20.634913
min     -51.666885
25%      -4.423743
50%      -0.212289
75%       7.680850
max      15.006352
```

```
count     4.000000
mean    -10.526363
std      54.825545
min     -91.574192
25%     -19.800632
50%      11.467437
75%      20.741706
max      26.533868
```

(a) weekly data (b) biweekly data (c) monthly data

Figure 4.6: Statistics of ARIMA residual errors

- Weekly case:

  The weekly residual distribution plots show a near Gaussian distribution with a minor skew to the left. We continue by examining the summary statistics of residual errors. It shows a mean of -2.958, which indicates the presence of bias, preventing the distribution of the residuals to be an idealized Gaussian, and hence here we need to deal with this bias.

- Biweekly case:

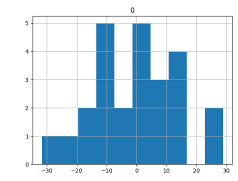
  The biweekly residual distribution plots show a distribution with a skew to the left, it is close to double Gaussian. We continue by examining the summary statistics of residual errors. It shows a mean of -4.178, which is not very close to zero, preventing the distribution of the residuals to be an idealized Gaussian, and hence here we also need to deal with bias.

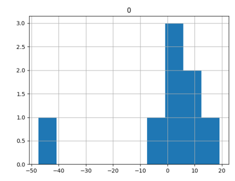
- Monthly case:

  The monthly residual plot shows a distribution with a skew to the left, again it is not close to Gaussian. We continue by examining the summary statistics of residual errors. It shows a mean of -10.526, which is far from zero, preventing the distribution of the residuals to be an idealized Gaussian, and hence here we also need to deal with bias.

Correcting Predictions with Forecast Errors Here we will try to improve the ARIMA results by applying bias correction to the prediction. We do this by adding the mean residual error to each forecast made. The plot and summary statistics below of the new residual errors show that the ARIMA performance improves slightly (look at RMSE values in figures

4.9(a),4.9(b), 4.9(c)). The mean appears to be very close to zero after the bias correction. The results are shown below for each case:


(a) weekly data


(b) biweekly data

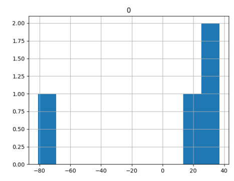
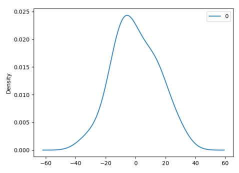
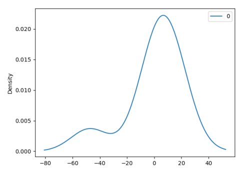

(c) monthly data

Figure 4.7: Histogram of corrected residuals


(a) weekly data


(b) biweekly data

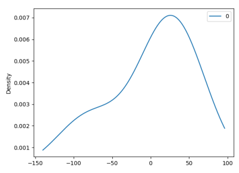
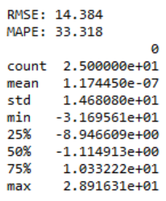
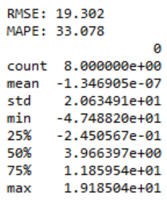

(c) monthly data

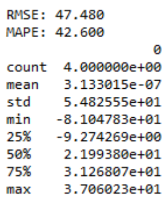
Figure 4.8: Density plots of corrected residual errors


```
RMSE: 14.384
MAPE: 33.318
                     0
count  2.500000e+01
mean   1.174450e-07
std    1.468080e+01
min   -3.169561e+01
25%   -8.946609e+00
50%   -1.114913e+00
75%    1.033222e+01
max    2.891631e+01
```

(a) weekly data


```
RMSE: 19.302
MAPE: 33.078
                     0
count  8.000000e+00
mean  -1.346905e-07
std    2.063491e+01
min   -4.748820e+01
25%   -2.450567e-01
50%    3.966397e+00
75%    1.185954e+01
max    1.918504e+01
```

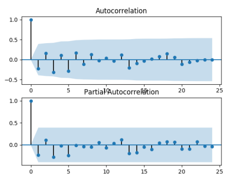
(b) biweekly data


```
RMSE: 47.480
MAPE: 42.600
                     0
count  4.000000e+00
mean   3.133015e-07
std    5.482555e+01
min   -8.104783e+01
25%   -9.274269e+00
50%    2.199380e+01
75%    3.126807e+01
max    3.706023e+01
```

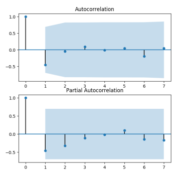
(c) monthly data

Figure 4.9: RMSE and MAPE in addition to Statistics of corrected residual errors

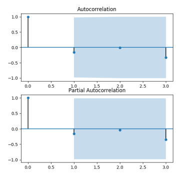
We now check the ACF and PACF of the residual errors to make sure we have no significant temporal structure in the data sets. We can see from the plots that there are only very few positive and negative correlations and that are with no significance. This indicates white noise residuals that exhibit no significant temporal structure.



(a) weekly data



(b) biweekly data



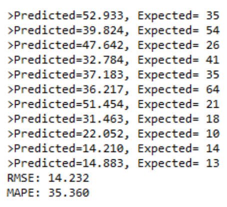
(c) monthly data

Figure 4.10: ACF and PACF of corrected monthly residual errors

4.4 Improving Results of ARIMA by Applying Box-cox Power Transforms

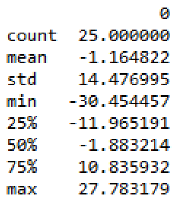
The Box-Cox data transform method supports both the square root and log transform, as well as a batch of associated transforms. It evaluates a collection of these transforms before selecting the best fit. The Box-Cox transform aims at removing power-based trends in the time series in order to render it more linear. In this section we will apply the box cox transform on the time series and we will re-apply the ARIMA model on the transformed data to check for improvements. Finally we will invert the box cox transform back to preserve the data.

- Weekly Case:



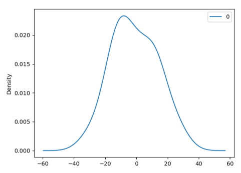
```
>Predicted=52.933, Expected= 35
>Predicted=39.824, Expected= 54
>Predicted=47.642, Expected= 26
>Predicted=32.784, Expected= 41
>Predicted=37.183, Expected= 35
>Predicted=36.217, Expected= 64
>Predicted=51.454, Expected= 21
>Predicted=31.463, Expected= 18
>Predicted=22.052, Expected= 10
>Predicted=14.210, Expected= 14
>Predicted=14.883, Expected= 13
RMSE: 14.232
MAPE: 35.360
```

(a) predictions and error measures

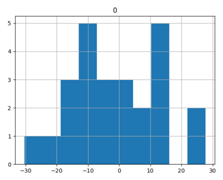


```
                0
count   25.000000
mean    -1.164822
std     14.476995
min    -30.454457
25%    -11.965191
50%     -1.883214
75%     10.835932
max     27.783179
```

(b) statistics



(c) histogram



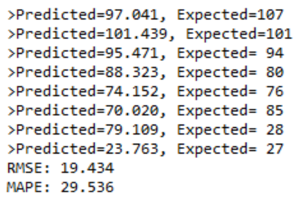
(d) density

Figure 4.11: Weekly data plots after box cox transform

The results shown above improves only slightly on the prediction accuracies reported prior to applying the transform, with a new RMSE at 14.232 as opposed to 14.685. We continue to study the residual errors of ARIMA applied on box cox transformed data:
From Figures 4.11(b), 4.11(c), and 4.11(d), we notice several things:

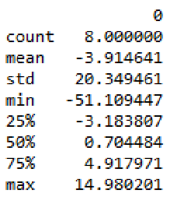
- The mean of the forecast residual errors is not zero.
- The residual errors don't show a Gaussian distribution.
- There is some bias that need to be corrected.

- Biweekly Case:



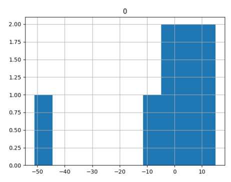
```
>Predicted=97.041, Expected=107
>Predicted=101.439, Expected=101
>Predicted=95.471, Expected= 94
>Predicted=88.323, Expected= 80
>Predicted=74.152, Expected= 76
>Predicted=70.020, Expected= 85
>Predicted=79.109, Expected= 28
>Predicted=23.763, Expected= 27
RMSE: 19.434
MAPE: 29.536
```

(a) predictions and error measures

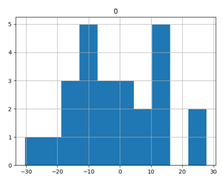


```
                0
count    8.000000
mean    -3.914641
std     20.349461
min    -51.109447
25%     -3.183807
50%      0.704484
75%      4.917971
max     14.980201
```

(b) statistics



(c) histogram



(d) density

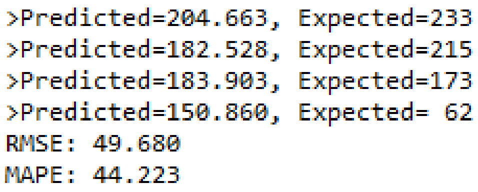
Figure 4.12: Bi-Weekly data plots after box cox transform

The results shown above improves only slightly on the prediction accuracies reported prior to applying the transform, with a new RMSE at 19.434 as opposed to 19.749. We continue to study the residual errors of ARIMA applied on box cox transformed data:

From Figures 4.12(b), 4.12(c), and 4.12(d), we notice several things:

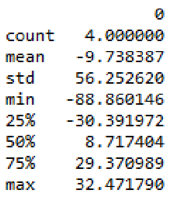
- The mean of the forecast residual errors is not zero.
- The residual errors don't show a Gaussian distribution.
- There is some bias that need to be corrected.

- Monthly Case:



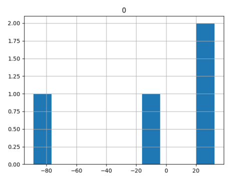
```
>Predicted=204.663, Expected=233
>Predicted=182.528, Expected=215
>Predicted=183.903, Expected=173
>Predicted=150.860, Expected= 62
RMSE: 49.680
MAPE: 44.223
```

(a) predictions and error measures

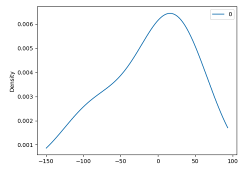


```
               0
count   4.000000
mean   -9.738387
std    56.252620
min   -88.860146
25%   -30.391972
50%     8.717404
75%    29.370989
max    32.471790
```

(b) statistics



(c) histogram



(d) density

Figure 4.13: Monthly data plots after box cox transform

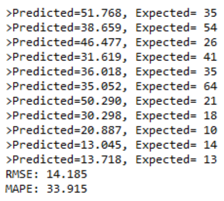
The results shown above do not improve on the prediction accuracies reported prior to applying the transform, with a new RMSE at 49.680 as opposed to 48.633 that is a worse result than before transform. We continue to study the residual errors of ARIMA applied on box cox transformed data:

From Figures 4.13(b), 4.13(c), and 4.13(d), we notice several things:

- The mean of the forecast residual errors is not zero.
- The residual errors don't show a Gaussian distribution.
- There is some bias that need to be corrected.

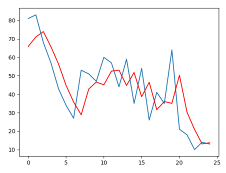
Similar to what we did earlier for bias correction, we will add the mean of residual errors to each forecast in order to apply bias correction. ARIMA predictions and their plots, along with the statistic and distribution plots of the corrected residual errors obtained from corrected box cox transformed data are shown below:

- Weekly Case:

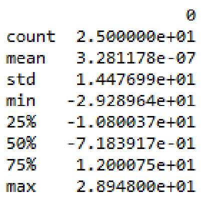


```
>Predicted=51.768, Expected= 35
>Predicted=38.659, Expected= 54
>Predicted=46.477, Expected= 26
>Predicted=31.619, Expected= 41
>Predicted=36.018, Expected= 35
>Predicted=35.052, Expected= 64
>Predicted=50.290, Expected= 21
>Predicted=30.298, Expected= 18
>Predicted=20.887, Expected= 10
>Predicted=13.045, Expected= 14
>Predicted=13.718, Expected= 13
RMSE: 14.185
MAPE: 33.915
```

(a) predictions and error measures

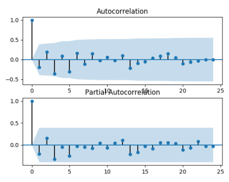


(b) Predictions Line plot



