TUNKU ABDUL RAHMAN UNIVERSITY COLLEGE

DEPARTMENT OF MATHEMATICAL AND DATA SCIENCE

SEMESTER I, SESSION 2022/2023

BMMS2074 STATISTICS FOR DATA SCIENCE

ASSIGNMENT (100%)

Student Names	Student ID	Contribution(%)	Signature
Cecilia Kong Xin Ru	22WMR05329	25%	cecilia
Cheng Zhi Lin	22WMR05331	25%	ChengZL
Koh Jian Yong	22WMR05347	25%	KohJY
Fong Wei Hao	22WMR05340	25%	weihao
Total :		**100%**	

Programme : RDS2S1

Tutorial Group : G4

Date of Submission : 1st October 2022

Lecturer : Dr. WAN YOKE CHIN

Tutor : Dr. PEI LING TAN

Comments :

APPENDIX A: Peer Evaluation Rubrics

Program Learning Outcomes	Evaluation Criteria	Competency Levels			
		0-1 Unsatisfactory	2-3 Fair	4 Good	5 Outstanding
Teamwork (25%)	Working with Team Members **(5%)**	Rarely listens to team members. Shares input but struggles to collaborate (either takes control, does not participate, or makes decisions without team input).	Often listens to, shares with, is patient with, and supports the efforts of the team members. Makes some decisions without team input.	Usually listens to team members and responds with appropriate input. Supports the efforts of the team and is respectful.	Almost always listens carefully to team members. Demonstrates patience and respect. Identifies and encourages team member strengths. Collaborates with team members in a group decision making process and shares input effectively.
	Time Management **(5%)**	Struggles to get things done by the deadlines. Team has to adjust deadlines or work responsibilities as a result.	Tends to procrastinate, but always gets things done by the deadlines. Team does not have to adjust deadlines or work responsibilities.	Uses time well throughout the project to ensure things deadlines are met. Assists other team members with tasks if the need arises.	Facilitates team's use of time throughout the project to ensure deadlines are met. Volunteers to assist other team members with tasks.
	Contributions **(5%)**	Rarely provides useful ideas when participating in the group and in classroom discussion. May refuse to participate.	Sometimes provides useful ideas when participating in the group and in classroom discussion. A satisfactory group member who does what is required.	Usually provides useful ideas when participating in the group and in classroom discussion. A strong group member who tries hard.	Routinely provides useful ideas when participating in the group and in classroom discussion. A leader who contributes a lot of effort.
	Attitude **(5%)**	Is often publicly critical of the project or the work of other members of the group. Is often negative about the task(s).	Is occasionally publicly critical of the project or the work of other members of the group. Usually has a positive attitude about the task(s).	Is rarely publicly critical of the project or the work of others. Often has a positive attitude about the task(s).	Is never publicly critical of the project or the work of others. Always has a positive attitude about the task(s).
	Leadership and Participation **(5%)**	Does what is required but hesitates to or does not take leadership over the entire project.	Takes some responsibility for project. Shows leadership on certain aspects of the project.	Takes responsibility when asked or elected, shows good organizational and leadership skills within the team.	Facilitates team assignment of responsibilities, ensuring that work is shared. Shows initiative and good organizational skills.

PART-II: Depth of Knowledge Assessment Rubrics

Program Learning Outcomes	Evaluation Criteria	Competency Levels				Score
		0-3 Unsatisfactory	4-7 Fair	8-11 Good	12-15 Outstanding	
Critical Thinking and Problem Solving (75%)	Written Communication (15%)	Attempts to use a consistent system for basic organization; minimal attempts to use sources to support ideas in the writing and these sources may not be correctly documented using an appropriate referencing style and/or may not be fully relevant to the task at hand.	Follows expectations appropriate to a specific discipline and/or writing task for basic organization, and content; use credible and/or relevant sources to support ideas and to document these sources properly using APA or Harvard referencing style.	Demonstrates consistent use of important conventions particular to a specific discipline and/or writing task; consistently use credible, relevant sources appropriate to the discipline and genre to support ideas and documents sources with few errors or exceptions using APA or Harvard referencing style.	Demonstrates detailed attention to and successful execution of a wide range of conventions particular to a specific discipline and/or writing task (including organization, content, formatting, and stylistic choices); synthesize a range of high-quality, credible, relevant sources that are appropriate for the discipline and genre to develop ideas and fully documents these sources using APA or Harvard referencing style.	
	Problem Solving Strategy and Approaches **(15%)**	Unable to identify an approach to possible solution.	Identifies a possible but very general approach to a solution without a clear sense of the steps to solve the problem.	Identifies a reasonable and problem specific possible approach to a solution with some sense of steps to be undertaken to reach a solution.	Identifies at least one reasonable and problem specific possible approach to a solution. Outlines several steps in detail and/or identifies another reasonable and problem specific possible approach.	
	Analysis **(15%)**	Demonstrates emerging understanding of the data analysis without showing evident e.g. diagrams, models, timelines, illustrations, explanations, or a series of steps in the report. The report fails to tie into basic concepts and build on prior knowledge.	Demonstrates moderate understanding of the data analysis that are somewhat evident e.g. diagrams, models, timelines, illustrations, explanations, or a series of steps within the report. The report may fail to tie into basic concepts and build on prior knowledge.	Demonstrates considerable understanding of the data analysis and are evident e.g. diagrams, models, timelines, illustrations, explanations, or a series of steps within the report. The report ties into basic concepts and builds on prior knowledge.	Demonstrates in-depth/ thorough understanding of the data analysis and are clearly evident e.g. diagrams, models, timelines, illustrations, explanations, or a series of steps throughout the report. The report ties into basic concepts and builds on prior knowledge.	

	Critical Thinking and Perspective Taking **(15%)**	Specific position is stated but is simplistic and obvious.	Information is presented with some interpretation or evaluation, but not enough to develop a coherent analysis or synthesis.	Specific position takes into account the complexities of an issue and acknowledges other viewpoints.	Questions are examined from a range of viewpoints, taking into account the complexities of an issue.	
	Conclusions and Related Outcomes (Implications and Consequences) **(15%)**	Conclusion is inconsistently tied to some of the information discussed; related outcomes (consequences and implications) are oversimplified.	Conclusion is logically tied to information (because information is chosen to fit the desired conclusion); some related outcomes (consequences and implications) are identified clearly.	Conclusion is logically tied to a range of information, including opposing viewpoints; related outcomes (consequences and implications) are identified clearly.	Conclusions and related outcomes (consequences and implications) are logical and reflect student's informed evaluation and ability to place evidence and perspective discussed in priority order.	
					Total:	

Table of Content

1.0 Introduction

In this Assignment for BMMS2074 Statistics for Data Science, there are 4 members in our group which are Koh Jian Yong, Fong Wei Hao, Cecilia Kong Xin Ru and Cheng Zhi Lin. In this assignment, we have to find a dataset for statistical analysis. In the end, we have chosen a dataset with the title “Monthly Electrical Production” which is obtained from Kaggle.

Our dataset is an univariate time series data as our dataset only consists of one variable which is “Value”. “Value” represents the electrical production of the month. Besides, our dataset is continuous time-series data as the data is collected within a specific time frame which our data is collected on the first day of every month from 1985 to 2017.

For time series data, we are required to forecast the future value based on past value. In this case, there are few models that can be used to forecast future value such as TBATS, Exponential Smoothing (ETS), Holt Winter Model, Holt’s Method Model and Seasonal Autoregressive Integrated Moving Average (SARIMA). We plan to find the best model in forecasting the future electrical production among the 4 models we have stated above.

2.0 Objective

In this assignment, time-series data is used in various contexts and industries. In order to forecast future values based on the historical data, we are computing the time-series data using R Studio. The major objective of this study is to determine the optimum model for predicting and forecasting the electrical production over the following 10 year. Additionally, from 2008 to 2017, the trend and seasonality of electrical production caught our attention. Furthermore, this research allows us to deepen our knowledge of the topic and our proficiency with the R studio. As we discovered in this assignment, making predictions and forecasts is possible by comprehending the idea of time series data.

3.0 Methodology

3.1 Visualization

A time series is a collection of data points arranged sequentially. A time series is a collection of following points in time with equal intervals. Beside ,time-series analysis consists of techniques for processing time series data in order to gather important information and other valuable characteristics. In a wide range of sectors, including finance, pharmaceuticals, social media, web services, and research, time-series data analysis is becoming increasingly essential. Visualizations are necessary in order to comprehend the time-series data. Without visualizations, any sort of data analysis falls short of being full. That's because a good visualization can reveal useful and fascinating insights into the data. In the following section, we'll discuss how to visualize our dataset using time plots and how to apply the decomposition methodology.

3.1.1 Time Plot

Time plot also known as the time series graph or time series plot which is a data visualization tool used to illustrate the time series data on a Cartesian plane with a time-related attribute on x-axis and attribute to be measured on y-axis. In this context, a time plot is very essential to show the data changes over a period of time, although it does not show the distribution of any categories of the data as pie charts or bar charts. Each data point on the time plot is connected with a straight line according to the time attribute. In this project, the x-axis is labeled with year and y-axis is labeled as value of electricity production of the day in the United State.In fact, time plot helps to detect the patterns of time series data which can be used for prediction of future data and trends. In our project assignment, the time series data is visualized graphically to ease the detection of any possible trend and seasonality to make a more accurate prediction on the future data. However, building time series models is the most effective way to forecast data uncertainty factors in the future. For instance, the autoregressive (AR) model can be used to forecast a future data based on the residual value as the uncertainty and the previous data when there is a dependence of the time series data on the previous data.

3.1.2 Decomposition

Decomposition is useful statistical techniques used to decompose the time series data in to the components as following :

1. Level : the average value of the time series data
2. Trend : the increasing or decreasing changes in the time series data
3. Seasonality : the repetitive short-term cycle in the time series data
4. Residual : the randomness or noise in the time series data

All the time series data have level and residual, while trend and seasonality are optional components. (Chourasia,2020)

Time Series Additive Model	Time Series Multiplicative Model
Y = T + S + R	Y = T x S x R
Y = Value of time series data T = Trend of the time series data S = Seasonality in the time series data R = Residual in the time series data	

Table 1 : Different Model of Time Series Data

As shown in Table 1, the component of the time series data can be combined with addictive or multiplicative. Hence, there will be two methods of decomposition which are addictive and multiplicative decomposition. Addictive decomposition is suitable to be applied on an addictive model of time series data which has a constant trend and seasonality components as time increases. In contrast, multiplicative decomposition should be applied on a multiplicative model on time series data where the trend and seasonality components increase over time. Hence, applying an appropriate decomposition is able to provide an abstract model to analyze time series data and explore the ways to consider and forecast the time series data (Brownlee, 2017). For our assignment, we have to determine the series of models and apply the correct decomposition to show.

3.2 Preprocessing

In data preprocessing, we will split our data into training sets and testing sets. The training set is used to feed and build the forecasting model. After the model is built, we will use the model to forecast the test data. Then, we will compare the predicted value and actual value from the test data to evaluate the performance of the model. Additionally, we also will determine whether or not our data is stationary or seasonal by plotting it. The reason why we verify is because a time series' distribution can change from one period to the next if it is non-stationary. It is challenging to imagine how one might draw any conclusions about the time series with only one observation for each time period. Also, it is similar to trying to hit a moving target while knowing where it has been but being unaware of its direction of movement. Because non-stationary data are unpredictable, it is generally impossible to model or predict them. The inferences made from the use of non-stationary time series may be incorrect because they could indicate a relationship between two variables where none exists. To obtain reliable, consistent results, the non-stationary data must be transformed into stationary data, which is covered in the section below 3.2.1.

3.2.1 Differencing

Differencing is an essential method for transformation of time series data to achieve stationary. For instance, differencing can stabilize the mean of a time series data by removing changes in the level of a time series dataset and reducing the trend and seasonality. Since the time series dataset is a seasonal dataset, seasonal differencing will be applied on the seasonal time series data. The below Figure 3.2.1 shows how differencing is carried out by deducting the prior observation from the present observation.

```
1. difference(t) = observation(t) - observation(t-1)
```

Figure 3.2.1

3.2.2 Seasonal Differencing

$$y'_t = y_t - y_{t-m},$$

m = number of seasons

Seasonality differencing is used to determine the difference between observed data and the observed data from the previous year. In fact, seasonal stationary should be applied when the time series data has stationary noise.

3.3 Models

3.3.1 ETS Model

ETS model is known as Error Trend Seasonality model which is a non-stationary and univariate time series forecasting method. There are some principles in this model for example, trend techniques model, exponential smoothing and ETS decomposition. “ERROR”, “TREND”, “SEASONALITY” are the three important variables to be used in the model for “smoothing”. Furthermore, how does the ETS model work? ETS will generate a prediction based on the weighted average of every observation in the input time series dataset. Instead of being constant like in a simple moving average, the weights are decreasing exponentially with time. The smoothing parameter is the weights are dependent on constant parameters. Figure 3.3.1.1 shown below shows the equations in state space for every model in the ETS framework. The upper part of the figure is for addictive error models while the below part is for multiplicative error models. In the Seasonal component, there are three characteristics which are N (None), A (Addictive), and M(Multiplicative). Another three characteristics for Trend components are N (None), A (Addictive), and Ad(Addictive Damped).

ADDITIVE ERROR MODELS

Trend	Seasonal N	Seasonal A	Seasonal M
N	$y_t = \ell_{t-1} + \varepsilon_t$ $\ell_t = \ell_{t-1} + \alpha\varepsilon_t$	$y_t = \ell_{t-1} + s_{t-m} + \varepsilon_t$ $\ell_t = \ell_{t-1} + \alpha\varepsilon_t$ $s_t = s_{t-m} + \gamma\varepsilon_t$	$y_t = \ell_{t-1}s_{t-m} + \varepsilon_t$ $\ell_t = \ell_{t-1} + \alpha\varepsilon_t/s_{t-m}$ $s_t = s_{t-m} + \gamma\varepsilon_t/\ell_{t-1}$
A	$y_t = \ell_{t-1} + b_{t-1} + \varepsilon_t$ $\ell_t = \ell_{t-1} + b_{t-1} + \alpha\varepsilon_t$ $b_t = b_{t-1} + \beta\varepsilon_t$	$y_t = \ell_{t-1} + b_{t-1} + s_{t-m} + \varepsilon_t$ $\ell_t = \ell_{t-1} + b_{t-1} + \alpha\varepsilon_t$ $b_t = b_{t-1} + \beta\varepsilon_t$ $s_t = s_{t-m} + \gamma\varepsilon_t$	$y_t = (\ell_{t-1} + b_{t-1})s_{t-m} + \varepsilon_t$ $\ell_t = \ell_{t-1} + b_{t-1} + \alpha\varepsilon_t/s_{t-m}$ $b_t = b_{t-1} + \beta\varepsilon_t/s_{t-m}$ $s_t = s_{t-m} + \gamma\varepsilon_t/(\ell_{t-1} + b_{t-1})$
A_d	$y_t = \ell_{t-1} + \phi b_{t-1} + \varepsilon_t$ $\ell_t = \ell_{t-1} + \phi b_{t-1} + \alpha\varepsilon_t$ $b_t = \phi b_{t-1} + \beta\varepsilon_t$	$y_t = \ell_{t-1} + \phi b_{t-1} + s_{t-m} + \varepsilon_t$ $\ell_t = \ell_{t-1} + \phi b_{t-1} + \alpha\varepsilon_t$ $b_t = \phi b_{t-1} + \beta\varepsilon_t$ $s_t = s_{t-m} + \gamma\varepsilon_t$	$y_t = (\ell_{t-1} + \phi b_{t-1})s_{t-m} + \varepsilon_t$ $\ell_t = \ell_{t-1} + \phi b_{t-1} + \alpha\varepsilon_t/s_{t-m}$ $b_t = \phi b_{t-1} + \beta\varepsilon_t/s_{t-m}$ $s_t = s_{t-m} + \gamma\varepsilon_t/(\ell_{t-1} + \phi b_{t-1})$

MULTIPLICATIVE ERROR MODELS

Trend	Seasonal N	Seasonal A	Seasonal M
N	$y_t = \ell_{t-1}(1 + \varepsilon_t)$ $\ell_t = \ell_{t-1}(1 + \alpha\varepsilon_t)$	$y_t = (\ell_{t-1} + s_{t-m})(1 + \varepsilon_t)$ $\ell_t = \ell_{t-1} + \alpha(\ell_{t-1} + s_{t-m})\varepsilon_t$ $s_t = s_{t-m} + \gamma(\ell_{t-1} + s_{t-m})\varepsilon_t$	$y_t = \ell_{t-1}s_{t-m}(1 + \varepsilon_t)$ $\ell_t = \ell_{t-1}(1 + \alpha\varepsilon_t)$ $s_t = s_{t-m}(1 + \gamma\varepsilon_t)$
A	$y_t = (\ell_{t-1} + b_{t-1})(1 + \varepsilon_t)$ $\ell_t = (\ell_{t-1} + b_{t-1})(1 + \alpha\varepsilon_t)$ $b_t = b_{t-1} + \beta(\ell_{t-1} + b_{t-1})\varepsilon_t$	$y_t = (\ell_{t-1} + b_{t-1} + s_{t-m})(1 + \varepsilon_t)$ $\ell_t = \ell_{t-1} + b_{t-1} + \alpha(\ell_{t-1} + b_{t-1} + s_{t-m})\varepsilon_t$ $b_t = b_{t-1} + \beta(\ell_{t-1} + b_{t-1} + s_{t-m})\varepsilon_t$ $s_t = s_{t-m} + \gamma(\ell_{t-1} + b_{t-1} + s_{t-m})\varepsilon_t$	$y_t = (\ell_{t-1} + b_{t-1})s_{t-m}(1 + \varepsilon_t)$ $\ell_t = (\ell_{t-1} + b_{t-1})(1 + \alpha\varepsilon_t)$ $b_t = b_{t-1} + \beta(\ell_{t-1} + b_{t-1})\varepsilon_t$ $s_t = s_{t-m}(1 + \gamma\varepsilon_t)$
A_d	$y_t = (\ell_{t-1} + \phi b_{t-1})(1 + \varepsilon_t)$ $\ell_t = (\ell_{t-1} + \phi b_{t-1})(1 + \alpha\varepsilon_t)$ $b_t = \phi b_{t-1} + \beta(\ell_{t-1} + \phi b_{t-1})\varepsilon_t$	$y_t = (\ell_{t-1} + \phi b_{t-1} + s_{t-m})(1 + \varepsilon_t)$ $\ell_t = \ell_{t-1} + \phi b_{t-1} + \alpha(\ell_{t-1} + \phi b_{t-1} + s_{t-m})\varepsilon_t$ $b_t = \phi b_{t-1} + \beta(\ell_{t-1} + \phi b_{t-1} + s_{t-m})\varepsilon_t$ $s_t = s_{t-m} + \gamma(\ell_{t-1} + \phi b_{t-1} + s_{t-m})\varepsilon_t$	$y_t = (\ell_{t-1} + \phi b_{t-1})s_{t-m}(1 + \varepsilon_t)$ $\ell_t = (\ell_{t-1} + \phi b_{t-1})(1 + \alpha\varepsilon_t)$ $b_t = \phi b_{t-1} + \beta(\ell_{t-1} + \phi b_{t-1})\varepsilon_t$ $s_t = s_{t-m}(1 + \gamma\varepsilon_t)$

Figure 3.3.1.1 Equations in state space for every model in the ETS framework.

3.3.2 ARIMA Model

ARIMA model is a statistical analysis model that uses time series data to predict the future trends of the series or to have a better understanding of the dataset. If a statistical model forecasts or predicts future values based on present values, it is considered autoregressive. Lagged moving averages are used by ARIMA to smooth the time series data. They are frequently employed in technical analysis to predict upcoming share price movements. Each ARIMA component operates as a parameter using a standard notation. A common notation for ARIMA models would be ARIMA with p, d, and q, where the integer values stand in for the parameters to denote the type of ARIMA model being employed. The parameters can be defined as:

- p: Lag order(AR)
- d: Degree of differencing (I)
- q: Order of moving average (MA)

3.3.3 SARIMA Model

$$ARIMA\ (p, d, q)(P,D,Q)_s$$

The seasonal component of the series can be directly modeled using the SARIMA model, an extension of the AMIRA model. Univariate data containing trends and seasonality are frequently forecasted by a SARIMA model. A SARIMA model is formed by adding 3 new hyperparameters to an ARIMA model which are autoregressive(AR), differencing (I) and the moving average (MA) for the seasonal component of the series. Besides, it also includes an additional parameter for the seasonality period. Although the seasonal component of the model is made up of terms that are quite similar to the non-seasonal components, it also takes into account the seasonal period's back shifts. There are four seasonal components that does not belongs to ARIMA that must be configured:

- P: Seasonal autoregressive order
- D: Seasonal difference order
- Q: Seasonal moving average order
- s: Number of periods for each seasons

3.3.4 Holt's Method Model

$$\begin{aligned} \text{Forecast equation} \quad & \hat{y}_{t+h|t} = \ell_t + hb_t \\ \text{Level equation} \quad & \ell_t = \alpha y_t + (1-\alpha)(\ell_{t-1} + b_{t-1}) \\ \text{Trend equation} \quad & b_t = \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)b_{t-1}, \end{aligned}$$

Holt's Method model is an extension to simple exponential smoothing to let the data with a trend can be forecasted. This model involves a forecast equation and two smoothing equation with the smoothing parameters α , β^*. (α for Level equation and β^* for Trend equation).

3.3.5 Holt Winter Model

$$\begin{aligned}\hat{y}_{t+h|t} &= \ell_t + hb_t + s_{t+h-m(k+1)} \\ \ell_t &= \alpha(y_t - s_{t-m}) + (1-\alpha)(\ell_{t-1} + b_{t-1}) \\ b_t &= \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)b_{t-1} \\ s_t &= \gamma(y_t - \ell_{t-1} - b_{t-1}) + (1-\gamma)s_{t-m},\end{aligned}$$

Holt Winter model extends Holt's method to capture the seasonality of the time series data. This model includes a forecasting equation and 3 smoothing equations which are level, trend and seasonal components with the 3 smoothing parameters. The three order parameters, (α) alpha, (β*) beta, and (γ)gamma, are what characterize a Holt-Winters model. The coefficient for level smoothing is specified by alpha. The trend smoothing coefficient is specified by beta. The seasonal smoothing coefficient is specified by gamma.

3.3.6 TBATS Model

TBATS aim to forecast time series with complex seasonal patterns using exponential smoothing.

1. T : Trigonometric seasonality
2. B : Box-Cox transformation
3. A : ARMA errors
4. T : Trend Components
5. S : Seasonal Components

TBATS model able to deal with complex seasonalities such as non-integer seasonality,non-nested seasonality and large-period seasonality. The TBATS model is applied to create detailed and long-term forecasts with no seasonality constraints.This model will choose the lowest value of Akaike Information Criterion (AIC) as the final model.

The Formula for TBATS are shown below :

$$y_t^{(\lambda)} = l_{t-1} + \phi\, b_{t-1} + \sum_{i=1}^{T} s_{t-m_i}^{(i)} + d_t$$

Figure 3.3.6.1 - Box-Cox transformation Formula

$$l_t = l_{t-1} + \phi b_{t-1} + \alpha d_t$$

Figure 3.3.6.2 - Local Level Formula

$$b_t = \phi\, b_{t-1} + \beta d_t$$

Figure 3.3.6.3 - Trend Component Formula

$$d_t = \sum_{i=1}^{p} \varphi_i d_{t-i} + \sum_{i=1}^{q} \theta_i e_{t-i} + e_t$$

Figure 3.3.6.4 - ARMA errors Formula

$$s_t^{(i)} = \sum_{j=1}^{(k_i)} s_{j,t}^{(i)}$$

Figure 3.3.6.5 - Seasonal Component Formula

$y_t^{(\lambda)}$: Time series at moment *t* (Box-Cox transformed)

$s_t^{(i)}$: *i*th seasonal component

l_t : local level

b_t : trend with damping

d_t : ARMA(p,q) process for residuals

e_t : Gaussian white noise

T : Amount of seasonalities

m_i : Length of *i*th seasonal period

k_i : Amount of harmonics for *i*th seasonal period

λ : Box-Cox transformation

α,β : Smoothing

ϕ : Trend damping

φ_i, Θ_i : ARMA(p,q) coefficients

$\gamma_1^{(i)}, \gamma_2^{(i)}$ **:** Seasonal smoothing (two for each period)

As shown in Figure 3.3.6.1, the model combines the level,trend,seasonal components and ARMA to get the result which explains the feature of TBATS, it combines different components to predict time series.