```
                   0
count   2.500000e+01
mean    3.281178e-07
std     1.447699e+01
min    -2.928964e+01
25%    -1.080037e+01
50%    -7.183917e-01
75%     1.200075e+01
max     2.894800e+01
```

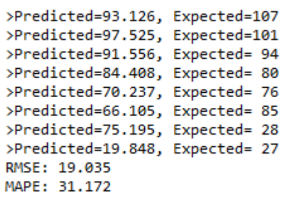
(c) statistics



(d) autocorrelation

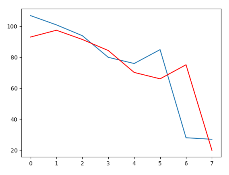
Figure 4.14: ARIMA Weekly Predictions and residual plots after bias correction and box cox transform

- Biweekly Case:

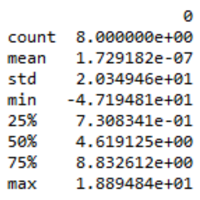


```
>Predicted=93.126, Expected=107
>Predicted=97.525, Expected=101
>Predicted=91.556, Expected= 94
>Predicted=84.408, Expected= 80
>Predicted=70.237, Expected= 76
>Predicted=66.105, Expected= 85
>Predicted=75.195, Expected= 28
>Predicted=19.848, Expected= 27
RMSE: 19.035
MAPE: 31.172
```

(a) predictions and error measures

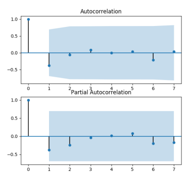


(b) Predictions line plot



```
                 0
count  8.000000e+00
mean   1.729182e-07
std    2.034946e+01
min   -4.719481e+01
25%    7.308341e-01
50%    4.619125e+00
75%    8.832612e+00
max    1.889484e+01
```

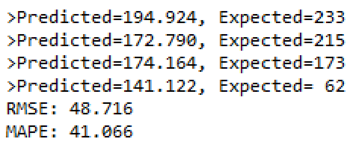
(c) statistics



(d) autocorrelation

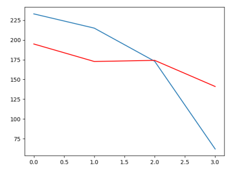
Figure 4.15: ARIMA Bi-Weekly Predictions and residual plots after bias correction and box cox transform

- Monthly Case:

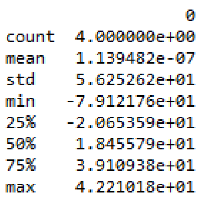


```
>Predicted=194.924, Expected=233
>Predicted=172.790, Expected=215
>Predicted=174.164, Expected=173
>Predicted=141.122, Expected= 62
RMSE: 48.716
MAPE: 41.066
```

(a) predictions and error measures

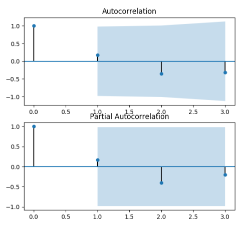


(b) Predictions line plot



```
                 0
count  4.000000e+00
mean   1.139482e-07
std    5.625262e+01
min   -7.912176e+01
25%   -2.065359e+01
50%    1.845579e+01
75%    3.910938e+01
max    4.221018e+01
```

(c) statistics



(d) autocorrelation

Figure 4.16: ARIMA Monthly Predictions and residual plots after bias correction and box cox transform

To sum it up, and since results showed improvement in performance after bias correcting the transformed data, we will choose ARIMA(2,0,1), ARIMA(0,2,1) with box cox transform and bias-correction in case of weekly and biweekly respectively to get better results), and ARIMA(3,2,0) with only bias correction in case of monthly. Below is a summary of the final RMSE measures of chosen ARIMA models in each case compared to persistence model (baseline) to check whether our model is good or not.

Case	P-RMSE	P-MAPE	ARIMA-RMSE	ARIMA-MAPE	Mean
Weekly Data	16.289	37.157	14.185	33.915	48.644
Biweekly Data	21.322	32.102	19.035	31.172	97.288
Monthly Data	65.609	58.607	48.716	41.066	210.791

Table 4.2: RMSE and MAPE results of persistence model and chosen ARIMA model compared to the mean for each data set

The table shows that ARIMA did a good job forecasting the training data since the RMSE and MAPE results beat the persistence model RMSE and MAPE, hence the model beats the baseline model. Notice also that compared to the mean of each data set, the model seems to predict with an error less than the mean which is a reasonable result since we don't want to have an error in predicting demand that is larger than the mean of a certain week or month.. In chapter 6 we will test the chosen ARIMA models on unseen data and compare the model performance accordingly.

Chapter 5

SARIMA Modeling

Time series are usually non-stationary and in order to achieve stationarity, the series has to be differenced by removing the signals (the trend or seasonality) from the series so that it is rendered consisting of only the noise or the irregular component to be modeled. [30]

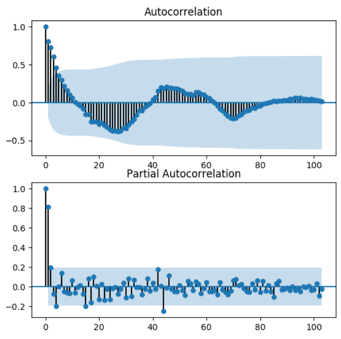
The ARIMA model is for non-seasonal, non-stationary data. Box and Jenkins have generalized this model to deal with seasonality. Their proposed model is known as the Seasonal ARIMA (SARIMA) model. As in the case with ARIMA, seasonal differencing of the appropriate order is used to remove non-stationarity from the series. However, extra parameters extending the ARIMA model are defined as follows. We denote the model by SARIMA(p,d,q)$\times$ $(P, D, Q)_s$ model [11]:

- p: nonseasonal AR parameter that can be configured using the autocorrelation function plot (ACF), indicating the number of lag observations included in the model.
- d: integration parameter indicating the number of times we will difference the time series to make it stationary.
- q: nonseasonal MA parameter that can be configured using the partial autocorrelation function (PACF), indicating the size of the moving average window.
- P: seasonal AR parameter that can be configured using the autocorrelation function plot (ACF) of seasonally differenced time series.
- D: integration parameter indicating the number of times we will apply seasonal difference to the time series to make it stationary.
- Q: seasonal MA parameter that can be configured using the partial autocorrelation function (PACF) of seasonally differenced time series.
- s: the number of observations per period (for example, in the case of monthly time series, $s = 12$, and for quarterly terms time series, $s = 4$).

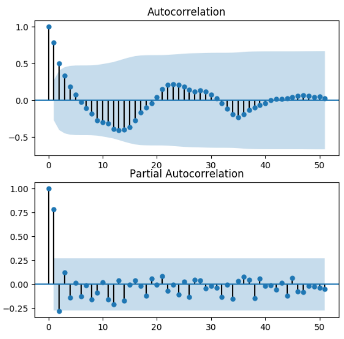
5.1 Parameter tuning

Literature review reveals that there isn't one way to find out optimal parameters for SARIMA model. Choosing these parameters can either be studied in a systematic way or chosen intuitively. In this section we will show both approaches and compare them to find the best model to be used for the rest of our study. It is important to know significant lags in order to have a good intuition on how to choose parameters for our model. In order to know the non-seasonal AR parameters (p) we look at the ACF and the points at which a sharp cut-off occur are what we need to check. For the non-seasonal MA parameters (q) we look at the PACF and consider the point outside the shaded blue part at which a sharp cut-off occurs. In contrast to ARIMA where we only look for only the first few significant lags, SARIMA requires that we extend the search to identify the lags that might show seasonality. The other notable difference is that we'd have to be extracting the seasonal parameters P and Q from the ACF and PACF plots.

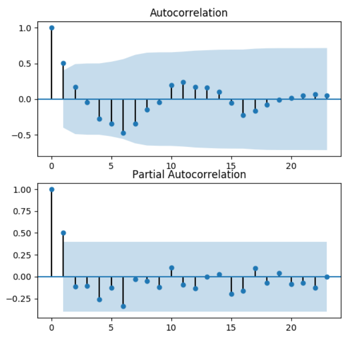
5.2 Studying Non-seasonal Parameters (p and q)



(a) weekly



(b) bi-weekly



(c) monthly

Figure 5.1: ACF and PACF of data

- Weekly Data: By looking at ACF in figure 5.1(a) we can see that the first 5 lags are significant (lags 0 to 4), this shows that we might consider studying p values 0,1,2,3,4. For the q values, the PACF shows several significant values to consider: 0,1,4,15,44.
- Biweekly Data:

  By looking at ACF in figure 5.1(b) we can see that the first 3 lags are significant (lags 0,1,2), this shows that we might consider studying p values 0,1,2. For the q values, the PACF shows also that the first 3 lags are significant so we might consider 0,1,2 for the q values.
- Monthly Data:

  By looking at ACF in figure 5.1(c) we can see that the first 2 lags are significant (lags 0,1), this shows that we might consider studying p values 0,1. For the q values, the PACF shows also that the first 2 lags are significant so we might consider 0,1 for the q values.

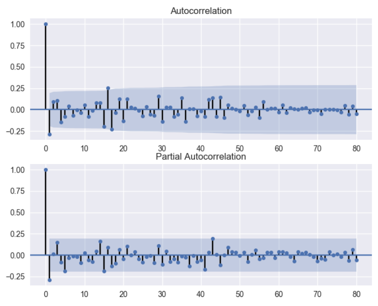
5.3 Studying Seasonal Parameters (P and Q)

At this stage we need to look at the autocorrelation function and partial autocorrelation function plots of the differenced data sets in order to determine the seasonal parameters of the model. Keep in mind that we will consider a period = 1 or 3 months for seasonality. This is based on our intuition that the demand changes either monthly or seasonally (every 3 months). Below are few rules to consider when looking at ACF and PACF and choosing our parameters [31]:

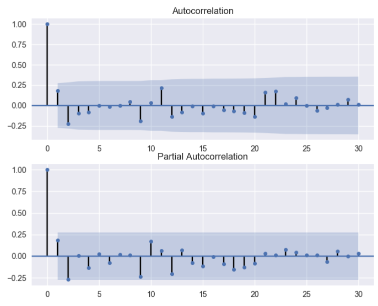
- rule 1: If the autocorrelation of the appropriately differenced series is positive at lag s, where s is the number of periods in a season, then consider adding an P term to the model [31].

- rule 2: If the autocorrelation of the differenced series is negative at lag s, consider adding an Q term to the model [31].
- rule 3: You should try to avoid using more than one or two seasonal parameters in the same model, as this is likely to lead to over-fitting of the data and/or problems in estimation [31].

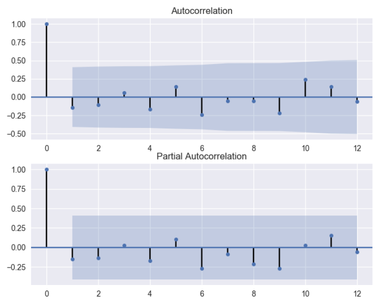
Below are the ACF and PACF plots of the differenced data:



(a) weekly



(b) bi-weekly



(c) monthly

Figure 5.2: ACF and PACF of data

- Weekly Data:

  With respect to our weekly data, we will consider two periods S=4 which means one month and S=12 which shows three months. In both cases, lag S shows negative correlation, even though it is more significant in case of S=4 thus by referring to rules we will try adding P parameter = 1 and Q parameter=0.
- Biweekly Data:

  With respect to the frequency S in this case we will consider it 2 (for one month) or 6(for three months). For S=6, there is very low autocorrelation so we will not add any seasonal parameters for this case. However for frequency 2, PACF shows negative autocorrelation so we will add an SMA parameter=1 to the model according to rules.
- Monthly Data:

  In the monthly case, concerning S values we will consider S=3. (We tried also S=1 but it showed no spikes in the ACF and PACF plot) In both cases, the value of autocorrelation at lag S is not significant hence we might need to try different possibilities to P and Q values.

5.4 Intuitive SARIMA Approach

Based on our exploration in Sections 3.6.1, 3.6.2, 3.6.3, 5.1) and on our intuition that demand for a red cross visit changes per month or per season (three months), we will propose several intuitive SARIMA models to evaluate:

Case	Proposed Model
Weekly Data	({0,1,2,3,4},1,{0,1,4})(0,{0,1},1,{4,12})
Biweekly Data	({0,1,2},1,{0,1,2})(0,{0,1},1,{2,6})
Monthly Data	({0,1},0,{0,1})(0,1,1,3)

Table 5.1: Proposed SARIMA Models to Try

Based on MAPE and RMSE values below are the chosen intuitive SARIMA models to be compared with BIC-based models:

Case	Proposed Model	RMSE	MAPE
Weekly Data	(1,1,0)(0,0,1,12)	15.102	36.064
Biweekly Data	(0,1,0)(0,0,1,6)	21.589	32.916
Monthly Data	(0,1,0)(0,1,1,3)	73.610	61.919

Table 5.2: Chosen Intuitive SARIMA Models

5.5 BIC-based Approach

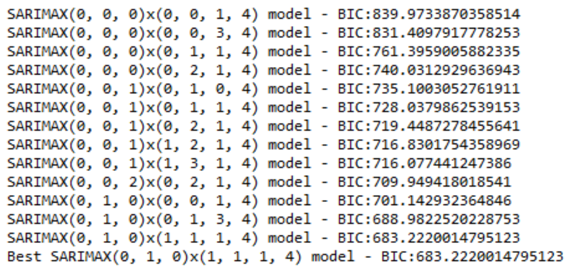
This approach relies on the principle that the model with least Byesian Information Criteria (BIC) is the preferred model to use. BIC is a criterion for model selection among a finite set of models. When fitting models, it is possible to increase the likelihood by adding parameters, but doing so may result in over-fitting. The BIC resolves this problem by introducing a penalty term for the number of parameters in the model [32].

In order to find the combination of parameters that result in the least BIC value, we will try a wide range of AR and MA parameters.

For non-seasonal AR parameter p and MA parameter q we set a range from 0 to 6, we did the same for seasonal parameters P and Q. For non-seasonal difference parameter d and seasonal parameter D we will try a range from 0 to 2. (to avoid over differencing the time series). For period we will consider a seasonality period of 3 months taking into consideration that the demand changes seasonally. Having set our ranges of values we will try all possible combinations to get the preferred model (the one with least BIC value).

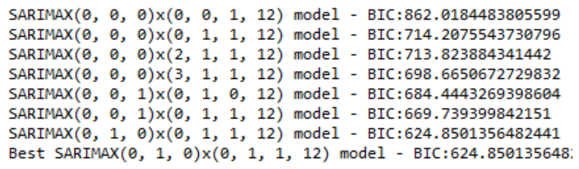
Below are the results obtained for each data set:

- Weekly Case:



```
SARIMAX(0, 0, 0)x(0, 0, 1, 4) model - BIC:839.9733870358514
SARIMAX(0, 0, 0)x(0, 0, 3, 4) model - BIC:831.4097917778253
SARIMAX(0, 0, 0)x(0, 1, 1, 4) model - BIC:761.3959005882335
SARIMAX(0, 0, 0)x(0, 2, 1, 4) model - BIC:740.0312929636943
SARIMAX(0, 0, 1)x(0, 1, 0, 4) model - BIC:735.1003052761911
SARIMAX(0, 0, 1)x(0, 1, 1, 4) model - BIC:728.0379862539153
SARIMAX(0, 0, 1)x(0, 2, 1, 4) model - BIC:719.4487278455641
SARIMAX(0, 0, 1)x(1, 2, 1, 4) model - BIC:716.8301754358969
SARIMAX(0, 0, 1)x(1, 3, 1, 4) model - BIC:716.077441247386
SARIMAX(0, 0, 2)x(0, 2, 1, 4) model - BIC:709.949418018541
SARIMAX(0, 1, 0)x(0, 0, 1, 4) model - BIC:701.142932364846
SARIMAX(0, 1, 0)x(0, 1, 3, 4) model - BIC:688.9822520228753
SARIMAX(0, 1, 0)x(1, 1, 1, 4) model - BIC:683.2220014795123
Best SARIMAX(0, 1, 0)x(1, 1, 1, 4) model - BIC:683.2220014795123
```

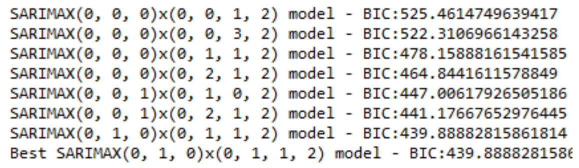
Figure 5.3: Reported BIC values after tuning parameters of weekly case (S=4)



```
SARIMAX(0, 0, 0)x(0, 0, 1, 12) model - BIC:862.0184483805599
SARIMAX(0, 0, 0)x(0, 1, 1, 12) model - BIC:714.2075543730796
SARIMAX(0, 0, 0)x(2, 1, 1, 12) model - BIC:713.823884341442
SARIMAX(0, 0, 0)x(3, 1, 1, 12) model - BIC:698.6650672729832
SARIMAX(0, 0, 1)x(0, 1, 0, 12) model - BIC:684.4443269398604
SARIMAX(0, 0, 1)x(0, 1, 1, 12) model - BIC:669.739399842151
SARIMAX(0, 1, 0)x(0, 1, 1, 12) model - BIC:624.8501356482441
Best SARIMAX(0, 1, 0)x(0, 1, 1, 12) model - BIC:624.850135648
```

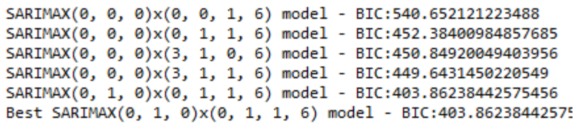
Figure 5.4: Reported BIC values after tuning parameters of weekly case (S=12)

- Biweekly Case:



```
SARIMAX(0, 0, 0)x(0, 0, 1, 2) model - BIC:525.4614749639417
SARIMAX(0, 0, 0)x(0, 0, 3, 2) model - BIC:522.3106966143258
SARIMAX(0, 0, 0)x(0, 1, 1, 2) model - BIC:478.15888161541585
SARIMAX(0, 0, 0)x(0, 2, 1, 2) model - BIC:464.8441611578849
SARIMAX(0, 0, 1)x(0, 1, 0, 2) model - BIC:447.00617926505186
SARIMAX(0, 0, 1)x(0, 2, 1, 2) model - BIC:441.17667652976445
SARIMAX(0, 1, 0)x(0, 1, 1, 2) model - BIC:439.88882815861814
Best SARIMAX(0, 1, 0)x(0, 1, 1, 2) model - BIC:439.88882815
```

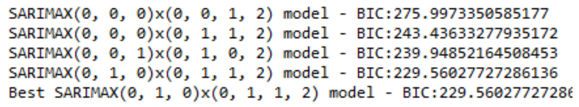
Figure 5.5: Reported BIC values after tuning parameters of biweekly case (S=2)



```
SARIMAX(0, 0, 0)x(0, 0, 1, 6) model - BIC:540.652121223488
SARIMAX(0, 0, 0)x(0, 1, 1, 6) model - BIC:452.38400984857685
SARIMAX(0, 0, 0)x(3, 1, 0, 6) model - BIC:450.84920049403956
SARIMAX(0, 0, 0)x(3, 1, 1, 6) model - BIC:449.6431450220549
SARIMAX(0, 1, 0)x(0, 1, 1, 6) model - BIC:403.86238442575456
Best SARIMAX(0, 1, 0)x(0, 1, 1, 6) model - BIC:403.8623844257
```

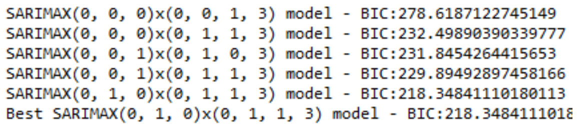
Figure 5.6: Reported BIC values after tuning parameters of biweekly case (S=6)

- Monthly Case:



```
SARIMAX(0, 0, 0)x(0, 0, 1, 2) model - BIC:275.9973350585177
SARIMAX(0, 0, 0)x(0, 1, 1, 2) model - BIC:243.43633277935172
SARIMAX(0, 0, 1)x(0, 1, 0, 2) model - BIC:239.94852164508453
SARIMAX(0, 1, 0)x(0, 1, 1, 2) model - BIC:229.56027727286136
Best SARIMAX(0, 1, 0)x(0, 1, 1, 2) model - BIC:229.5602772728
```

Figure 5.7: Reported BIC values after tuning parameters of monthly case (S=2)



```
SARIMAX(0, 0, 0)x(0, 0, 1, 3) model - BIC:278.6187122745149
SARIMAX(0, 0, 0)x(0, 1, 1, 3) model - BIC:232.49890390339777
SARIMAX(0, 0, 1)x(0, 1, 0, 3) model - BIC:231.8454264415653
SARIMAX(0, 0, 1)x(0, 1, 1, 3) model - BIC:229.89492897458166
SARIMAX(0, 1, 0)x(0, 1, 1, 3) model - BIC:218.34841110180113
Best SARIMAX(0, 1, 0)x(0, 1, 1, 3) model - BIC:218.348411101
```

Figure 5.8: Reported BIC values after tuning parameters of monthly case (S=3)

The results of the BIC-Based SARIMA showing least BIC values are summarized in the table below:

Case	Proposed Model	least BIC value
Weekly Data	(0,1,0)(0,1,1,12)	624.850
Biweekly Data	(0,1,0)(0,1,1,6)	403.862
Monthly Data	(0,1,0)(0,1,1,3)	218.348

Table 5.3: BIC-based SARIMA Models

5.6 Comparison between Intuitive and BIC-based SARIMA models

Recall the table of BIC-based SARIMA models (table 5.3), and the table of chosen intuitive SARIMA models (table 5.2). In this section we will compare the MAPE values of both models for each data set and we will choose the one with least MAPE value to do our predictions.

The table below shows the MAPE values of intuitive model versus BIC-based model for each data set. Note that in SARIMA model the differencing is embedded as parameters in the model in order to model a stationary data set;however, the predictions made and tested are the real predictions and not those on differenced data:

Case	BIC-based	Intuitive
Weekly Data	49.146	36.064
Biweekly Data	50.371	32.916
Monthly Data	61.222	61.919

Table 5.4: MAPE values of BIC-based vs Intuitive SARIMA Models

Notice that in weekly and Biweekly data sets the intuitive models showed better performance (lower MAPE values). In the monthly case the results are close but the BIC model showed better performance.

To sum it up, the below table shows the chosen SARIMA models done on training data with their RMSE values compared to persistence RMSE and mean.

Case	P-RMSE	P-MAPE	SARIMA-RMSE	SARIMA-MAPE	Mean
Weekly Data	16.289	37.157	15.102	36.064	48.644
Biweekly Data	21.322	32.102	21.589	32.916	97.288
Monthly Data	65.609	58.607	72.480	61.222	210.791

Table 5.5: RMSE and MAPE results of persistence model and chosen SARIMA model compared to the mean of each data set

5.7 Why Using ARIMA and SARIMA and not ML Models for Univariate Time Series Forecasting?

One may wonder why to use ARIMA and SARIMA models when their results are not so satisfying. Well when dealing with univariate time series data it seems that the best thing to use is these Box-Jenkins methods. To make sure of that we tried to forecast the univariate time series of weekly data using other machine learning models like support vector machines, multi-layer-perceptron, linear regression, decision trees, random forests.. The results were really bad compared to the RMSE and MAPE results of ARIMA and SARIMA. The table below shows some of the results obtained when applying some machine learning models on weekly univariate time series to do forecasts on training data:

Model	RMSE	MAPE
Linear Regression	154.0154	266.6665
SVM	105.4132	147.9113
MLP	74.2764	168.5576
DT	26.8542	50.5673
RF	21.4583	50.7023
Persistence	16.289	37.157
ARIMA	14.185	33.915
SARIMA	15.102	36.064

Table 5.6: RMSE and MAPE results of machine learning models applied on training weekly univariate time series compared to persistence, ARIMA and SARIMA models

Next we will use the chosen ARIMA and SARIMA models to make predictions on out of sample data.

Chapter 6

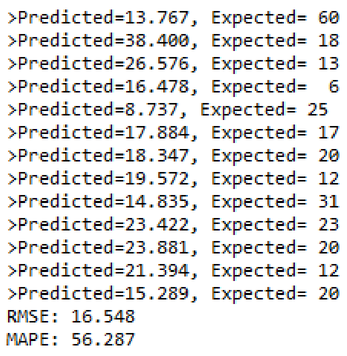
Making Predictions and Model Evaluation

After we're done choosing the preferred ARIMA and SARIMA models for our forecasts, we will now apply these models and test their performance on data from the year 2017.

6.1 ARIMA Final Predictions

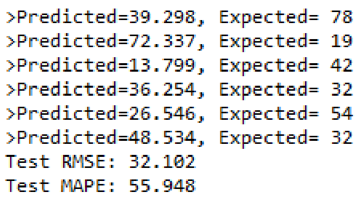
Below are the final predictions done using ARIMA(2,0,1), ARIMA(0,2,1), and ARIMA(3,2,0) with box cox transform and bias-correction to get better results in predicting weekly, biweekly and monthly demand respectively. Recall ch. 4 sec. 4.4

Note that the plots show the predictions in red, versus the actual values in blue.



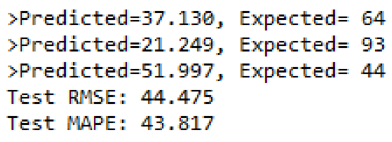
```
>Predicted=13.767, Expected= 60
>Predicted=38.400, Expected= 18
>Predicted=26.576, Expected= 13
>Predicted=16.478, Expected=  6
>Predicted=8.737, Expected= 25
>Predicted=17.884, Expected= 17
>Predicted=18.347, Expected= 20
>Predicted=19.572, Expected= 12
>Predicted=14.835, Expected= 31
>Predicted=23.422, Expected= 23
>Predicted=23.881, Expected= 20
>Predicted=21.394, Expected= 12
>Predicted=15.289, Expected= 20
RMSE: 16.548
MAPE: 56.287
```

(a) weekly