3.4 Test

3.4.1 Augmented Dickey Fuller test (ADF test)

$$y_t = c + \beta t + \alpha y_{t-1} + \phi \Delta Y_{t-1} + e_t$$

where,

- y(t-1) = lag 1 of time series
- delta Y(t-1) = first difference of the series at time (t-1)

This statistical test, known as the Augmented Dickey Fuller test (ADF test), is frequently used to determine whether a time series is stationary or not. When examining the stationarity of a series, it is one of the statistical tests that is most frequently applied. Next, the ADF test is a statistical significance test so there will be a hypothesis test with a null hypothesis and an alternative hypothesis. Therefore, the test statistic will be calculated and p-value is reported.

The unit root test, of which the ADF test is a subset, is an appropriate way to assess a time series' stationarity. Time series have the property of unit root, which renders it non-stationary. A time series has a unit root if alpha = 1, which indicates that it does. ADF basically has a unit root test-like null hypothesis. In other words, since Y (t-1) has a coefficient of 1, there is a unit root.

The series is regarded as non-stationary if not rejected. One of the most widely used types of unit root tests is the Augmented Dickey-Fuller test, which was created using the equation above.

Last but not least, to reject the null hypothesis, the p-value obtained must be less than 0.05 (significance level). Since the null hypothesis is rejected, we can conclude that the time series data is stationary. On the other hand, if the p-value is more than 0.05, the null hypothesis is not rejected and we can conclude that the time series data is not stationary.

3.4.2 Kwiatkowski–Phillips–Schmidt–Shin (KPSS) test

$$x_t = r_t + \beta t + \varepsilon_1.$$

The Kwiatkowski–Phillips–Schmidt–Shin (KPSS) test is used to test whether the time series data is stationary around a mean or linear trend or is not stationary due to a unit root. Its null hypothesis states that the time series data is stationary while its alternative hypothesis states that the data is not stationary.

The KPSStest is based on linear regression. A series will be broken down into 3 parts which are a deterministic trend (βt), a random walk (rt), and a stationary error (εt), with the regression equation above. It will have a fixed element for an intercept if the time series data is stationary.

3.4.3 Autocorrelation Function (ACF)

$$\rho_k = \frac{E[(Y_t - \mu_y)(Y_{t-k} - \mu_y)]}{\sqrt{E[(Y_t - \mu_y)^2]E[(Y_{t-k} - \mu_y)^2]}} = \frac{Cov(Y_t, Y_{t-k})}{\sigma_{Y_t}\sigma_{Y_{t-k}}}$$

Figure 3.4.3.1 Equation to calculate autocorrelation with lag k

$$r_k = \frac{\sum_{t=k+1}^{n}(y_t - \bar{y})(y_{t-k} - \bar{y})}{\sum_{t=1}^{n}(y_t - \bar{y})^2}$$

Figure 3.4.3.2 Equation to calculate the The sample autocorrelation at lag k, denoted by rk

$$\gamma_k = \frac{1}{n}\sum_{t=k+1}^{n}(y_t - \bar{y})(y_{t-k} - \bar{y})$$

Therefore, $r_k = \frac{\gamma_k}{\gamma_0}$ for $k = 1,2,\ldots,m$, where $m \leq N$.

Figure 3.4.3.2 Equation to calculate the series of autocovariance, γ

One of the important steps in Exploratory Data Analysis is ACF which is also known as Autocorrelation Function. ACF will be helpful because it offers a partial description of the process for modeling purposes, even though it is typically impossible to get the complete description of a stochastic process. Figure 3.4.3.1 shows the formula that defines the autocorrelation with lag *k*. By considering the time series value $y_1, y_2, y_3, \ldots, y$, there is a simpler formula shown in Figure 3.4.3.2 which gives the similar answers. The combination the r value, the autocorrelation at 1,2,... make up the ACF where r_1 is the successive values of Y relate to each other and r_2 shows how Y values from different periods relates to one another and so on. Autocorrelation Function aids in identifying AR and MA parameters for the ARIMA model through the correlogram. Correlogram is a plot of sample autocorrelation coefficient rk against lag k for k = 0,1, …, m, where m is usually much less than n. We can get information about the random series, short-term correlation, non-stationary series, outliers and so on by interpreting ACF.

3.4.4 Partial Autocorrelation Function (PACF)

$$r_{kk} = \begin{cases} r_1 & \text{if } k = 1 \\ \dfrac{r_k - \sum_{j=1}^{k-1} r_{k-1,j} r_{k-j}}{1 - \sum_{j=1}^{k-1} r_{k-1,j} r_j} & \text{if } k = 2, 3, \ldots \end{cases}$$

$$\text{where } r_{kj} = r_{k-1,j} - r_{kk} r_{k-1,k-j} \text{ for } j = 1, 2, 3, \ldots, k-1$$

Figure 3.4.4.1 Equation to calculate the sample partial autocorrelation with lag, k

Partial autocorrelation function (PACF) is the measure of correlation between a time series data at a particular time and the lagged version of itself. Figure 3.4.4.1 shows the equation to determine the ACF value at lag k. The plot of PACF is known as the partial autocorrelation plots. For the AR model, the PACF is used to estimate the order value where the number of spikes after lag 0 indicates the expected order value of the AR model. For MA model, the PACF correlogram can be used to estimate the order value, where MA(1) model is expected when there is no sinusoidal wave pattern in the correlogram, while MA(p) model is expected when there is sinusoidal wave pattern in correlogram.

3.4.5 Box Pierce Q test and Ljung Box Test

$$Q = n \sum_{k=1}^{h} r_k^2$$

Figure 3.4.5.1 Equation of Box – Pierce Q test

n : The number of observations in the time series data

k : The number of lag

h : The maximum number of lag to be considered

$$Q^* = n(n+2)\sum_{k=1}^{h}(n-k)^{-1} r_k^2$$

Figure 3.4.5.2 Equation of Ljung – Box test

n : The number of observations in the time series data

k : The number of lag

h : The maximum number of lag to be considered

Box Pierce Q test and Ljung Box are different forms of Portmanteau test which is used to determine the compliance of the time series model to the White Noise model. Figure 3.4.5.1 and Figure 3.4.5.2 show the equation of Box-Pierce Test and Ljung Box Test respectively. Both test are used to examine the following hypotheses :

1. H_o : There is no significant difference between the mean of the residual and the value of zero.
2. H_1 : There is a significant difference between the mean of the residual and the value zero.

When the p-value of the Box Pierce Q test and Ljung Test is less than 0.05 indicates that the null hypothesis is rejected and the mean of the residual is significantly different from zero, so the model does not comply with White Noise model and is not suitable for forecasting purposes. On the other hand, when the p-value of the Box Pierce Q Test and Ljung Test is greater than 0.05, the null hypothesis is rejected and the mean of the residual is not significantly different from zero. Thus, the model complies with White Noise model and is suitable for forecasting purposes.

4.0 Data Sources

In this project, the time series dataset used is from Kaggle which is an open community that allows users to explore and publish datasets, explore and build models in web-based data-science environments, work with other data scientists and machine learning engineers, and enter competitions to solve data science challenges.

This time series dataset is publicly available on the Kaggle website, and it is the dataset with industrial production of electric, and gas utilities in the United States from the years of 1985 to 2018 with the frequency of Monthly production output. The dataset contains the 397 row of record and 2 attributes.

Our Time Series Dataset's Source is shown in the section below :

https://www.kaggle.com/datasets/shenba/electricity-production

Time Series Dataset - Electricity Production

The following are descriptions of the Dataset's attributes:

Attribute	Data Description	Data Type
DATE	Date of the electricity production in date format of dd-mm-yyyy.	Datetime
Value	Value of electricity production of the month.	Float

5.0 Data Analysis

5.1 Importing Time Series Data

```
#Importing Time Series Dataset
print(getwd())
setwd("C:/Users/cecil/Downloads/")
data <- read.csv("Electric_Production.csv")
```

Figure 5.1.1

In order to begin making predictions and forecasting on our electric production, as shown in figure 5.1 above, we will first import the csv file which its dataset name is Electric Production.

5.2 Data Overview

```
> #Display Summary of Dataset
> summary(data)
    DATE                Value
Length:397          Min.   : 55.32
Class :character    1st Qu.: 77.11
Mode  :character    Median : 89.78
                    Mean   : 88.85
                    3rd Qu.:100.52
                    Max.   :129.40
```

Figure 5.2.1 Command with Summary(data)

```
> #Listing our some data from each column
> str(data)
'data.frame':   397 obs. of  2 variables:
 $ DATE : chr  "01-01-1985" "02-01-1985" "03-01-1985" "04-01-1985" ...
 $ Value: num  72.5 70.7 62.5 57.5 55.3 ...
```

Figure 5.2.2 Command with str(data)

The functions of ‘summary’ and ‘str’ are used to look into the dataset information. Firstly, the function of ‘summary’ is used to show the summary of the time series dataset as shown in the Figure above .It is clearly shows that there is a total number of 397 rows in the time series dataset and the column date is in character type while the value is a float type which shows a descriptive statistics information such as minimum value, first quartile, median, mean, third

quartile and the maximum value. Next, The str function is then used to provide information regarding the rows (observations) and columns (variables), as well as additional information such as the names of the columns, class of each column, and a few of the initial observations of each column.

5.3 Data Cleaning

5.3.1 Checking The Null Value

```
> #Checking for any null/missing value
> for(i in colnames(data)){
+   cat(sum(is.na(i)),"Null values in the coulumn : ", i, "\n")
+ }
0 Null values in the coulumn :  DATE
0 Null values in the coulumn :  Value
>
```

Figure 5.3.1.1 Checking Null Value

In the session of data cleaning, the process of checking null values is done as shown in Figure 5.3.1.1. And, from the result shown in Figure 5.3.1.1 we can see that there are no missing values for any of the attributes in the dataset.

5.3.2 Changing Date Format From Character To Date

```
#Changing data format from char into date
data$DATE <- dmy(data$DATE)
summary(data)
str(data)
```

Figure 5.3.2.1

```
> data$DATE <- dmy(data$DATE)
> summary(data)
      DATE                 Value
 Min.   :1985-01-01   Min.   : 55.32
 1st Qu.:1993-01-04   1st Qu.: 77.11
 Median :2001-01-07   Median : 89.78
 Mean   :2001-01-21   Mean   : 88.85
 3rd Qu.:2009-01-10   3rd Qu.:100.52
 Max.   :2018-01-01   Max.   :129.40
> str(data)
'data.frame':   397 obs. of  2 variables:
 $ DATE : Date, format: "1985-01-01" ...
 $ Value: num  72.5 70.7 62.5 57.5 55.3 ...
>
```

Figure 5.3.2.1

The format of the attribute "DATE" must be changed from character to date format in order to enable further analysis. The result can be verified using the functions "summary" and "str" after transformation to the date format, as shown in Figure 5.3.2.1. It shows that the "DATE" column successfully changed into a date type, and the information from the descriptive analysis is displayed in Figure 5.3.2.1 above.

5.3.3 Data Visualization

```
#Visualize the density for Value Column
options(repr.plot.width=12, repr.plot.height=12)
valuePlot = ggplot(data, aes(Value)) + geom_histogram(bins = 10, aes(y = ..density..)
                  ,col ="red",fill = "red", alpha=0.3) + geom_density() + xlim(c(0, 140))
valuePlot
```

Figure 5.3.3.1

In order to get insight and understand the pattern of our time series dataset, we can visualize any informative graph after importing the Electric Production csv file from our dataset. We have plotted a histogram diagram in the figure 5.3.3.1 above to help us understand the data by using the visualization technique.