```
>Predicted=39.298, Expected= 78
>Predicted=72.337, Expected= 19
>Predicted=13.799, Expected= 42
>Predicted=36.254, Expected= 32
>Predicted=26.546, Expected= 54
>Predicted=48.534, Expected= 32
Test RMSE: 32.102
Test MAPE: 55.948
```

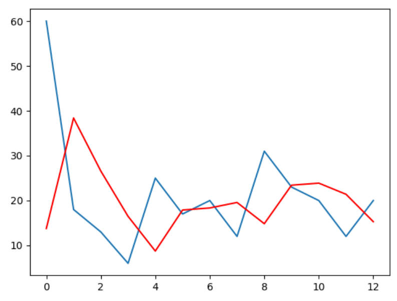
(b) bi-weekly

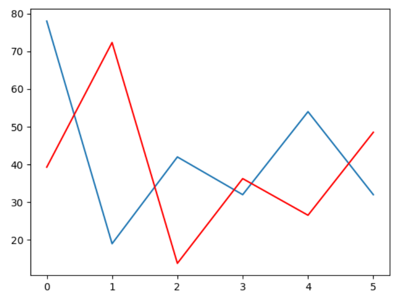


```
>Predicted=37.130, Expected= 64
>Predicted=21.249, Expected= 93
>Predicted=51.997, Expected= 44
Test RMSE: 44.475
Test MAPE: 43.817
```

(c) monthly

Figure 6.1: ARIMA Final Predictions on out of sample data


(a) weekly


(b) bi-weekly

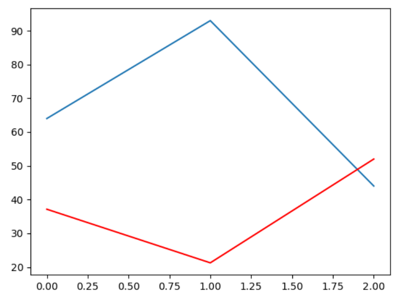

(c) monthly

Figure 6.2: plot of Final ARIMA predictions (red) versus actual validation data (blue)

- Weekly Case:

  Few comments on the above results:

  - Concerning the weekly forecast, the error is 16.5 demands. This result is very close to the average performance obtained by persistence model where the result of RMSE was around 16.5 demands per week.Figure 3.3(a)
  - If we have a look at the plot of predictions, we can see how the predictions look some how of similar trend as the original plot .

- Bi-Weekly Case:

  Few comments on the above results:

  - Concerning the bi-weekly forecast, in average there will be an error of 32 demands. This result is worse than the average performance obtained by persistence model where the result of RMSE was around 21 demands per month.Figure 3.3(b)
  - Concerning the plot, it seems that the predictions show an overall similar trend as the true observations however it is coming later by one time step.

- Monthly Case:

  Few comments on the above results:

  - Concerning the monthly forecast, in average there will be an error of 44.7 demands. This result is better than the average performance obtained by persistence model where the result of RMSE was around 65.6 demands per month.Figure 3.3(c)
  - Concerning the plot, we only have 3 predictions. We cannot get much information about the performance of the model since the validation data is very few, we might need to get more data and validate our work again.

The table below summarizes the final RMSE results of ARIMA done on unseen data with respect to the mean and persistence model RMSE:

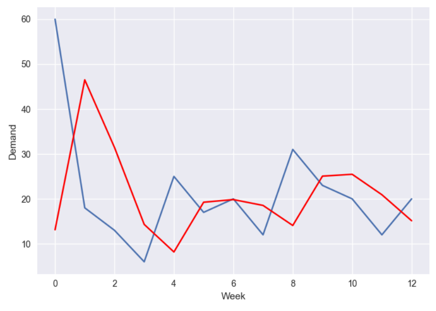
Case	P-RMSE	P-MAPE	ARIMA-RMSE	ARIMA-MAPE	Mean
Weekly Data	16.289	37.157	16.548	56.287	48.644
Biweekly Data	21.322	32.102	32.102	55.948	97.288
Monthly Data	65.609	58.607	44.475	43.817	210.791

Table 6.1: RMSE and MAPE results of persistence model and ARIMA model done on unseen data compared to the mean

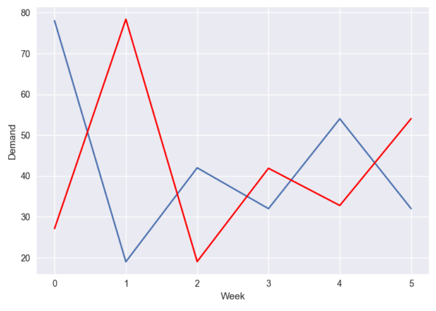
The RMSE results are not better than the persistence model but even worse. This indicates that the model did not perform very well in predicting out of sample data.

6.2 SARIMA Final Predictions

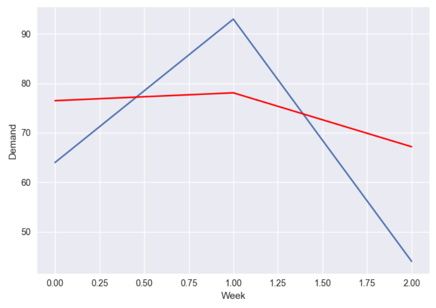
Here we showed the final predictions done using the chosen intuitive SARIMA(1,1,0)(0,0,1,12) and SARIMA(0,1,0)(0,0,1,6) models for weekly and bi-weekly cases respectively and BIC-based SARIMA(0,1,0)(0,1,1,3) for monthly case:



(a) weekly

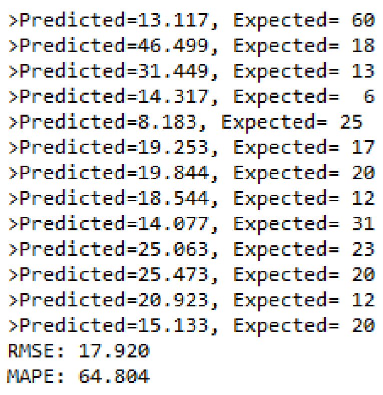


(b) bi-weekly



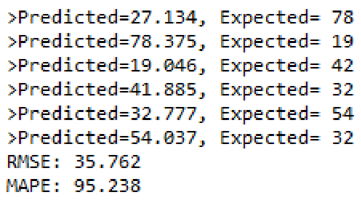
(c) monthly

Figure 6.3: plot of Final SARIMA predictions (red) versus actual validation data (blue)



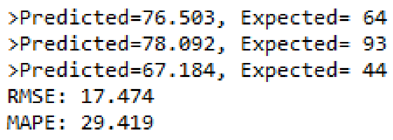
```
>Predicted=13.117, Expected= 60
>Predicted=46.499, Expected= 18
>Predicted=31.449, Expected= 13
>Predicted=14.317, Expected=  6
>Predicted=8.183, Expected= 25
>Predicted=19.253, Expected= 17
>Predicted=19.844, Expected= 20
>Predicted=18.544, Expected= 12
>Predicted=14.077, Expected= 31
>Predicted=25.063, Expected= 23
>Predicted=25.473, Expected= 20
>Predicted=20.923, Expected= 12
>Predicted=15.133, Expected= 20
RMSE: 17.920
MAPE: 64.804
```

(a) weekly



```
>Predicted=27.134, Expected= 78
>Predicted=78.375, Expected= 19
>Predicted=19.046, Expected= 42
>Predicted=41.885, Expected= 32
>Predicted=32.777, Expected= 54
>Predicted=54.037, Expected= 32
RMSE: 35.762
MAPE: 95.238
```

(b) bi-weekly



```
>Predicted=76.503, Expected= 64
>Predicted=78.092, Expected= 93
>Predicted=67.184, Expected= 44
RMSE: 17.474
MAPE: 29.419
```

(c) monthly

Figure 6.4: SARIMA Final Predictions on out of sample data

Case	P-RMSE	P-MAPE	SARIMA-RMSE	SARIMA-MAPE	Mean
Weekly Data	16.289	37.157	17.920	64.804	48.644
Biweekly Data	21.322	32.102	35.762	95.238	97.288
Monthly Data	65.609	58.607	17.474	29.419	210.791

Table 6.2: RMSE and MAPE results of persistence model and SARIMA model done on unseen data compared to the mean

- Weekly Case:

  For the weekly case we notice that the predictions have similar trend as actual data. The RMSE value shows that we were wrong by around 18 demands per week which is not a very small error considering an average of almost 49 demands per week. The MAPE value shows a percentage of almost 65% of error which again is not small.

- Bi-Weekly Case:

  For the biweekly case we can see again a similar trend of predictions as that of actual observations with a little shift to the right. The MAPE is very high, and the RMSE result shows that we were wrong about almost 36 demands which is noticeable knowing that the average is almost 97.

- Monthly Data:

  Surprisingly the monthly case showed a relatively better performance where the MAPE result is 29% and we were wrong by 17 demands, RMSE= 17, given that the average is 211 demands per month. This might be because this case has very little amount of data (24 observations for training and only 3 observations for out of sample data).

6.3 Summary

This chapter showed the RMSE and MAPE performance measures of forecasts done on testing data taken from year 2017. If you have a close look at the predictions versus actual values in each case (weekly, biweekly or monthly) you may notice that the error per week, two weeks or even month is not little, it is noticeable and one cannot make decisions using such results, however; it is important that we studied these linear models applied on time series data in order to obtain a base line performance to which other models can be compared to. That's why we need to look for other ways to do a prediction that gives more knowledge to the red cross and civil defense and help them make decisions.

Next, we will use machine learning models to predict a numeric value of demand or to classify whether a weekly demand is High or Low based on a set of features which will be added to the data.

Chapter 7

Using Machine Learning Models to Predict Demand using Multivariate Data

Having done linear models on univariate time series data and since no good results were shown using these models and since we believe there are other features that might help predict demand other than date, we decided to build a multivariate data set that can be used to predict demand using machine learning models.

In this approach we had to try two ways, either to predict the numeric value of demand which is done through regression machine learning models, or the second way that is to predict whether the demand is going to be low (L) or High (H) using classification models. Thus, in this chapter we will handle our problem as a regression prediction problem and a classification prediction problem using multivariate data.

After deciding on what to add as features that may have some influence on demand, we will study the importance of these and try to pick up those with highest importance.

In both approaches, several machine learning models will be applied to the data with all features and data with selected features. Finally all results will be compared and analyzed.

Our main purpose behind this project is to give red cross and civil defense a prior notice about the rate of demand for car and fire accidents at certain times in the future. Predicting a numeric value for the demand rate appeared not to have a good accuracy based on the RMSE values that were shown using univariate data, this is mainly because the collected data was little and the history of data was not big enough to predict future records. Therefore, in this section we tried to expand the problem and include other features that might help give a good prediction of demand with an acceptable error. Thus given a set of feature values the model should be able to predict a demand with a certain acceptable error or classify with certain accuracy whether the demand will be low ("L") or high ("H"). For example, lets say we know the weather conditions of next week, the coming Lebanese events and the demand (class or value) of few weeks ago, given those we can run a classifier to classify whether the demand will be high or low next week or a certain regression model to predict a numeric value of demand with some error to consider.

Note that we will work on one data set in this chapter that is the weekly data set, and the work can be repeated in an exact way on other data sets and will show similar results. Before applying models we need to prepare our data set for both approaches.

In the first section of this chapter sec. 7.1 we will give an overview over the features used to build up the data, then we will explain how the data was labeled and changed into a format accepted by Weka to be modeled sec. 7.2. After that we will study the importance of features and how much they are influencing the demand rate in sec. 7.3. The final two sections show the application of regression (sec 7.4) and classification (sec 7.5) machine learning models respectively on training data and then tested on unseen data.

7.1 Data Description

Feature Selection is one of the most important concepts in machine learning which has a huge impact on the performance of your model. Choosing good features to train your model can highly influence the performance of models. Irrelevant or partially relevant features can negatively impact model performance.

In our work we tried to search for features that might have an impact on the demand for red cross and civil defense in case of accidents. Recall that our data was first shaped as time series data and modeled using univariate time series models. Using the time structure we were able to collect data features that most probably contribute to the output.

The first important data feature that was gathered with the help of time structure is weather. This includes temperature, precipitation, wind speed, and season. Our assumption is that these features are in some way correlated to traffic and fire accidents and hence influence the demand rate. Weather condition is one important thing that might cause car or fire accidents, weather data includes temperature, wind speed, precipitation and season [33]. Another important feature that was gathered from the temporal structure of data is the trend feature. This feature was detected during the seasonal decomposition of data done earlier in ch. 3 sec. 3.5. Recall the plot of weekly data decomposition:

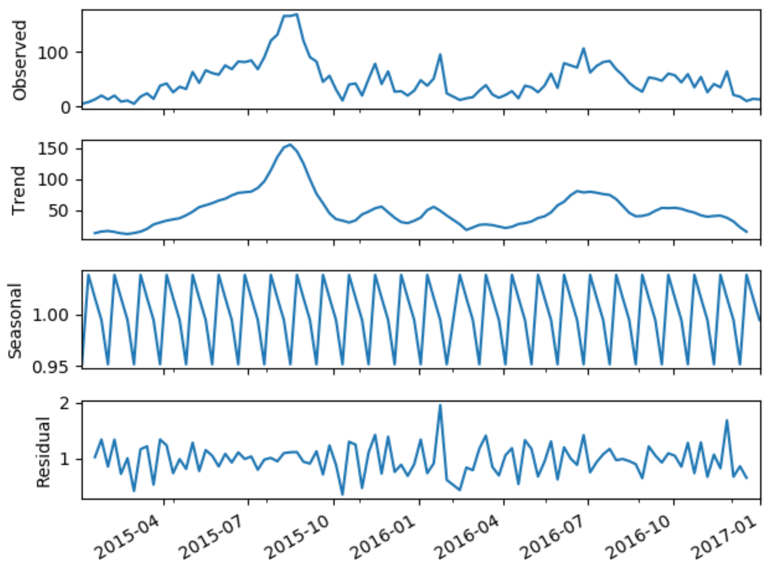


Figure 7.1: Seasonal decomposition of weekly data

Focus on the trend line plot (second plot from above 7.1), it shows the increase and decrease in demand without the fine grained fluctuations shown in the original line plot. What we did was extracting this trend component and adding it as a feature in our multivariate data.

Another important feature that was added to data is the lag feature that shows demand rate of previous weeks. In order to know how many lag features we have to add we looked at the autocorrelation plot of weekly data and searched for significant points (those outside the shaded part). Recall the autocorrelation plot of weekly data:

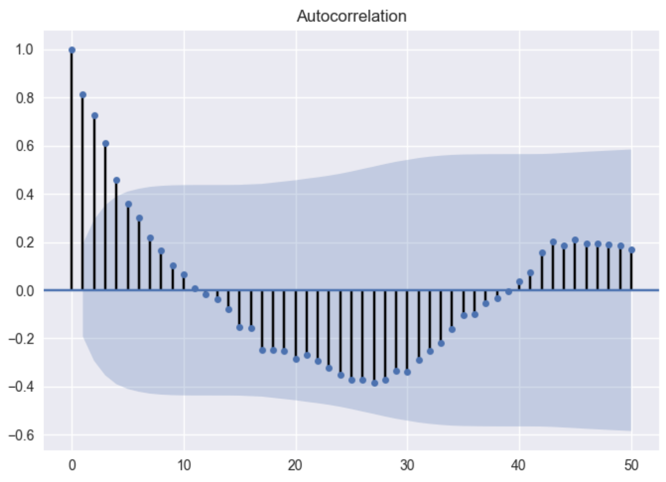


Figure 7.2: Zoom into the ACF of Weekly data to catch lag points

The first five points appear outside the shaded part, thus the first four lags (lag=1,2,3 and 4) show significance and it might be useful to add those as features. Adding these features is pretty easy. We just need to assume that the output demand (or class) is the current one, and take the one before for lag 1, the two before for lag 2 etc..

In addition to that, several events that happen during the year in Lebanon might affect the rate of demand at certain dates, for instance, official holidays (which include all holidays set by the government), festivals, football games, elections etc.. For example an event such as new year eve will definitely cause much accidents.

Moreover, in the past years Syria events caused a large number of refugees to come to Lebanon which increased the traffic in Lebanon and may have caused an increase in the demand rate for red cross or civil defense. To sum it up, several features were chosen to build our multivariate data (to know more about the details of these events check appendices B and C):

- Min,Max and Average temperature
- Average wind speed
- Average precipitation
- Season
- Number of Syria events
- Number of Lebanon events
- trend
- lag1: demand on the previous week
- lag2: demand two weeks before now
- lag3: demand three weeks before now
- lag4: demand four weeksbefore now.

The importance of these features and their impact on demand is definitely not the same, some features might have a very little influence and might affect on our model performance, therefore we need to make sure we select good features to have better results. In the next section we will study features importance.

7.2 Labeling Data

As we mentioned earlier the problem has changed from a time series forecasting problem into a regression or classification problem using machine learning models. For the regression part we simply use the weekly demand as output (similar to univariate data used in earlier chapters). For the classification problem we need to set the classes we want the classifier to learn. In our case and since our aim is to predict the severity of demand we will set classes or labels that give us such information depending on the actual numeric value of demand in a specific week. In order to know how to choose our labels we need to study the statistics of the data set and see how demand values are distributed taking into consideration that data in classification problems should be balanced so that the model won't be biased to a certain class:

count	104.00
mean	48.644
std	34.129
min	5.00
25%	23.5
50%	41.00
75%	64.50
max	168.00

Table 7.1: Statistics of Weekly Data

The statistics of weekly data shows that we have 104 records, the mean demand is almost 49, the min is 5 (demands per week) and the max is 168 (demands per week). By looking at the 50% percentile that shows a median of 41 demands we notice that the data can be divided into two classes and hence we assign two labels to the data:

- label "L" for demand < 41
- label "H" for demand $>= 41$

The distribution of labels appeared to be 50 records with "L" labels and 54 with "H" labels, which is a balanced distribution.

7.3 Studying Features

A very important step to consider is to measure the influence of each feature on red cross and civil defense demand. Before doing this, we will recall all features used in this project. Below is a table that shows a detailed description of our feature sets:

Feature Set	Included Features
Weather	Lebanon Temperature(Min,Max,average) Precipitation (average) Wind speed (average) Season (numeric values: 1=Fall, 2=Winter, 3=Spring, 4=Summer)
Events	Number of Syrian events(clashes,strikes,massacre,recapture,attacks...) Number of Lebanese events(sports, festivals, elections, holidays)
Trend	trend values extracted from the decomposition of data
Lags	lag1 lag2 lag3 lag4

Table 7.2: Details of feature sets

In order to evaluate the effect of these features on demand we will plot a stacked line plots of each feature and check how each varies with demand, the X-axis shows the week number of the two years 2015 and 2016 (since we're using weekly data):

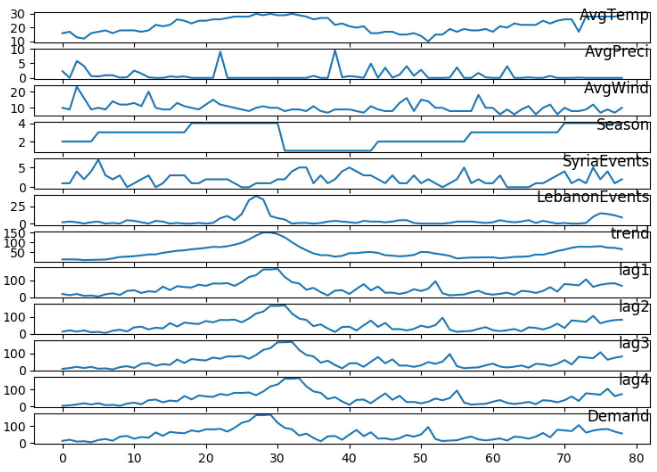


Figure 7.3: Stacked Line plots of each variable compared to demand

After analyzing the plots we notice that the average temp, season, Lebanon Events and trend features are mostly related to the variation in demand with respect to time. Of course trend and lag features will show correlation to demand and this should be obvious. What is actually interesting is the relation between temperature, season, Lebanon events and demand, we will plot these alone in a separate stacked plot to clarify this relation:

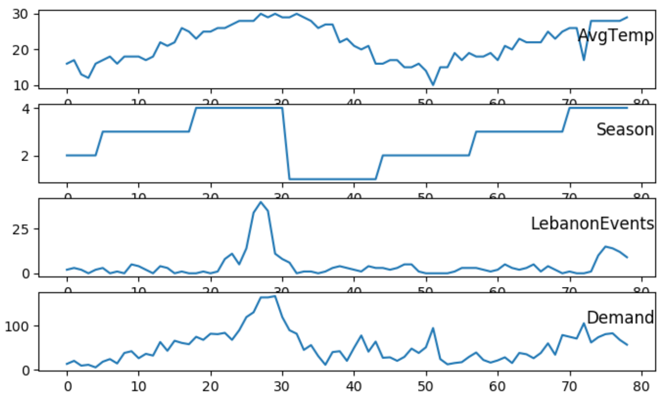


Figure 7.4: Stacked Line plots of temperature, season, Lebanon events and demand variables

This exploratory analysis of features leads us to the study of feature importance since we notice here that some features are more related to demand than others.

Feature importance is one method to help sort out what might be more useful in when modeling. The method involves fitting a random forest model (RandomForestRegressor), and summarizing the relative feature importance scores (appears on y-axis of the plot).[27]

A large-ish number of trees is used to ensure the scores are somewhat stable. Additionally, the random number seed is initialized to ensure that the same result is achieved each time the code is run.[27]

The results of feature importance on classification data set and regression data set are shown below:

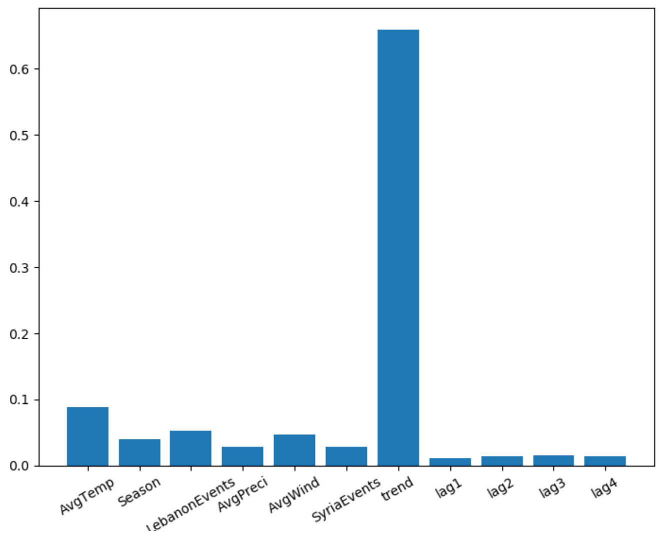


Figure 7.5: Feature importance scores of all features done on classification data set (the one with labels)

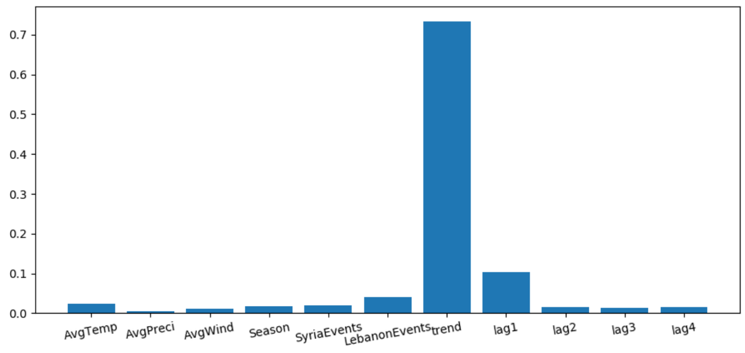


Figure 7.6: Feature importance scores of all features done on regression data set

Notice that in all cases the features show importance even though the trend feature shows the highest importance score (around 0.7). Since other features look dwarfed in the plot we can also use feature selection to automatically identify and select those input features that are most predictive.

A popular method for feature selection is called Recursive Feature Selection (RFE).

RFE works by creating predictive models, weighting features, and pruning those with the smallest weights, then repeating the process until a desired number of features are left.[27]

The figures below show the results of RFE with a random forest predictive model and sets the desired number of input features to 5. We will apply it on both data sets the labeled one and the one with numeric values:

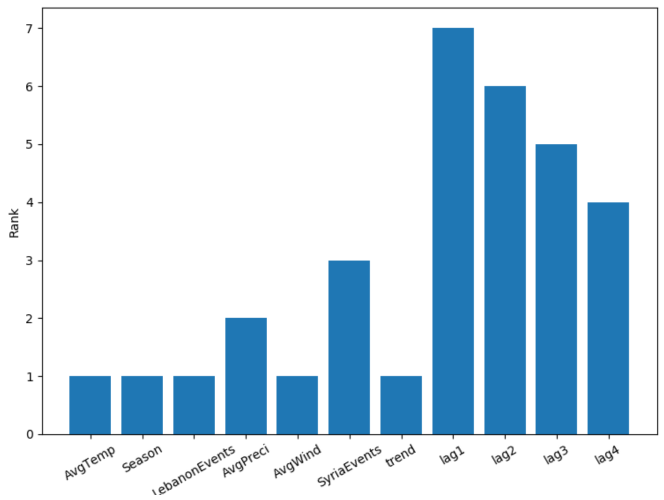


Figure 7.7: Rank scores of all features produced by RFE on labeled data

The model also produces the list of features with highest ranks (rank=1) for the case of labeled data the selected features (top ranks) are:

- AvgTemp
- Season
- LebanonEvents
- trend
- AvgWind

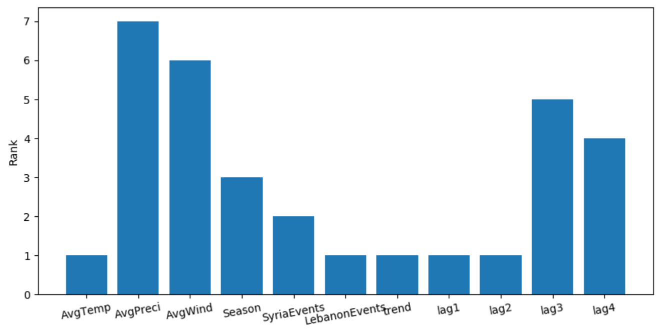


Figure 7.8: Rank scores of all features produced by RFE on regression data

The model also produces the list of features with highest ranks (rank=1) for the case of regression data the selected features (top ranks) are different:

- AvgTemp
- LebanonEvents
- trend
- lag1
- lag2

The results conform with the feature importance graph (fig 7.5) where the selected features seem to have higher importance than other features.

In addition, note that what we showed earlier in the stacked line plots is actually confirmed in both approaches (classification and regression) where in both the selected features actually show relation to demand in the plots (figure 7.3).

Later we will apply models on data with all features included and data with the selected features only and compare performances.

7.4 Using Regression Models to Predict Demand

Recall that before applying ARIMA and SARIMA models on univariate data we established a baseline performance using the persistence model and the RMSE result for the weekly data was **16.289** see table 3.1. We will keep in mind this value in order to compare it with regression machine learning models performance.

In this section we will train the data we've prepared earlier however instead of low or high values we will use the actual value of demand (in this case weekly demand) for the lag features and for the output feature. We will use weka to apply our models and report RMSE as error measure to be compared and analyzed.

Below is a list of the regression models that will be studied in this section:

- Linear Regression (simple linear regression (SLR))
- Polynomial Regression (Support Vector Machines (SVM) and Gaussian Processes (GP))
- Regression trees (Decision tree (DT))
- Regression forests (Random forest(RF))
- Neural networks (Multilayer Perceptron(MLP))

Note that here also we will apply models on data with all features included and data with selected features (recall sec. 7.3).

In order to avoid over-fitting while training our models, we will take away the last 20 records of data to be used as out of sample data on which the chosen model will be applied to test its performance. Besides, we will use the 10-fold cross validation for evaluating the model on training data.

The table below shows the RMSE results of the several regression machine learning models we applied:

Model	RMSE	MAE	RAE	RRSE
SLR	15.4276	11.87	40.5892%	41.6764%
SVM	**12.8267**	**9.4813**	**32.4211%**	**34.6503%**
GP	20.8925	17.6047	60.199%	56.439%
DT	25.6845	18.2977	62.5686%	69.3844%
RF	18.0754	14.2965	48.8866%	48.829%
MLP	22.4581	16.5375	56.5499%	60.6685%
Persistence	16.289	12.625	-	-

Table 7.3: Evaluation metrics of Regression ML models done on training data with **ALL Features** included

If we compare the obtained RMSE and MAE measures with the persistence model RMSE and MAE (16.289 and 12.625 respectively) we notice that only simple linear regression and support vector machines beat the persistence scores. The SVM model shows the lowest RMSE of 12.8267, this means that the model was wrong by an average of around 13 demands per week on training data. Now we will train the same models on data with selected features and see how the performance vary. We will use the features selected by the RFE algorithm applied on regression data sec 7.3.

Model	RMSE	MAE	RAE	RRSE
SLR	15.4276	11.87	40.5892%	41.6764%
SVM	13.7252	10.1391	34.6706%	37.0773%
GP	23.196	19.5969	67.0114%	62.6618%
DT	25.6845	18.2977	62.5686%	69.3844%
RF	17.6948	13.8382	47.3195%	47.8008%
MLP	18.8475	14.2885	48.8593%	50.9147%
Persistence	16.289	12.625	-	-

Table 7.4: Evaluation metrics of Regression ML models done on training data with **SELECTED Features** included

Notice now that similar to the previous conclusion the SLR and SVM are the only models that beat the persistence model performance with an RMSE of 15.4276 and 13.7252 respectively and MAE of 11.87 and 10.13 respectively. The other models show a very little improvement in performance than the ones applied on data with all features. Since SVM on all features show a better performance than the one applied on selected features we will stick to it being applied on data with all features since it shows the best performance on training data.

For the sake of comparing models' performances on predicting numeric demand we will again compare the SVM RMSE result on training data (weekly data) to that of chosen ARIMA and SARIMA (applied on weekly data), below is a table that shows the RMSE results of these models compared to persistence:

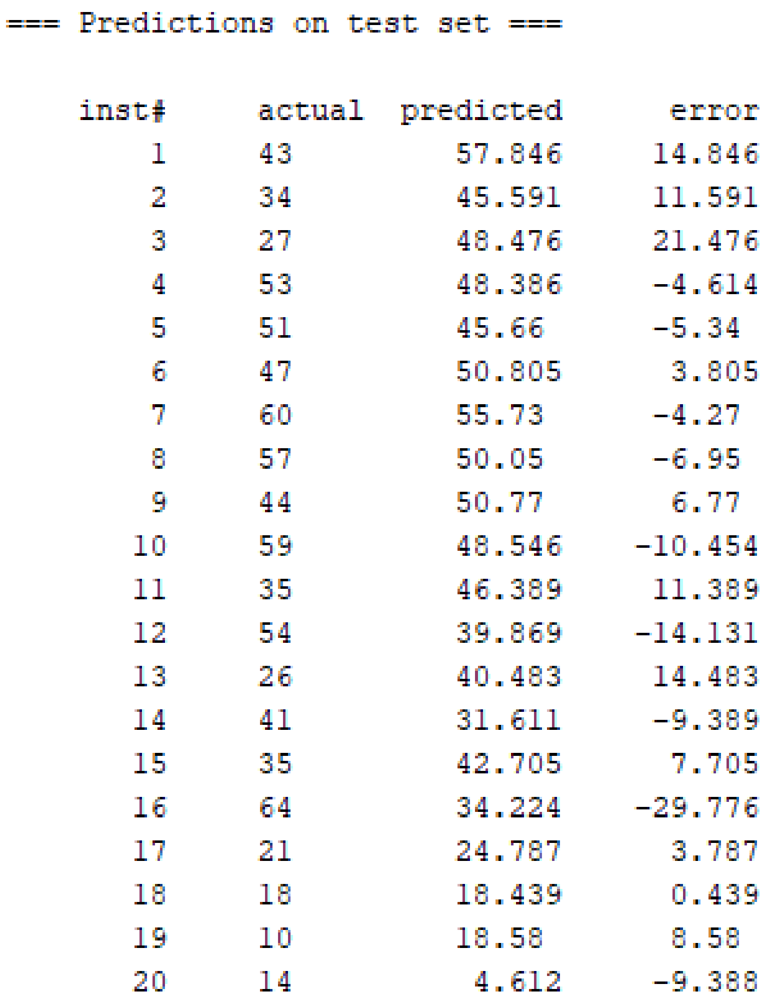
Model	RMSE
SVM	12.8267
ARIMA	14.185
SARIMA	15.102
Persistence	16.289

Table 7.5: RMSE of SVM model compared to ARIMA, SARIMA and Persistence done on training data

Note that SVM beats all other models in predicting numeric demand on training data.

To sum it up, we will choose SVM model to be tested on unseen data since it showed the best performance among others in the training process.

Testing Model on Unseen Data After we studied several machine learning models to predict demand, and taking into consideration that compared to baseline persistence model done earlier, the SVM performed best on data, we will test this model on unseen data (including all features) the predictions versus actual demand are shown in the figure 7.9:



```
=== Predictions on test set ===

 inst#     actual  predicted      error
     1     43         57.846     14.846
     2     34         45.591     11.591
     3     27         48.476     21.476
     4     53         48.386     -4.614
     5     51         45.66      -5.34
     6     47         50.805      3.805
     7     60         55.73      -4.27
     8     57         50.05      -6.95
     9     44         50.77       6.77
    10     59         48.546    -10.454
    11     35         46.389     11.389
    12     54         39.869    -14.131
    13     26         40.483     14.483
    14     41         31.611     -9.389
    15     35         42.705      7.705
    16     64         34.224    -29.776
    17     21         24.787      3.787
    18     18         18.439      0.439
    19     10         18.58       8.58
    20     14          4.612     -9.388
```

Figure 7.9: Predictions vs actual demand using SVM regression model on test data

The reported RMSE shows **11.9405** which is an error of around 12 demands per week, a result that is not very far from the training RMSE and also it beats the baseline model that showed RMSE=16.289. This shows that the model in general is not over-fitting and performing fairly well in predicting demand. Below is a summary of the RMSE values of ARIMA, SARIMA and Persistence done test data compared to that of SVM:

Model	RMSE
SVM	11.9405
ARIMA	16.548
SARIMA	17.920

Table 7.6: RMSE of SVM model compared to ARIMA, SARIMA done on testing data

Again notice that SVM model beats the ARIMA and SARIMA models in predicting a numeric value of demand on unseen data

Next, we will try the second approach in which we try to classify demand as high or low based on a set of given features and using several machine learning classifiers.

7.5 Using Classification Models to Classify Demand

In this section we will start studying the performance of some popular machine learning classifiers on training data.

Similar to any modeling process we need here as well to establish a base line performance to which we can compare our results to make sure our models are performing well. After doing some research we noticed that sickit has a dummy classifier that can be used to establish base line performance. This dummy classifier has a parameter "strategy" which we can set into three different strategies: stratified, most frequent and uniform:

- stratified: According to this strategy, the classifier looks at the class distribution in our target variable to make its predictions. [34]
- most frequent: In the case of most-frequent, the dummy classifier will always predict the label which occurs most frequently in the training set.[34]
- uniform: Strategy uniform generates predictions uniformly at random.[34]

Studying the performance is mainly related to accuracy measures produced by each model, the higher the accuracy measures the better the performance of model. In this thesis we will focus mainly on four accuracy measures:

- Precision: Precision is the ratio of correctly predicted positive observations to the total predicted positive observations. The question that this metric answer is of all demands that labeled as high, how many actually are high? High precision relates to the low false positive rate.
- Recall (or sensitivity): Recall is the ratio of correctly predicted positive observations to all observations in actual class - yes. The question recall answers is: Of all the demands that truly are high, how many did we label?
- F-measure: F-measure is the weighted average of Precision and Recall. Therefore, this score takes both false positives and false negatives into account. Intuitively it is not as easy to understand as accuracy, but F1 is usually more useful than accuracy, especially if you have an uneven class distribution. Accuracy works best if false positives and false negatives have similar cost. If the cost of false positives and false negatives are very different, it's better to look at both Precision and Recall. [35]
- AUC: It is one of the most important evaluation metrics for checking any classification model's performance. It is also written as AUROC (Area Under the Receiver Operating Characteristics). It tells how much model is capable of distinguishing between classes. An excellent model has AUC near to the 1 which means it has good measure of separability. A poor model has AUC near to 0 which means it has worst measure of separability. [36]

In order to avoid over-fitting while training our models, we will take away the last 20 records of data to be used as out of sample data on which the chosen model will be applied to test its performance.

The models will be applied on training data with all the features and data with selected features (recall sec. 7.3). We will try several possible classification models and use the 10-fold cross validation for evaluating the model on training data.

Before going into the modeling process in which we use different classifiers, below are the accuracy measures obtained by the stratified, most frequent and uniform algorithms. These results will be used in comparing models' performances later.

Model	Precision	Recall	F-measure
stratified	0.51	0.51	0.51
most frequent	0.55	0.55	0.55
uniform	0.45	0.45	0.45

Table 7.7: Accuracy Measures of baseline ML models done on training data

The results show that before and after feature selection the accuracy measures remained the same for the three base line models. We will use these results to compare our models applied on data with all features and data with selected features.

After trying many popular classification models we chose few that have shown the best accuracy measures among others, these include:

- Simple logistic (SL)
- Decision Tree (DT)
- Logistic model (L)
- Random Forest (RF)
- Adaboost (AB)

The accuracy measures of each model applied on our training data set having all features are summarized in the tables below:

Model	Precision	Recall	F-measure	AUC
SL	0.903	0.899	0.899	0.956
DT	**0.917**	**0.899**	**0.898**	**0.856**
L	0.861	0.861	0.861	0.938
RF	0.838	0.835	0.836	0.931
AB	0.868	0.861	0.861	0.970

Table 7.8: Accuracy Measures of ML models done on training data with **ALL Features** included

Notice that almost all models show better accuracy results compared to the three base line models. Which implies that our models have a good performance. Since the decision tree model showed the highest precision 0.917 and a relatively high F-measure and Recall, we will adopt this model for now even though it does not show the best AUC measure.

We will examine next whether these same models perform better on data with selected features.

Recall the feature selection process that we studied earlier in sec. 7.3. We will remove the features that showed low importance scores (high ranks) and use the features selected by the RFE technique.

The accuracy measures of ML models applied on data with selected features are shown in the table below:

Model	Precision	Recall	F-measure	AUC
SL	0.910	0.900	0.900	0.959
DT	0.917	0.900	0.899	0.860
L	**0.913**	**0.913**	**0.913**	**0.964**
RF	0.888	0.888	0.887	0.966
AB	0.888	0.888	0.888	0.973

Table 7.9: Accuracy Measures of ML models done on training data with **SELECTED Features**

Again, we can see from table 7.9 that all models beat the baseline models. The logistic model improved noticeably after we removed some features where the F-measure became 0.913 which is the highest among all models. The accuracy measures of decision tree model still show very good performance with a high precision 0.917 and an F-measure of 0.899 however Adaboost shows the highest AUC measure that is 0.973.

Since the Logistic model done on data with selected features shows high accuracy measures in terms of precision, recall, F-measure and AUC (see table 7.9) we will adopt it to be tested on out of sample data.

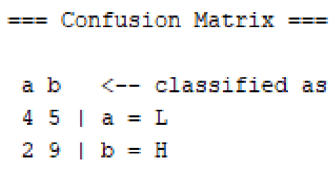
Testing Model on Unseen Data After we studied several machine learning models to classify the demand whether high or low, and taking into consideration that compared to baseline models done earlier the classification models performed better on data with selected features, we will now test our chosen model that is logistic model on testing data (with the same selected features) that was kept aside before the whole learning process as unseen data. We will also apply the baseline models (Ripper, J48 and zeroR) on test data to compare performance to logistic model.

The accuracy results of logistic and baseline models applied on testing data are shown in the table below (table 7.10):

Model	Precision	Recall	F-measure	AUC
L	**0.654**	**0.650**	**0.636**	**0.586**
stratified	0.45	0.45	0.45	-
most frequent	0.55	0.55	0.55	-
uniform	0.50	0.50	0.50	-

Table 7.10: Accuracy Measures of Logistic model done on testing data with selected features

Compared to all base line algorithm, the model seems to do a good job on predicting test data. The accuracy measures look fairly good, it shows that the model was able to classify around 65% of instances correctly that is 13 out of 20 and 35% incorrectly that is 7 instances out of 20. Below is the confusion matrix of the model that clarifies how did the model classify, what are the true and false positives and negatives..