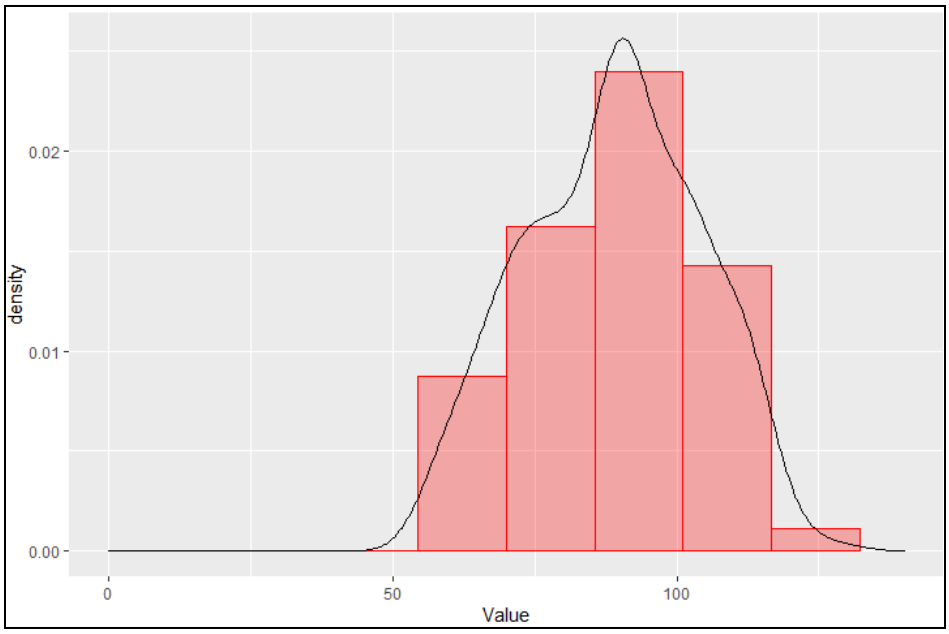

Figure 5.3.3.2

By creating histogram plots, the density for the attribute "Value" has been visualized in this session. The outcome shown in Figure 5.3.3.2 allows one to observe the density of the attributes.

```
#plot ori data
y <- data$value
y<- ts(y, start = c(1985,1), end = c(2018, 1), frequency = 12)
plot(y,xlab = "Year ", ylab="Electricity Production", main="Monthly Electricity Production from 1985 to 2018")
```

Figure 5.3.3.3

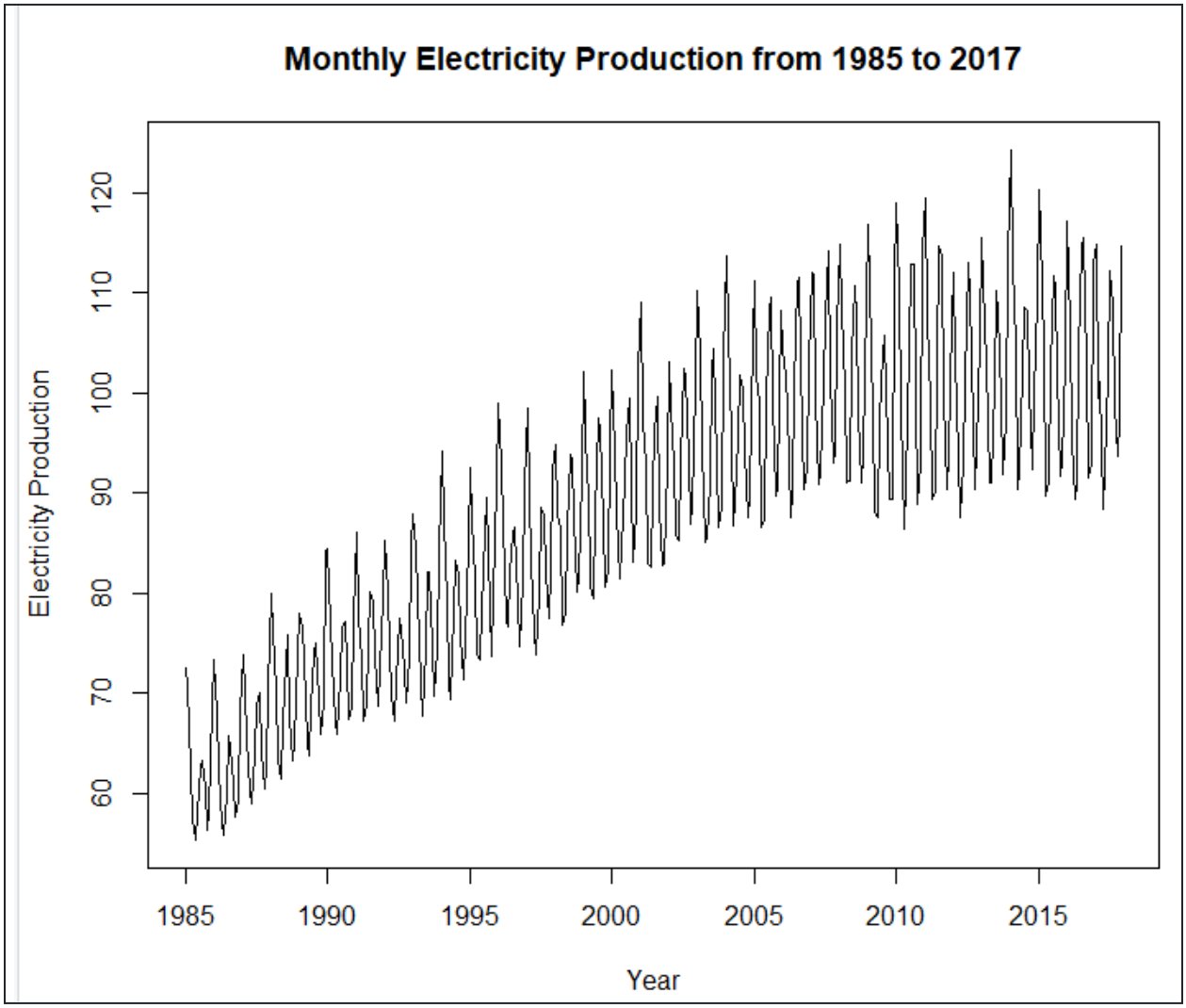

Figure 5.3.3.4

Our time series data are plotted in Figure 5.3.3.4 using the frequency of 12 and the 'Value' column because it is a visualization of monthly data. Our time series data begin in January 1985 and ends in December 2017.

```
> ggseasonplot(y, year.labels=TRUE, year.labels.left=TRUE) +
+   ylab("Electricity Production") +
+   ggtitle("Seasonal plot: Monthly Electricity Production in United State")
>
```

Figure 5.3.3.5

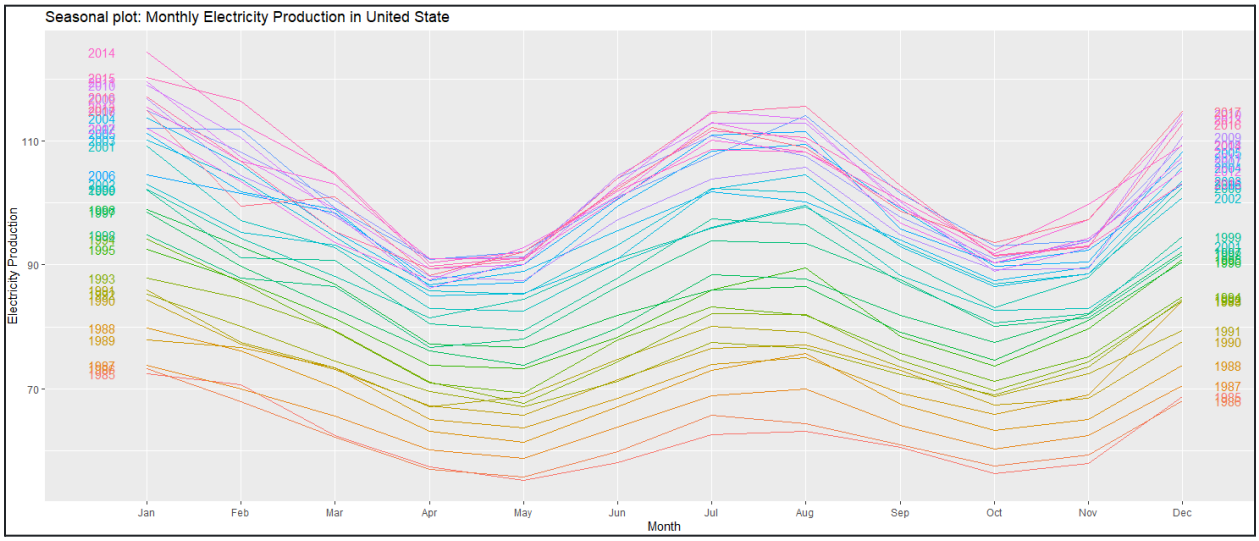

Figure 5.3.3.6 Seasonplot

From figure 5.3.3.6, the seasonal plot indicates that there is a seasonal pattern in our time series data.

5.3.4 Check For Duplicate

```
cat("There are ",length(unique(data[["DATE"]])),"  days of record in the dataset")
```

Figure 5.3.4.1

```
> cat("There are ",length(unique(data[["DATE"]])),"  days of record in the dataset")
There are  397   days of record in the dataset
```

Figure 5.3.4.2

In Figure 5.3.4.2, the number of unique values in the attribute "DATE" is checked after the density of the attribute "Value" is examined. There are 397 unique dates in all, so there are no duplicate values in our time series dataset that would affect our analysis going forward.

5.4 Split Into Training And Testing Set

```
# Use train data from 1985 to 2007 for forecasting
train = window(y, start=1985, end=c(2007,12))
```

Figure 5.4.1 Split Time series data into training set

```
# Use remaining data from 2008 to 2018 to test accuracy
test = window(y, start=2008, end=c(2017,12))
```

Figure 5.4.2 Split Time series data into testing set

Figure 5.4.1 and Figure 5.4.2 shows that we split our data into training sets and testing sets. The training set is used to feed and build the forecasting model. After the model is built, we will use the model to forecast the test data.

5.5 Decomposition

In order to share the observed value, trend, seasonal, and residual components in the time series, the addictive model is used in this project, as shown in Figure 5.5.1 below.

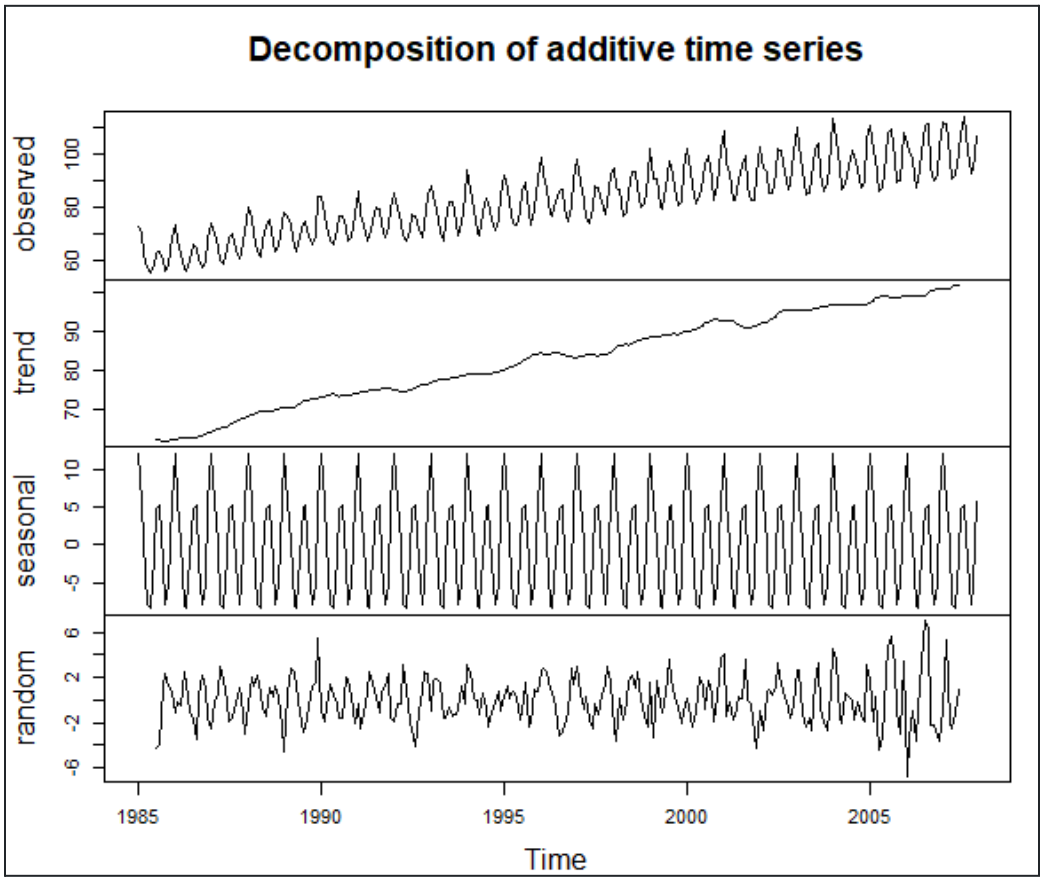

Figure 5.5.1 Decomposition of Additive Time Series

Figure 5.5.1 illustrates the Decomposition of Additive Time Series before the process of differencing. We can see that the time series data has an increasing trend from the figure above. Additionally, the largest increase occurred after 1987 and slowed down in 2007. The production of electricity follows a seasonal pattern in our data, as shown in Figure 5.5.1. By using the method of "Testing the Stationary of Time Series," we can see that a stationary process' mean and variance do not change over time and that the process exhibits a trend. The "Dickey-Fuller test" will be applied to evaluate the stationary as shown in the below section because the time series depicted above does not seem to be stationary.

```
        Augmented Dickey-Fuller Test

data:  train
Dickey-Fuller = -7.5883, Lag order = 6, p-value = 0.01
alternative hypothesis: stationary

Warning message:
In adf.test(train) : p-value smaller than printed p-value
```

Figure 5.5.2 Augmented Dickey-Fuller Test

1. H_o : The time series is not stationary due to certain time dependent structure and inconsistent variance over time.
2. H_1 : The time series is stationary.