```
=== Confusion Matrix ===

 a b   <-- classified as
 4 5 | a = L
 2 9 | b = H
```

Figure 7.10: Confusion Matrix of logistic model applied on test data

The logistic model matrix shows that 5 instances that are actually Low were classified as High and 4 instances were truly classified as low. This result is not very satisfying however it is not risky as well, since classifying true low as high will give a wrong information to the red cross and civil defense to get prepared to a high demand on accidents in case of low demands which is not very bad after all. On the other hand, 2 out of 11 instances with high labels were classified incorrectly as low and 9 were correctly classified as high. This is a good result, and will be a helpful information for red cross and civil defense with no much risk.

Summary To sum it up, in this chapter we studied the multivariate data and how it was collected and labeled to be modeled using regression models and classified using machine learning classifiers. We also studied the data features and selected four features according to some importance and rank scores. In addition, for achieving a baseline performance we used the persistence model RMSE and MAE values to be compared to regression models error measures and study performance accordingly. For the classification problem we applied three dummy classifiers in order to achieve a baseline performance to our study.

Finally we trained regression and classification models on data with all features versus data with selected features and chose the SVM model in case of regression problem and the logistic model in case of classification model since both showed the best performance during training. Both models were then applied on unseen data and results showed that the numeric predictions were fairly acceptable and that the classifier did a good job compared to baseline models.

In the next chapter will use LIME technique in order to explain the chosen model's predictions and how much we can actually trust our classifier.

Chapter 8

LIME Test: How much do we trust our model?

We most of the time look at machine learning models as black boxes where we give it some input and get predictions as output. Understanding the reasons behind these predictions is , however, important in assessing trust, and this is really significant if we're going to take some decisions based on a prediction as in our project. Such understanding also provides insights into the model, which can be used to transform an untrustworthy model or prediction into a trustworthy one.[37]

In our project, we will use LIME, a "novel explanation technique that explains the predictions of any classifier in an interpretable and faithful manner, by learning an interpretable model locally around the prediction" [37]. LIME which stands for "Local Interpretable Model-Agnostic Explanations" is an algorithm that can explain the predictions of any classifier or regressor in a faithful way, by approximating it locally with an interpretable model. By "explaining a prediction", the authors mean presenting textual or visual artifacts that provide qualitative understanding of the relationship between the instance's features and the model's prediction. They argue that explaining predictions is an important aspect in getting humans to trust and use machine learning effectively, if the explanations are faithful and intelligible.

Before moving into the application of Lime on our models and data and interpreting the results, we will give a brief idea on how Lime test work.

8.1 How does Lime Test work?

Behind the workings of lime lies the big assumption that every complex model is linear on a local scale. While this is not justified in the paper [37] it is not difficult to convince yourself that this is generally sound, you usually expect two very similar observations to behave predictably even in a complex model. Lime then takes this assumption to its natural conclusion by asserting that it is possible to fit a simple model around a single observation that will mimic how the global model behaves at that locality. The simple model can then be used to explain the predictions of the more complex model locally. [38]

The general approach lime takes to achieve this goal is as follows [38]:

- For each prediction to explain, permute the observation n times.
- Let the complex model predict the outcome of all permuted observations.
- Calculate the distance from all permutations to the original observation.
- Convert the distance to a similarity score.
- Select m features best describing the complex model outcome from the permuted data.
- Fit a simple model to the permuted data, explaining the complex model outcome with the m features from the permuted data weighted by its similarity to the original observation.
- Extract the feature weights from the simple model and use these as explanations for the complex models local behavior.

Having understood the general overview on Lime next we will look at the results produced by Lime when applied on the testing data using the chosen model.

8.2 Applying Lime on Test data

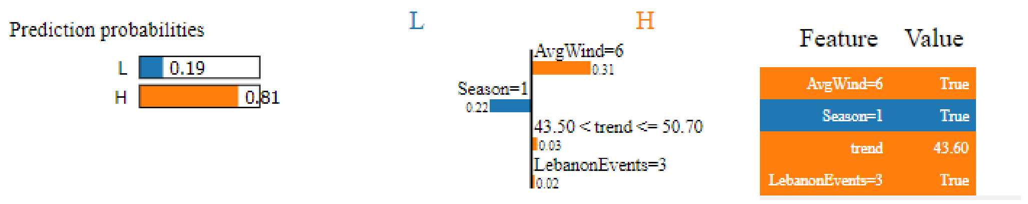
The models we chose were Logistic model in case of classification problem and the SVM model in case of regression problem. In order to understand better how the models are classifying or predicting demand we will apply LIME algorithm on the test data and check results.

For the implementation of LIME we will use the python environment where we can use the implementations and default parameters of scikit-learn. LIME can be installed using Anaconda (On ANACONDA distribution – pip install LIME), and libraries can be imported there. In order to view bar charts we used Jupyter notebook to run our code.[37]

The implementation follows several steps [37]:

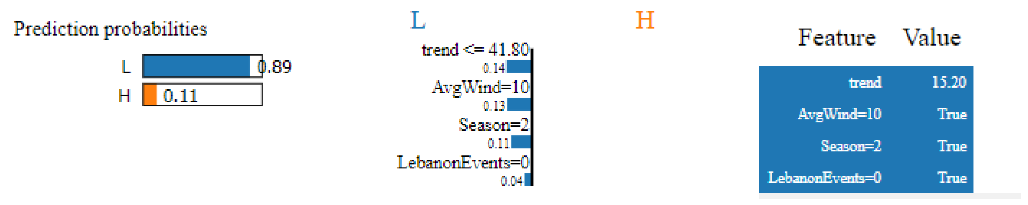
- For each classifier create a function that will return the predicted probability for the target variable (demand rate) given the set of features.
- Create a concatenated list of all feature names which will be utilized by the LIME explainer in subsequent steps.
- Create the LIME explainer
- Obtain the explanations from LIME for particular values in the validation dataset.

The figures below show how lime interprets our model predictions, the first figure is for a local observation with High label and the second is for Low labeled observation. In both cases the Lime test shows the actual values of features and colors the ones that most contribute to the actual prediction in the same color.



Feature	Value
AvgWind=6	True
Season=1	True
trend	43.60
LebanonEvents=3	True

Figure 8.1: Explaining individual predictions of Logistic Classifier trying to determine if a demand is Low "L" or High "H" (True Value = "H")



Feature	Value
trend	15.20
AvgWind=10	True
Season=2	True
LebanonEvents=0	True

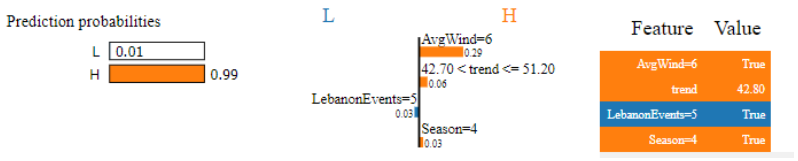
Figure 8.2: Explaining individual predictions of Logistic Classifier trying to determine if a demand is Low "L" or High "H" (True Value = "L")

The bar chart represents the importance given to the most relevant features. Color indicates which class the prediction contributes to (blue for "L", orange for "H").

In fig. 8.1, the model predicts that demand is H (higher probability=0.81), and LIME highlights the features that led to the prediction. Average wind speed, trend and number of Lebanon events are portrayed as contributing to the "H" prediction, while season is an evidence against it (season=1 is Fall), this means that at this local observation the season is Fall and it seems that the model interprets Fall season to be contributing to a low demand not high however in this case the demand is high even though it's Fall. This is explained by the explainer which tells us that the season= 1, meaning Fall, is an evidence against the "H" prediction, but given that other features contribute to High demand the model was able to predict a correct prediction. With these, one can make an informed decision about whether to trust the model's prediction. Well we know from earlier data analysis that demand is most of the time high during spring and summer seasons where most of the events take place so we understand why the season=1 contributes to Low demand not high.

On the other hand, fig. 8.2 shows that the model predicts Low demand (higher probability= 0.89). The bar chart shows that all features values in this case contributes to the prediction of Low demand. The trend has a low value (15.20) the average wind speed is low, the season = 2 meaning Winter season and no Lebanon events occurred during this week hence the Lime interpretation gives us a good information about the model performance, and shows that the model at this local observation did a good job predicting a low demand since all features contribute to this result.

For the sake of understanding better how does LIME interpret predictions we tried to test it on false prediction. We chose an observation that is actually low but was predicted as high, the figure-8.3 shows the bar chart result from lime test applied on this local observation:



Feature	Value
AvgWind=6	True
trend	42.80
LebanonEvents=5	True
Season=4	True

Figure 8.3: Explaining individual predictions of Logistic Classifier trying to determine if a demand is Low "L" or High "H" (True Value = "L", Predicted value= "H")

Notice here that only the LebanonEvents feature contribute to low demand label (low number of events contribute to low demand). The AvgWind, trend and season values show that the demand should be high since their values contribute to high demand, notice the high trend value, the summer season (Season=4) and the average wind=6. This case shows that the LIME test interpreted the features well however the model prediction was false.

LIME is not a feature importance tool because it is not restricted to this information it tells more about the performance of the model and how any model is achieving its predictions thus telling if this model is trustworthy or not. However LIME aids in the process of selecting features. If one notes that a classifier is untrustworthy, a common task in machine learning is feature engineering, i.e. modifying the set of features and retraining in order to improve generalization. Explanations of LIME can aid in this process by presenting the important features, particularly for removing features that the users feel do not generalize [37].

Chapter 9

Thesis Summary

9.1 Thesis contribution

This thesis proposes a real-time forecasting system for demand on red cross and civil defense in case of accidents based on machine learning approaches. In addition to stochastic linear models applied on univariate time series, this thesis proposes classification methods to build a comprehensive and robust model that can be applied on multivariate data to predict the severity of demand given a set of features. Exploratory data analysis is a must for any data science project, it shows the main components of a data set and the important statistics that help in the prediction process. This thesis adopts two main approaches for forecasting demand. The first is for univariate time series forecasting and based on linear parametric models ARIMA and SARIMA for which parameters were configured and studied to give a higher performance. The second approach is for multivariate data and based on machine learning models to predict whether the demand is high or low given a set of significant features.

During the EDA we studied the important visualizations of the time series which showed that the data has an increasing and decreasing trends and no clear global trend. In addition to that the visualizations showed an existence of some seasonality signals. For that reason and since ARIMA and SARIMA assume data stationarity we had to apply differencing to time series to make sure data is stationary. The EDA also showed that the data is not normally distributed which also made us try transforming of time series by Box cox transform. Having done all the preprocessing of data we finally applied ARIMA model with parameters chosen using ACF and PACF plots and grid search technique. The predictions were then corrected and improved after studying the residuals.

In order to deal with seasonality we used the SARIMA modeling method which adds the seasonality factor and make use of it for modeling time series data. Two main approaches were studied in this case: an intuitive approach that was based on our intuition and our study of parameters and evaluated using MAPE. The other approach was based on a grid search for seasonal and non-seasonal parameters and evaluated using BIC value. The two approaches are then compared and the better was chosen to be applied on unseen data.

The best ARIMA and SARIMA models that were chosen after the whole study were applied on unseen data of 2017. The RMSE values were used to compare models' performances. ARIMA showed a better performance but still not a very good performance compared to persistence model which was considered the base line performance.

Because we were unsatisfied by the ARIMA and SARIMA performance on time series data, we tried to change the problem to a multivariate problem using machine learning. Two approaches were followed, regression problem in which we aim to predict demand as numeric, and a classification problem in which we predict whether the demand is high or low. In addition to that we added features other than time that we thought had influence on the demand rate such as weather data and number of Syrian and Lebanese events. Time components such as trend and lag variables were also used as features. Labeling the data was done using the descriptive statistics of the weekly data which showed that the median is 41 demands and hence we can set all demands less than 41 as "Low" and those greater than 41 as "High". Feature Importance and selection was done later and showed that the season, avg wind speed, trend and number of Lebanese events are the most significant features. Therefore we applied several machine learning models on data with all features versus data with selected features and compared the accuracy measures. The results showed that data with selected features showed a better performance. The model that showed best accuracy on training data is the Logistic model so it was chosen to be tested on unseen data (the last 20 records of 2016). The model showed 65% accuracy which was not bad.

Finally in order to understand better how the model predicted and how much can we trust its predictions we underwent LIME test which is an algorithm that explains predictions of classifiers based on a local prediction interpretation. The lime test showed that our model is basically doing well giving a higher probability to the actual value of observation. The test explains the prediction by giving weights to features that are mostly contributing to the prediction and shows those that are evidences against the prediction.

9.2 Future research direction

9.2.1 Spatio-temporal Analysis

The basic idea behind this project was to forecast the demand on red cross and civil defense in case of traffic or fire accidents. We had a goal at the beginning to forecast in addition to time the location of accidents, that's why we got during the data collection process the latitude and longitude at which the accident took place. However due to the small size of data we were unable to do the spatial forecasting, therefore in our future research we aim at collecting more spatial data and do the spatio-temporal analysis which will lead us to a spatio-temporal forecast that is to predict when and where an accident occurs.

9.2.2 Red Cross and Civil Defense Dispatch System

The goal of this thesis is to build a real-time estimator for the demand forecasting on red cross and civil defense, which assists and supports their service operations. Red cross for example can dispatch its ambulances according to the forecasted demand in the future. A meaningful direction is to build a system to provide decision support for red cross and civil defense dispatches on the basis of the demand estimator.

9.2.3 Incorporate more models

The main work of this thesis is to validate the idea of combining single models for the demand forecasting problem. So we only focus on the basic version of the machine learning models. Future research can incorporate the rich variants of those models.

Appendix A

Abbreviations

ARIMA	Autoregressive Integrated Moving Average
AR	Autoregressive
MA	Moving Average
SARIMA	Seasonal Autoregressive Integrated Moving Average
VAR	Vector Autoregrassion
VMA	Vector Moving Average
RMSE	Root Mean Squared Error
MAPE	Mean Absolute Percentage Error
SVM	Support Vector Machine
MLP	Mltilayer Perceptron
LSTM	Long Short Term Model
ANN	Artificial Neural Networks
RFE	Recursive Feature Selection
NB	Naive Bayes
SL	Simple Logistic
DT	Decision Tree
a.k.a.	also known as
LIME	Local Interpretable Model-Agnostic Explanations
AUROC	Area Under the Receiver Operating Characteristics
AUC	Area Under Curve
RAE	Relative Absolute Error
RRSE	Root Relative Square Error
MAE	Mean Absolute Error
VDC	Venture Development Center
ML models	Machine Learning Models

Appendix B

Lebanese Events

B.1 Official Holidays

include:

- Eid Al Adha
- Christmas Day
- Prophet Mohamad Birthday
- Hijra New Year
- Feast of Assumption
- Rafik Hariri Memorial
- Martyr's Day
- Labor Day
- Orthodox and Aremenian Christmas
- New year day
- Independence Day
- All Saints' Day

B.2 Football Games

include all important Lebanese football games done during 2015 and 2016

B.3 Festivals

Festivals include:

- Jounieh International Summer Festival,
- Zouk Mikeal International Music Festival,
- Ehdeniyat,
- Batroun International Festival,
- Byblos International Festival,
- Baalbek International Festival,
- Beiteddine Art Festival
- Cedars International Festival,

- Dbayeh International Festival,
- Tyre Festival,
- Tannourine Cedars Night Festival,
- Kobayat Festival

B.4 Other Lebanese Events

include:

- Elections
- Official Exams
- Terrorists Attacks

Appendix C

Syria Events

Include events collected using VDC:

- Major Attacks
- Air Strikes
- Major Clashes
- Execution
- Surrender
- Recapture
- Massacre
- Evacuation
- Assassination
- Alliances
- Ethnic Clearance
- Cease Fire
- Elections

In order to know more about the nature of these events refer to this link: https://github.com/elhamfar/MyFirst-Project/blob/master/Large%20scale%20events%20Timeline.xlsx and refer to this link to check the number of Lebanese and Syrian events per week in a time series form: https://github.com/elhamfar/MyFirst-Project/blob/master/Events.pdf

Bibliography

[1] A. Tall, S. J. Mason, M. van Aalst, P. Suarez, Y. Ait-Chellouche, A. A. Dialloand, and L. Braman, "Using seasonal climate forecasts to guide disaster management: The red cross experience during the 2008 west africa floods," *International Journal of Geophysics*, vol. 2012.

[2] L. Braman, M. Aalst, S. Mason, P. Suarez, Y. Ait-Chellouche, and A. Tall, "Climate forecasts in disaster management: Red cross flood operations in west africa, 2008," *Disasters*, vol. 37, 2012.

[3] J. S. Hill, "Red cross develops innovative mechanism to predict and prepare for flood risks," 2017.

[4] O. O. Bozkurt, G. Biricik, and Z. C. Taysii, "Artificial neural network and sarima based models for power load forecasting in turkish electricity market," 2017.

[5] K. Pottie, D. Grunerb, and O. Magwoodd, "Canada's response to refugees at the primary health care level," vol. 28(1):e2811803.

[6] W. H. O. 2017, *WORLD HEALTH RANKINGS LIVE LONGER LIVE BETTER*, 2017.

[7] S. Makridakis, E. Spiliotis, and V. Assimakopoulos, "Statistical and machine learning forecasting methods: Concerns and ways forward," *PLOS One*, 2018.

[8] "Wikipedia : Demand forecasting," 2019.

[9] G. E. P. Box and G. M. Jenkins, *Time Series Analysis: Forecasting and Control.* Upper Saddle River, NJ, USA: Prentice Hall PTR, 3rd ed., 1994.

[10] A. C. L. Cheng-Few Lee, John C. Lee, *Statistics for Business and Financial Economics.* Upper Saddle River, NJ, USA: Springer-Verlag New York, 2013.

[11] J. Brownlee, *Introduction to Time Series Forecasting with Python.* 2017.

[12] R. Xu, "Machine learning for real-time demand forecasting," 2015.

[13] K. Hipel and A. McLeod, *Time Series Modelling of Water Resources and Environmental Systems.* Developments in Water Science, Elsevier Science, 1994.

[14] Kihoro, "Seasonal time series forecasting : A comparative study of arima and ann models," 2004.

[15] "Improving artificial neural networks' performance in seasonal time series forecasting," *INFORMATION SCIENCES.*

[16] J. Faraway and C. Chatfield, "Time series forecasting with neural networks: A comparative study using the airline data," *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, vol. 47, pp. 231 – 250, 02 1998.

[17] J. Flaherty and R. Lombardo, "Modelling private new housing starts in australia," 01 2000.

[18] Y.-H. Song and X.-F. Wang, *Operation of Market-oriented Power Systems.* Power Systems, Springer-Verlag London, 2003.

[19] P. Zhang, "Zhang, g.p.: Time series forecasting using a hybrid arima and neural network model. neurocomputing 50, 159-175," *Neurocomputing*, vol. 50, pp. 159–175, 01 2003.

[20] E. Alpaydin, *Introduction to Machine Learning (Adaptive Computation and Machine Learning).* The MIT Press, 2004.

[21] T. Hastie, R. Tibshirani, and J. Friedman, *The elements of statistical learning: data mining, inference and prediction.* Springer, 2 ed., 2009.

[22] G. Oğcu, O. F. Demirel, and S. Zaim, "Forecasting electricity consumption with neural networks and support vector regression," *Procedia - Social and Behavioral Sciences*, vol. 58, pp. 1576 – 1585, 2012. 8th International Strategic Management Conference.

[23] S. Wang, L. Tang, and L. Yu, "Sd-lssvr-based decomposition-and-ensemble methodology with application to hydropower consumption forecasting," pp. 603 – 607, 05 2011.

[24] Y. Hou, "Forecast on consumption gap between cities and countries in china based on arma model," pp. 342–345, 11 2010.

[25] A. Al-Smadi and D. M. Wilkes, "On estimating arma model orders," in *1996 IEEE International Symposium on Circuits and Systems. Circuits and Systems Connecting the World. ISCAS 96*, vol. 2, pp. 505–508 vol.2, May 1996.

[26] L. Xuemei, D. Lixing, S. Ming, X. Gang, and L. Jibin, "A novel air-conditioning load prediction based on arima and bpnn model," in *Proceedings of the 2009 Asia-Pacific Conference on Information Processing - Volume 01*, APCIP '09, (Washington, DC, USA), pp. 51–54, IEEE Computer Society, 2009.

[27] J. Brownlee, *Feature Selection for Time Series Forecasting with Python.*

[28] P. Patil, *What is Exploratory Data Analysis?*

[29] P. S. P. Cowpertwait and A. V. Metcalfe, *Introductory Time Series with R.* Springer Publishing Company, Incorporated, 1st ed., 2009.

[30] S. Shakya, *ARIMA and SARIMA.*

[31] R. Nau, *Summary of rules for identifying ARIMA models*, 2017.

[32] E. P. Clement, *Using Normalized Bayesian Information Criterion (Bic) to Improve Box - Jenkins Model Building*, 2014.

[33] S. Thorsen, *Past Weather in Beirut, Lebanon*, 1995-2019.

[34] G. Subramanian, *How to automatically create Base Line Estimators using scikit learn.*, June 6, 2017.

[35] R. Joshi, *Accuracy, Precision, Recall and F1 Score: Interpretation of Performance Measures*, 2016.

[36] S. Narkhede, *Understanding AUC - ROC Curve*, 2018.

[37] M. T. Ribeiro, S. Singh, and C. Guestrin, ""why should i trust you?": Explaining the predictions of any classifier," in *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, (New York, NY, USA), pp. 1135–1144, ACM, 2016.

[38] T. L. Pedersen and M. Benesty, *Understanding lime*, 2018.