The above figure 5.5.2 illustrates how we used the Augmented Dickey-Fuller Test and hypothesis testing to determine whether or not our time series dataset was stationary before differentiating. Since the p-value for the result is 0.01 in Figure 5.5.2 is less than 0.05, we reject H0 and determine that the time series is stationary. However, we can see from Figure 5.5.1 Decomposition of Additive Time Series that the time series' trend is upward and that there is a seasonality pattern in the data. Since the Augmented Dickey-Fuller Test (ADF Test) is used to determine whether or not a time series data that is not seasonal is stationary, we can draw the conclusion that the ADF Test is inappropriate for testing this particular time series data because it is seasonal. Therefore, we would like to introduce another statistical test, the Kwiatkowski-Phillips-Schmidt-Shin (KPSS) test, to demonstrate that the time series data is not stationary.

```
        KPSS Test for Level Stationarity

data:  train
KPSS Level = 4.53, Truncation lag parameter = 5, p-value = 0.01
```

Figure 5.5.3 Kwiatkowski-Phillips-Schmidt-Shin (KPSS) Test

1. H_o : The time series is trend stationary
2. H_1 : The time series is not trend stationary

The above figure 5.5.3 illustrates how we used the Kwiatkowski-Phillips-Schmidt-Shin (KPSS) test and hypothesis testing to determine whether or not our time series dataset was trend stationary before differentiating. We reject H0 because the result in Figure 5.5.3, where p-value = 0.01, is less than 0.05, and we draw the conclusion that the time series is not stationary. We must perform differencing in order to produce a stationary time series data because the time series is not stationary.After testing the stationary of the time series, ACF and PACF are used to measure and plot the average correlation between data points in a time series and previous values of the series measured.

Firstly, ACF is used to measure and plot the average correlation between datapoints in a time series and previous values of the series measured as shown in Figure 5.5.5.

```
> acf(train,main='Autocorrelations')
```

Figure 5.5.4 ACF

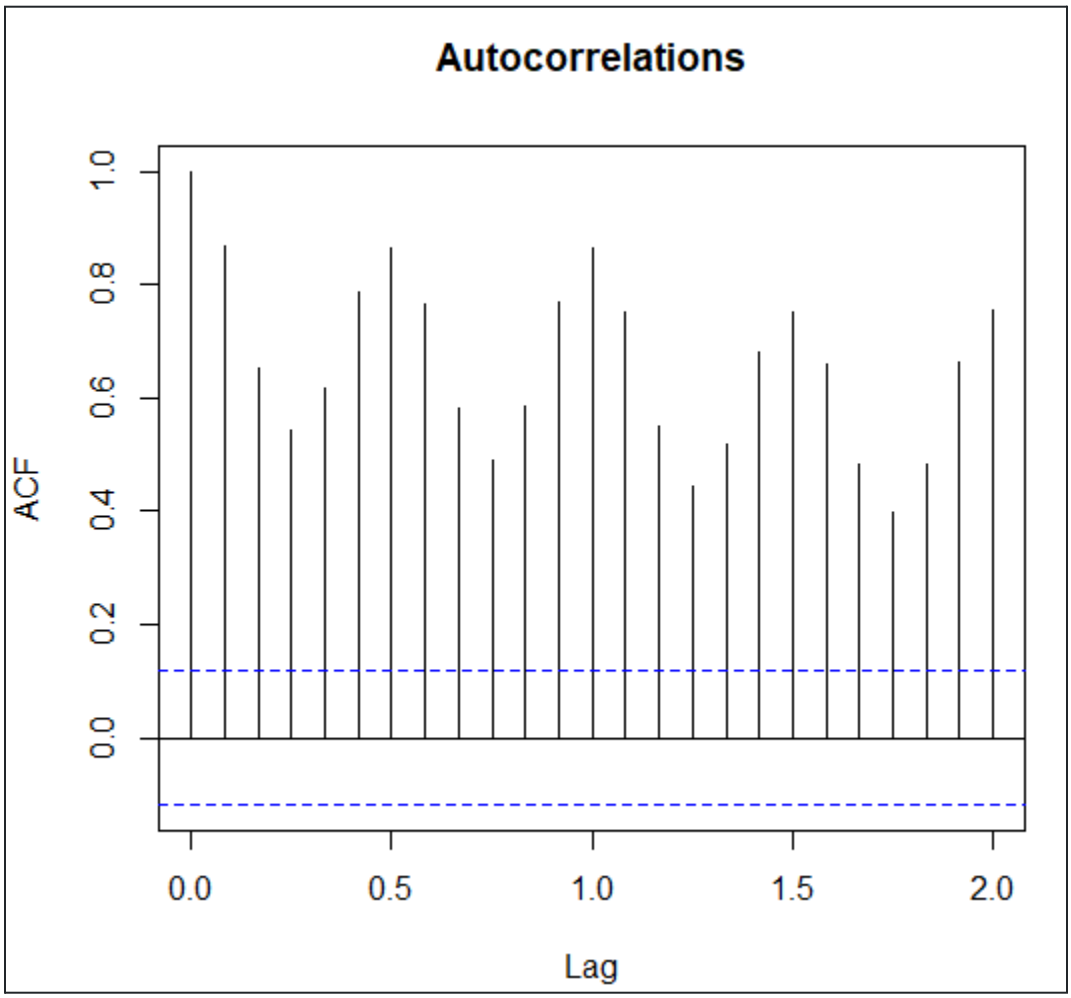

Figure 5.5.5

Lastly,the PACF is used to measure and plot the average correlation between data points in a time series and previous values of the series is measured as shown in Figure 5.5.7.

```
> pacf(y,main='Partial Autocorrelations')
```

Figure 5.4.6 PACF

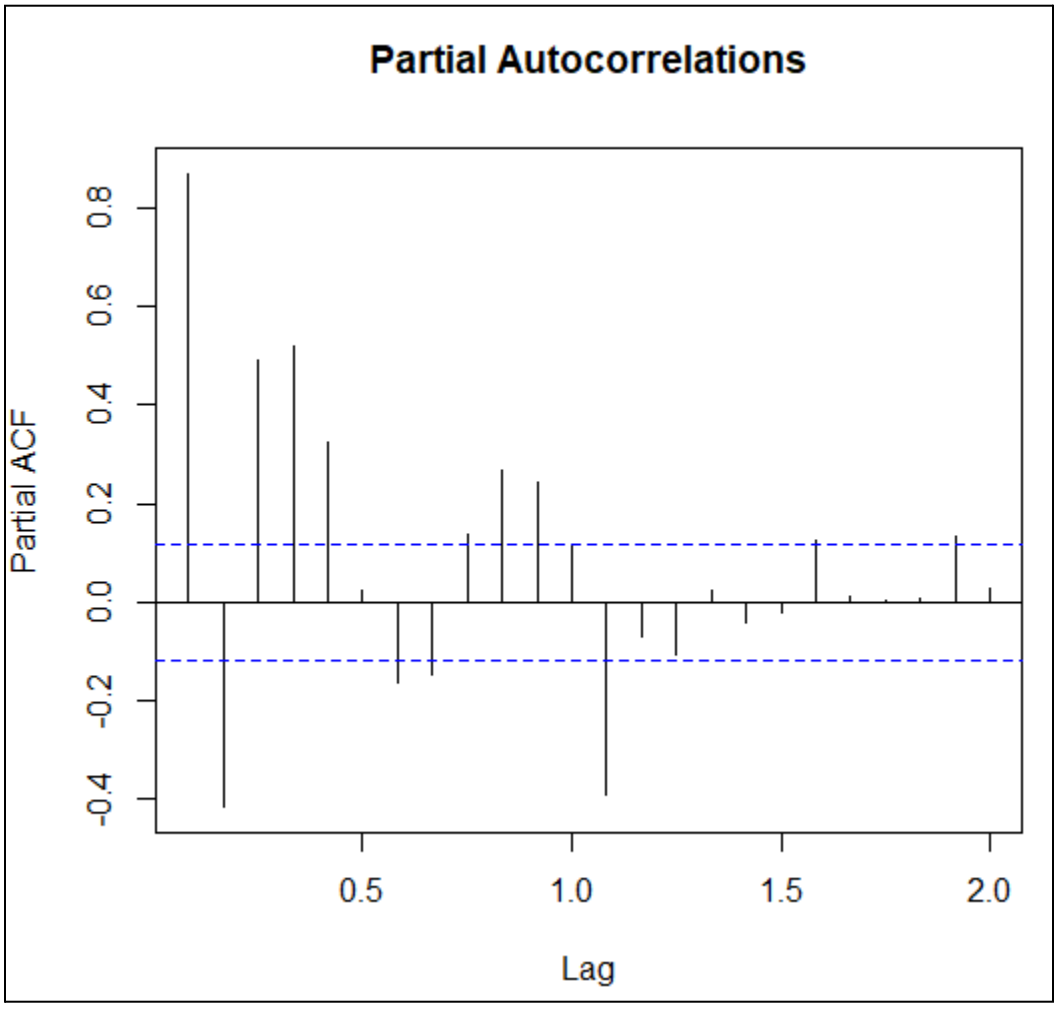

Figure 5.5.7

5.6 Differencing

5.6.1 Seasonal Differencing

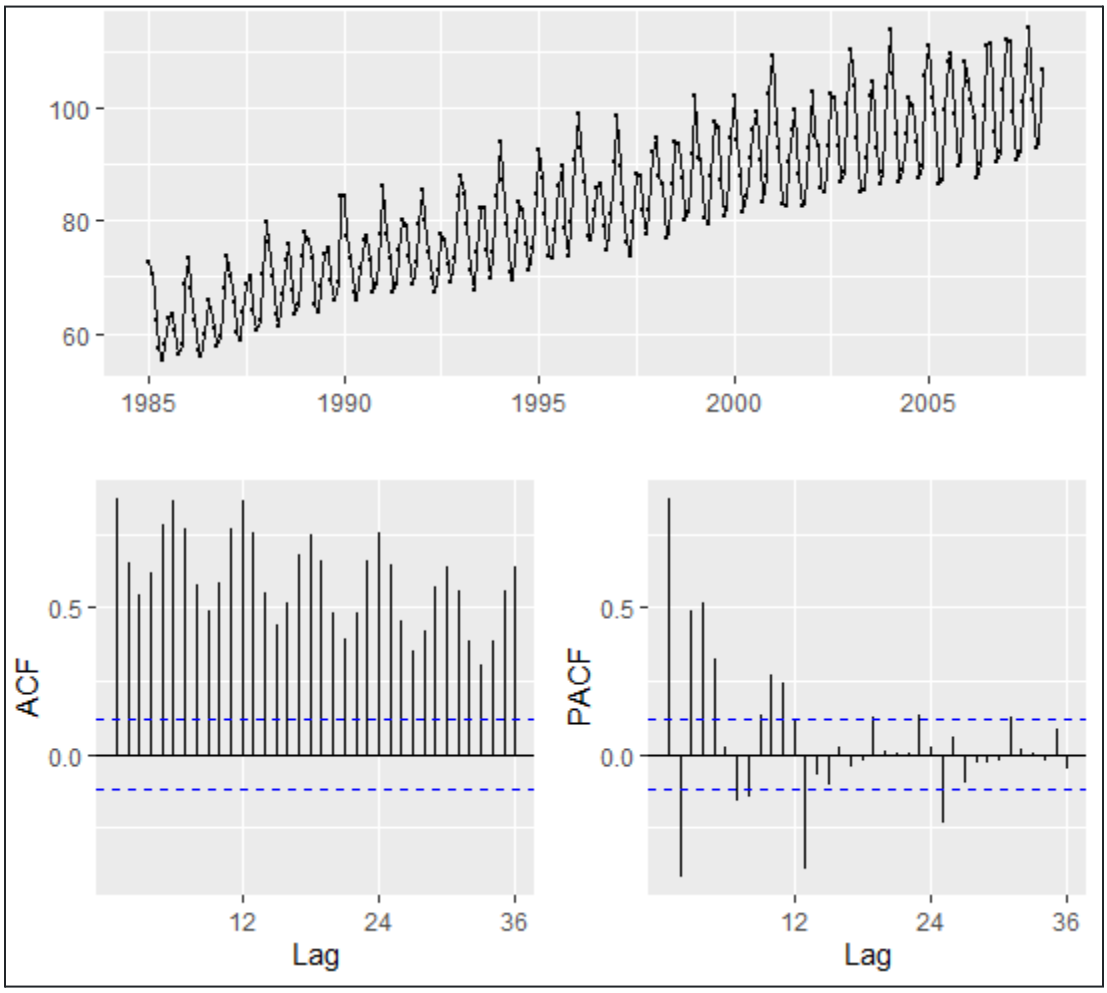

Figure 5.6.1.1 Correlogram before differencing

From figure 5.6.1.1, we can see our data has some seasonal effect as the value in ACF spikes at every lag 12, 24, 36.To obtain a stationary time series, seasonal differencing is performed on the train dataset of the time series. After performing the differencing, the result is shown by plotting the graph using the stationary time series obtained by differencing as shown in Figure 5.6.1.3 .

```
> #doing seasonal differencing
> seasonalDiff <- diff(train,lag=12)
> plot(seasonalDiff)
```

Figure 5.6.1.2

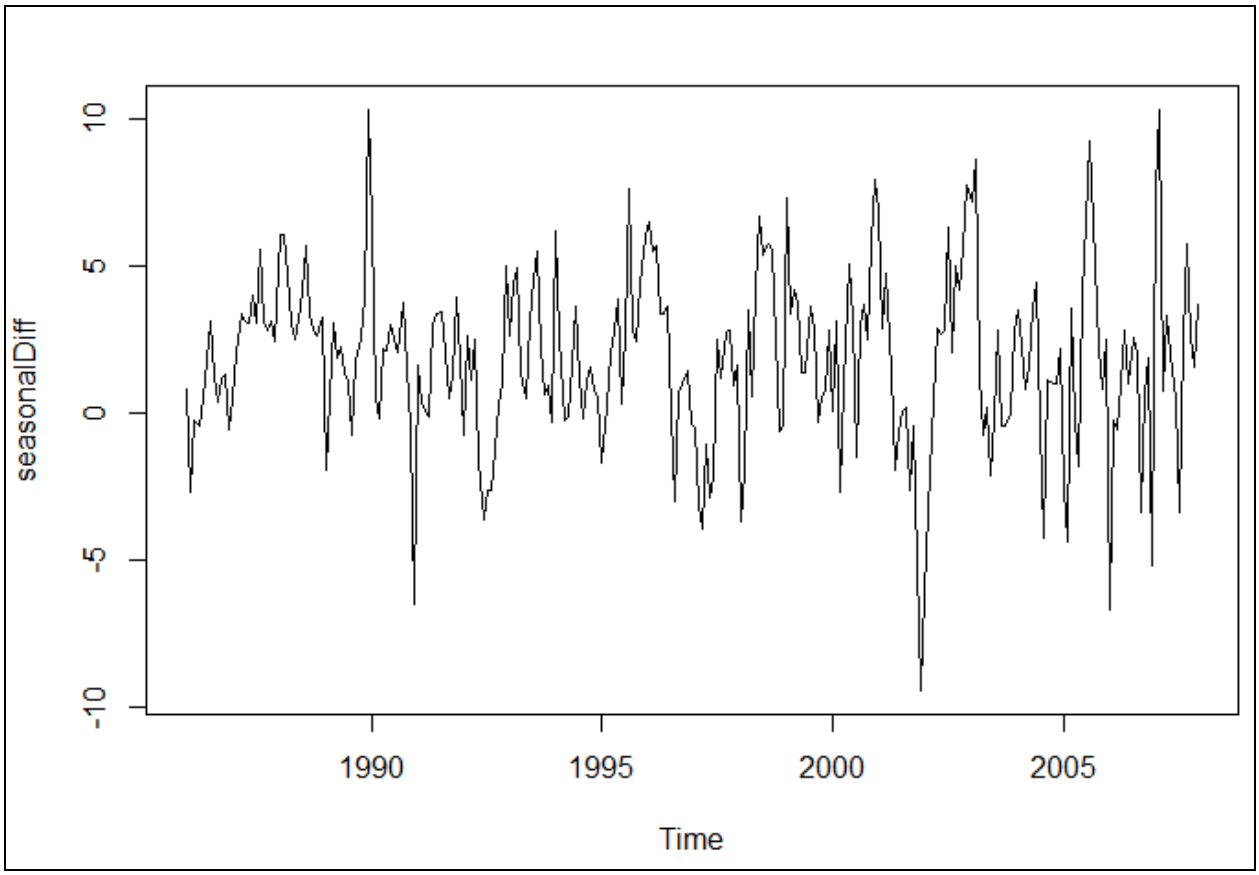

Figure 5.6.1.3 Graph for Visualizing The Stationary Time Series

5.6.2 Stationary Testing With KPSS

```
> kpss.test(seasonalDiff)

        KPSS Test for Level Stationarity

data:  seasonalDiff
KPSS Level = 0.058486, Truncation lag parameter = 5, p-value = 0.1

Warning message:
In kpss.test(seasonalDiff) : p-value greater than printed p-value
```

Figure 5.6.2.1 Kwiatkowski-Phillips-Schmidt-Shin (KPSS) Test After 1st Differencing

1. H_o : The time series is trend stationary
2. H_1 : The time series is not trend stationary

We do not reject H0 because the result in Figure 5.6.2.1, where p-value = 0.1, is more than 0.05, and we draw the conclusion to conclude that the time series is stationary. Hence, We do not need

to move on to the second differencing because the time series is stationary after the first differencing.

5.6.3 Seasonal Differenced Series, ACF and PACF

The ACF and PACF are used to make a manual guess on the modernity of our ARIMA model (p,d,q)(P,D,Q)[12].

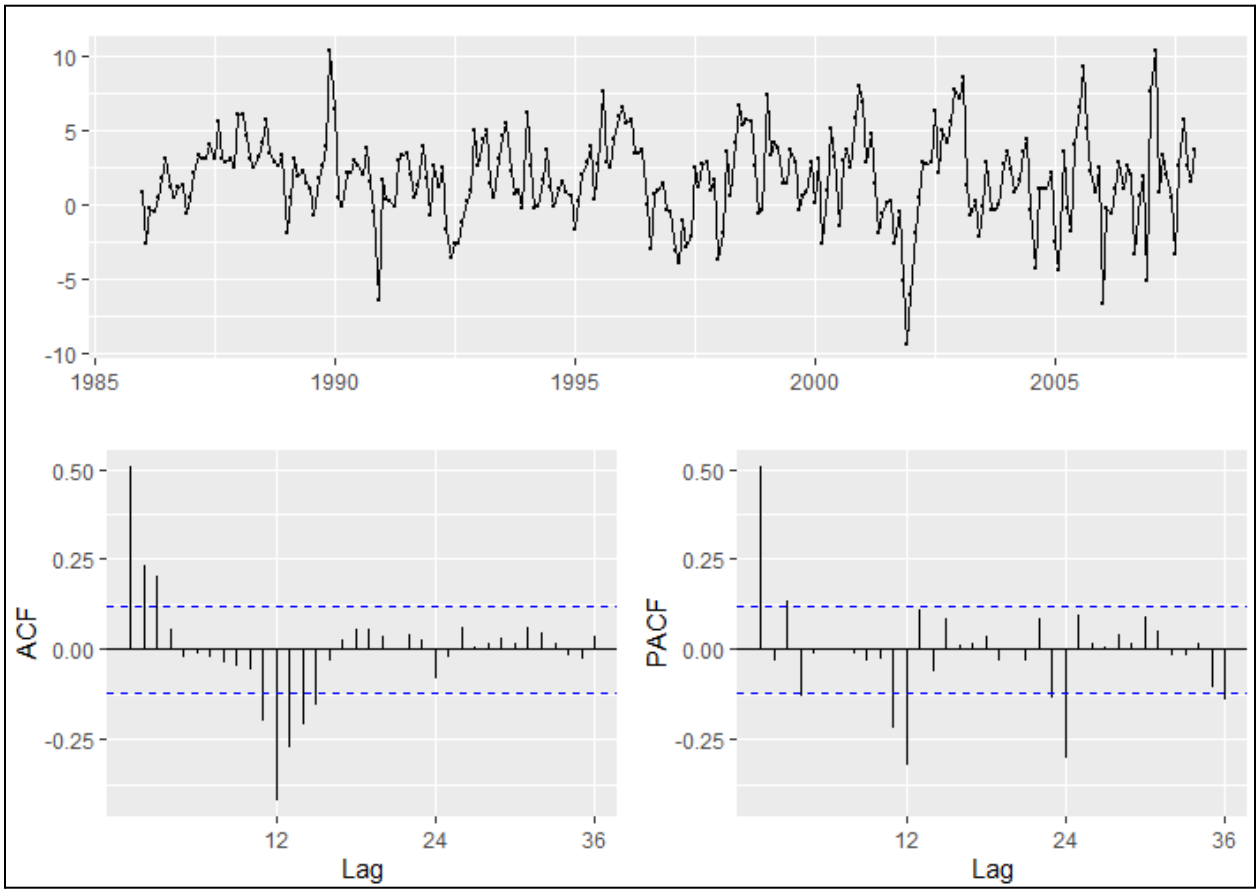

Figure 5.6.3.1 Correlogram after differencing

6.0 Results

6.1 SARIMA

The first model we will be using is the SARIMA model. From Figure 5.6.3.1, by observing the ACF and PACF of the time series data, we can simply guess the pdq and PDQ of the SARIMA model. Based on ACF, it is showing a slowing decaying pattern, so the q and Q will be set to 0. Besides, we only do one times seasonal differencing for the data so the d = 0 and D = 1. Lastly, from the PACF diagram we can see that only the first spike is obvious so p = 1 and at lag 12 and lag 24, there are also spikes on them, so P = 2. As conclusion, our manually guess SARIMA model will be ARIMA(1,0,0)(2,1,0)[12]

```
> sarimaTrain_man <- arima(train,c(1,0,0),seasonal = list(order=c(2,1,0),period=12))
> summary(sarimaTrain_man)

Call:
arima(x = train, order = c(1, 0, 0), seasonal = list(order = c(2, 1, 0), period = 12))

Coefficients:
         ar1     sar1     sar2
      0.8280  -0.5233  -0.3847
s.e.  0.0373   0.0642   0.0662

sigma^2 estimated as 5.473:  log likelihood = -602.34,  aic = 1212.69

Training set error measures:
                    ME     RMSE      MAE       MPE     MAPE      MASE       ACF1
Training set 0.5544311 2.288043 1.707054 0.6434009 1.986573 0.2922309 -0.1976098
>
```

Figure 6.1.1 : ARIMA(1,0,0)(2,1,0)[12]

From Figure 6.1.1, we can observe the coefficient of the model which are:

The model's coefficient information is as below :

ar1, $\phi_1 = 0.828$

sar1,$\Phi_1 =$ -0.5233

sar2,$\Phi_2 = -0.3847$

```
> coeftest(sarimaTrain_man)

z test of coefficients:

      Estimate Std. Error z value  Pr(>|z|)
ar1   0.827981   0.037343 22.1723 < 2.2e-16 ***
sar1 -0.523342   0.064204 -8.1512 3.603e-16 ***
sar2 -0.384684   0.066185 -5.8123 6.163e-09 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Figure 6.1.2 : Coefficient test of ARIMA(1,0,0)(2,1,0)[12]

After identifying the coefficient of the model we would like to test the significance of the coefficient of the model.

1. H_o : Coefficient is not significant
2. H_1 : Coefficient is significant

From the figure above, we can see that the p-value of ar1, sar1, sar2 and c are 2.2e-16 (0.00000000000000022), 3.603e-19 (0.000000000000000003603) and 6.163e-09 (0.000000006163) respectively. Based on the figure above, it shows that all p-values are lower than $\alpha = 0.05$, therefore we successfully reject H0 and conclude that all 3 coefficients are significant.

```
> Box.test(sarimaTrain_man$residuals,lag=12)

        Box-Pierce test

data:  sarimaTrain_man$residuals
X-squared = 37.774, df = 12, p-value = 0.0001672
```

Figure 6.1.3: Box Test of ARIMA(1,0,0)(2,1,0)[12]

Furthermore, we will use the Box-Pierce test to determine whether the ARIMA(1,0,0)(2,1,0)[12] model's residual shows a white noise pattern or not. This is due the fact that if there are any signals in the model's residuals, we must update the model otherwise the forecasting process may not be successful.

Hypothesis of Box-Pierce test

1. H_o : The ARIMA(1,0,0)(2,1,0)[12] model does not show a lack of fit.
2. H_1 : The ARIMA(1,0,0)(2,1,0)[12] model does show a lack of fit.

Figure 6.1.3 shows that the p-value is equal to 0.00001672, which is smaller than 0.05 and indicates that we reject the null hypothesis and conclude that the ARIMA(1,0,0)(2,1,0)[12] model does show a lack of fit and does not have a white noise in its residuals.

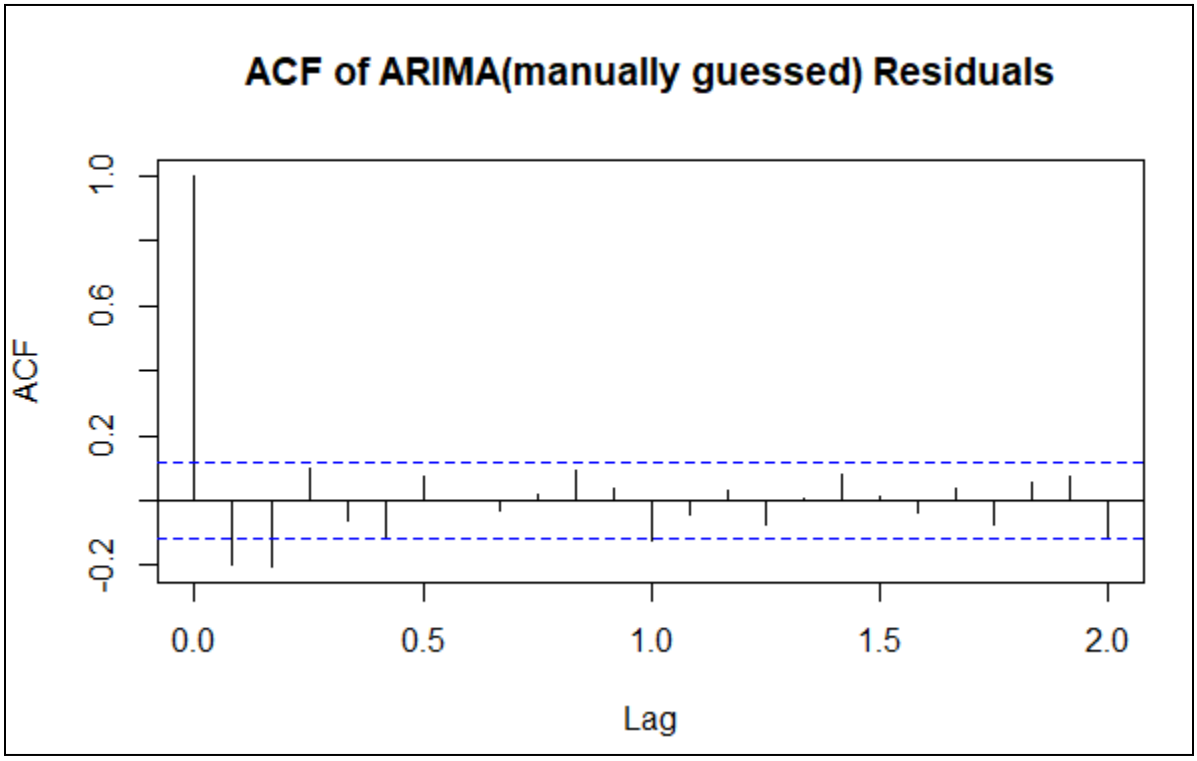

Figure 6.1.4: ACF of Residuals of ARIMA(1,0,0)(2,1,0)[12]

From the ACF, we notice that there are two obvious spikes exceeding the blue dotted line which are the second and third spikes which means that the residuals of the model do not show a white noise. Hence, from the ACF we can get the same conclusions with the Box-Test. So after determining whether the residuals of the models are white noise or not, we will proceed to forecasting the future 10 years of Electric Production with ARIMA(1,0,0)(2,1,0)[12] model.

```
> forecast_sarimaTrain_man <- forecast(sarimaTrain_man,h=120)
> print(forecast_sarimaTrain_man)
         Point Forecast     Lo 80     Hi 80     Lo 95     Hi 95
Jan 2008      112.27400 109.27590 115.27210 107.68880 116.85920
Feb 2008      107.87429 103.98189 111.76668 101.92138 113.82719
Mar 2008      100.09779  95.69606 104.49952  93.36593 106.82966
Apr 2008       89.60492  84.88567  94.32417  82.38745  96.82239
May 2008       90.67430  85.74918  95.59941  83.14198  98.20661
Jun 2008      100.97215  95.91074 106.03356  93.23139 108.71290
Jul 2008      108.85780 103.70504 114.01057 100.97733 116.73827
Aug 2008      112.39929 107.18482 117.61376 104.42445 120.37413
Sep 2008      100.15859  94.90224 105.41494  92.11970 108.19749
Oct 2008       91.68643  86.40156  96.97130  83.60392  99.76895
Nov 2008       92.61243  87.30809  97.91676  84.50014 100.72471
Dec 2008      107.01042 101.69278 112.32806  98.87779 115.14305
Jan 2009      109.41028 103.81511 115.00545 100.85321 117.96736
Feb 2009      106.11824 100.34050 111.89598  97.28196 114.95453
Mar 2009       99.43459  93.53496 105.33423  90.41189 108.45730
Apr 2009       89.06197  83.08021  95.04373  79.91366  98.21029
May 2009       90.69410  84.65667  96.73152  81.46066  99.92754
Jun 2009      100.86443  94.78915 106.93972  91.57308 110.15578
Jul 2009      109.53794 103.43684 115.63905 100.20711 118.86878
Aug 2009      112.34172 106.22297 118.46046 102.98391 121.69953
Sep 2009       98.69512  92.56431 104.82593  89.31886 108.07138
Oct 2009       91.39519  85.25612  97.53425  82.00630 100.78407
Nov 2009       92.71965  86.57493  98.86436  83.32211 102.11718
Dec 2009      105.47834  99.32975 111.62693  96.07488 114.88179
Jan 2010      110.84312 104.56321 117.12303 101.23882 120.44742
Feb 2010      108.57715 102.20878 114.94553  98.83756 118.31674
Mar 2010       99.44534  93.01702 105.87366  89.61407 109.27660
Apr 2010       89.82252  83.35343  96.29161  79.92890  99.71614
May 2010       91.22446  84.72756  97.72136  81.28832 101.16061
Jun 2010      100.92582  94.40993 107.44171  90.96063 110.89101
Jul 2010      108.69166 102.16278 115.22054  98.70660 118.67672
Aug 2010      113.03211 106.49434 119.56987 103.03345 123.03076
Sep 2010       99.99303  93.44917 106.53688  89.98507 110.00099
Oct 2010       92.05871  85.51069  98.60673  82.04437 102.07305
Nov 2010       93.16632  86.61544  99.71720  83.14761 103.18503
Dec 2010      106.18322  99.63038 112.73606  96.16151 116.20492
```

Figure 6.1.5: Forecast results by ARIMA(1,0,0)(2,1,0)[12] for 2008-2010

Jan 2011	111.19669	104.34340	118.04997	100.71549	121.67788
Feb 2011	107.96732	100.91546	115.01918	97.18243	118.75221
Mar 2011	99.69607	92.51125	106.88090	88.70783	110.68432
Apr 2011	89.63438	82.35980	96.90896	78.50887	100.75989
May 2011	90.94013	83.60466	98.27560	79.72149	102.15877
Jun 2011	100.93583	93.55890	108.31276	89.65379	112.21787
Jul 2011	108.87350	101.46828	116.27871	97.54820	120.19880
Aug 2011	112.69343	105.26888	120.11797	101.33857	124.04829
Sep 2011	99.87715	92.43939	107.31492	88.50207	111.25223
Oct 2011	91.82383	84.37701	99.27064	80.43490	103.21275
Nov 2011	92.89159	85.43857	100.34460	81.49318	104.28999
Dec 2011	106.40392	98.94666	113.86118	94.99902	117.80882
Jan 2012	110.46065	102.78838	118.13292	98.72693	122.19437
Feb 2012	107.34072	99.52448	115.15697	95.38681	119.29464
Mar 2012	99.56085	91.64741	107.47429	87.45829	111.66341
Apr 2012	89.44038	81.46099	97.41976	77.23696	101.64379
May 2012	90.88500	82.86072	98.90928	78.61292	103.15708
Jun 2012	100.90705	92.85213	108.96196	88.58812	113.22598
Jul 2012	109.10395	101.02810	117.17980	96.75300	121.45489
Aug 2012	112.60514	104.51497	120.69531	100.23229	124.97799
Sep 2012	99.43855	91.33858	107.53853	87.05071	111.82639
Oct 2012	91.69154	83.58485	99.79822	79.29343	104.08965
Nov 2012	92.86357	84.75228	100.97485	80.45843	105.26871
Dec 2012	106.01728	97.90285	114.13172	93.60732	118.42724
Jan 2013	110.70986	102.43343	118.98628	98.05216	123.36756
Feb 2013	107.90325	99.51759	116.28892	95.07848	120.72803
Mar 2013	99.53518	91.07543	107.99492	86.59711	112.47324
Apr 2013	89.61429	81.10414	98.12445	76.59913	102.62946
May 2013	91.02324	82.47869	99.56778	77.95548	104.09099
Jun 2013	100.91827	92.35023	109.48631	87.81458	114.02196
Jul 2013	108.91340	100.32929	117.49751	95.78513	122.04166
Aug 2013	112.78163	104.18652	121.37674	99.63655	125.92672
Sep 2013	99.71267	91.11003	108.31531	86.55606	112.86928
Oct 2013	91.85113	83.24333	100.45893	78.68663	105.01563
Nov 2013	92.98392	84.37258	101.59526	79.81401	106.15383
Dec 2013	106.13473	97.52097	114.74849	92.96112	119.30834

Figure 6.1.6: Forecast results by ARIMA(1,0,0)(2,1,0)[12] for 2011-2013

Jan 2014	110.86258	102.06300	119.66216	97.40478	124.32038
Feb 2014	107.84990	98.92517	116.77464	94.20070	121.49911
Mar 2014	99.60063	90.59110	108.61016	85.82175	113.37952
Apr 2014	89.59791	80.53071	98.66511	75.73082	103.46500
May 2014	90.97210	81.86557	100.07863	77.04486	104.89934
Jun 2014	100.92347	91.79008	110.05686	86.95515	114.89179
Jul 2014	108.92447	99.77271	118.07623	94.92806	122.92089
Aug 2014	112.72323	103.55889	121.88757	98.70759	126.73887
Sep 2014	99.73794	90.56499	108.91088	85.70913	113.76674
Oct 2014	91.81850	82.63966	100.99734	77.78067	105.85633
Nov 2014	92.93171	83.74883	102.11460	78.88770	106.97572
Dec 2014	106.22200	97.03634	115.40765	92.17375	120.27024
Jan 2015	110.68679	101.33080	120.04278	96.37804	124.99554
Feb 2015	107.66143	98.19043	117.13242	93.17679	122.14606
Mar 2015	99.57625	90.02722	109.12529	84.97226	114.18024
Apr 2015	89.53958	79.93741	99.14175	74.85433	104.22483
May 2015	90.94568	81.30726	100.58411	76.20498	105.68639
Jun 2015	100.91643	91.25323	110.57964	86.13783	115.69503
Jul 2015	108.99198	99.31182	118.67213	94.18746	123.79650
Aug 2015	112.68590	102.99414	122.37766	97.86364	127.50817
Sep 2015	99.61926	89.91956	109.31897	84.78485	114.45368
Oct 2015	91.77418	82.06903	101.47933	76.93144	106.61693
Nov 2015	92.91274	83.20386	102.62162	78.06429	107.76119
Dec 2015	106.13115	96.41971	115.84258	91.27879	120.98351
Jan 2016	110.72004	100.85688	120.58320	95.63564	125.80444
Feb 2016	107.78059	97.81475	117.74643	92.53915	123.02202
Mar 2016	99.56383	89.52821	109.59946	84.21567	114.91199
Apr 2016	89.57641	79.49322	99.65960	74.15551	104.99731
May 2016	90.97918	80.86352	101.09484	75.50861	106.44975
Jun 2016	100.91811	90.78024	111.05598	85.41358	116.42265
Jul 2016	108.95239	98.79933	119.10545	93.42462	124.48016
Aug 2016	112.72790	102.56444	122.89137	97.18422	128.27159
Sep 2016	99.67165	89.50106	109.84224	84.11707	115.22623
Oct 2016	91.80993	81.63445	101.98540	76.24788	107.37198
Nov 2016	92.94275	82.76393	103.12157	77.37559	108.50991
Dec 2016	106.14512	95.96401	116.32624	90.57445	121.71579

Figure 6.1.7: Forecast results by ARIMA(1,0,0)(2,1,0)[12] for 2014-2016

Jan 2017	110.77026	100.43817	121.10235	94.96869	126.57183
Feb 2017	107.79073	97.35640	118.22506	91.83279	123.74866
Mar 2017	99.57971	89.07586	110.08356	83.51546	115.64396
Apr 2017	89.57957	79.02833	100.13082	73.44284	105.71631
May 2017	90.97181	80.38820	101.55542	74.78558	107.15805
Jun 2017	100.91994	90.31420	111.52568	84.69986	117.14002
Jul 2017	108.94714	98.32625	119.56803	92.70389	125.19039
Aug 2017	112.72028	102.08902	123.35154	96.46117	128.97939
Sep 2017	99.68989	89.05152	110.32825	83.41991	115.95986
Oct 2017	91.80827	81.16504	102.45150	75.53085	108.08569
Nov 2017	92.93434	82.28778	103.58091	76.65182	109.21686
Dec 2017	106.17276	95.52390	116.82161	89.88674	122.45877

Figure 6.1.8: Forecast results by ARIMA(1,0,0)(2,1,0)[12] for 2017

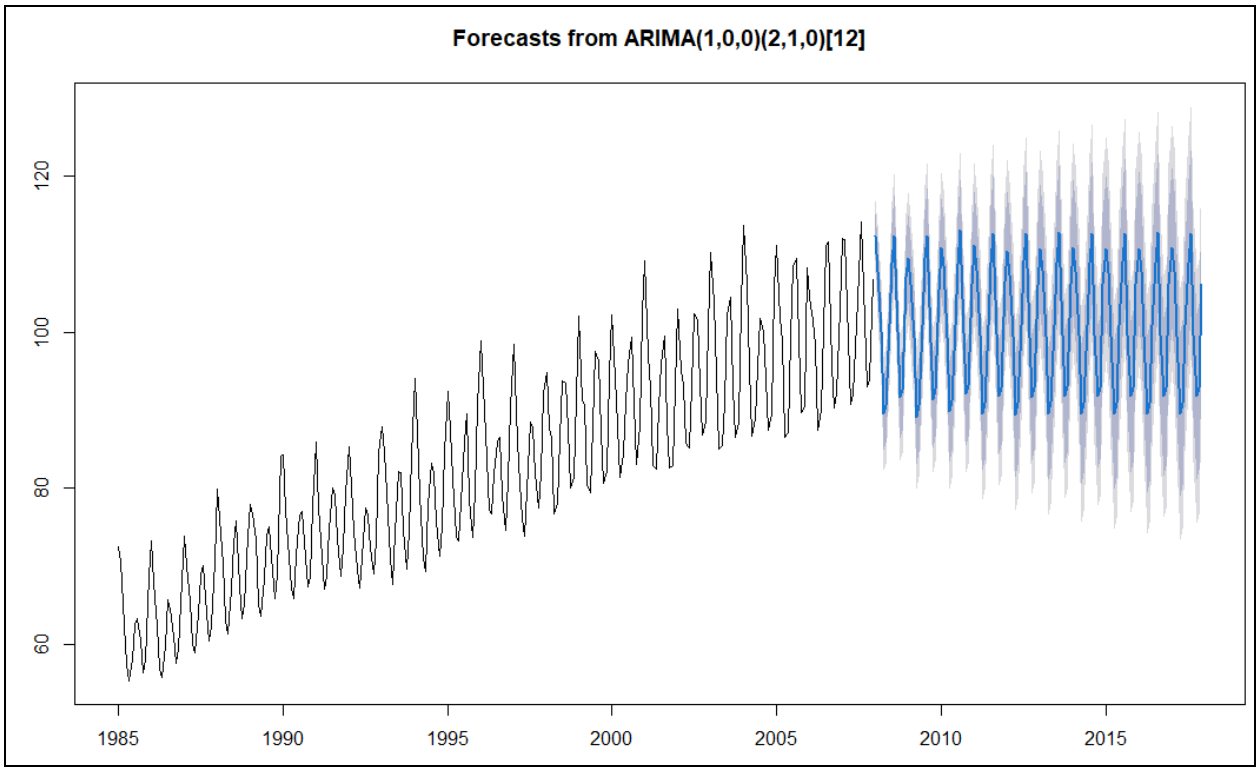

Figure 6.1.9: Forecast results by ARIMA(1,0,0)(2,1,0)[12]

Since our test data is from 2008 - 2017,which is 10 years, then we will use the forecasting model to forecast for 10 years by using h =120 which means we are doing forecasting for 120 months (10 years).

```
> error=test-forecast_sarimaTrain_man$mean
> #(error)
> mse = mean(error*error)
> print(mse)
[1] 14.27835
> rmse = sqrt(mse)
> print(rmse)
[1] 3.778671
>
```

Figure 6.1.10: RMSE and MSE of ARIMA(1,0,0)(2,1,0)[12]

Figure 6.1.10 shows how we calculate the Root Mean Square Error (RMSE) for the ARIMA model, we will use the test data to minus the forecasted value by the model to get the value of error. Then we will calculate the Mean Square Error (MSE) by getting the mean of error multiplied by error (error^2). Lastly we will square root the MSE and the value of RMSE of the model is 3.778671. We will not use the accuracy(),a built-in function in R, to evaluate the

performance of the model as the accuracy as the function evaluates the model based on the train data. By right, to avoid overfitting, we will evaluate the model based on test data, hence we are comparing the predicted test data and actual value of test data to calculate the RMSE as the evaluation of the model.

6.2 ARIMA Suggested By The Auto.arima

Auto.arima is a built-in function in R, which it will determine the best ARIMA model based on our time series data in term of lower values of AIC.

```
> arimaTrain <- auto.arima(train,ic="aic",trace=TRUE)

 Fitting models using approximations to speed things up...

 ARIMA(2,0,2)(1,1,1)[12] with drift         : Inf
 ARIMA(0,0,0)(0,1,0)[12] with drift         : 1290.787
 ARIMA(1,0,0)(1,1,0)[12] with drift         : 1177.123
 ARIMA(0,0,1)(0,1,1)[12] with drift         : 1128.968
 ARIMA(0,0,0)(0,1,0)[12]                    : 1374.215
 ARIMA(0,0,1)(0,1,0)[12] with drift         : 1218.023
 ARIMA(0,0,1)(1,1,1)[12] with drift         : 1140.514
 ARIMA(0,0,1)(0,1,2)[12] with drift         : 1130.8
 ARIMA(0,0,1)(1,1,0)[12] with drift         : 1177.024
 ARIMA(0,0,1)(1,1,2)[12] with drift         : 1139.02
 ARIMA(0,0,0)(0,1,1)[12] with drift         : 1194.784
 ARIMA(1,0,1)(0,1,1)[12] with drift         : 1124.189
 ARIMA(1,0,1)(0,1,0)[12] with drift         : 1216.513
 ARIMA(1,0,1)(1,1,1)[12] with drift         : 1133.869
 ARIMA(1,0,1)(0,1,2)[12] with drift         : 1126.112
 ARIMA(1,0,1)(1,1,0)[12] with drift         : 1177.699
 ARIMA(1,0,1)(1,1,2)[12] with drift         : 1132.049
 ARIMA(1,0,0)(0,1,1)[12] with drift         : 1122.224
 ARIMA(1,0,0)(0,1,0)[12] with drift         : 1215.113
 ARIMA(1,0,0)(1,1,1)[12] with drift         : 1131.924
 ARIMA(1,0,0)(0,1,2)[12] with drift         : 1124.157
 ARIMA(1,0,0)(1,1,2)[12] with drift         : 1130.174
 ARIMA(2,0,0)(0,1,1)[12] with drift         : 1121.931
 ARIMA(2,0,0)(0,1,0)[12] with drift         : 1215.285
 ARIMA(2,0,0)(1,1,1)[12] with drift         : 1134.082
 ARIMA(2,0,0)(0,1,2)[12] with drift         : 1123.791
 ARIMA(2,0,0)(1,1,0)[12] with drift         : 1179.529
 ARIMA(2,0,0)(1,1,2)[12] with drift         : 1132.921
 ARIMA(3,0,0)(0,1,1)[12] with drift         : 1122.291
 ARIMA(2,0,1)(0,1,1)[12] with drift         : 1120.464
 ARIMA(2,0,1)(0,1,0)[12] with drift         : 1213.115
 ARIMA(2,0,1)(1,1,1)[12] with drift         : 1135.595
```

Figure 6.2.1: Result of auto.arima

```
ARIMA(2,0,1)(0,1,2)[12] with drift         : 1122.271
ARIMA(2,0,1)(1,1,0)[12] with drift         : 1176.644
ARIMA(2,0,1)(1,1,2)[12] with drift         : 1131.964
ARIMA(3,0,1)(0,1,1)[12] with drift         : 1122.831
ARIMA(2,0,2)(0,1,1)[12] with drift         : 1121.192
ARIMA(1,0,2)(0,1,1)[12] with drift         : 1124.277
ARIMA(3,0,2)(0,1,1)[12] with drift         : 1124.83
ARIMA(2,0,1)(0,1,1)[12]                    : Inf

Now re-fitting the best model(s) without approximations...

ARIMA(2,0,1)(0,1,1)[12] with drift         : 1155.066

Best model: ARIMA(2,0,1)(0,1,1)[12] with drift
```

Figure 6.2.2 : Result of auto.arima

Both Figure 6.2.1 and Figure 6.2.3 show the result of auto.arima. From Figure 6.2.3, we can see that the suggested best ARIMA model is ARIMA(2,0,1)(0,1,1)[12] with drift. Since the seasonality components are included in the model, the model can also be considered as a SARIMA model.

Next, we will use the Box-Pierce test to test whether the residual of the model is a white noise to make sure that none of the signal has escaped from the model and ended up in the residuals. This is because if there are some signals in the residuals in the model, we need to revise the model or else the model cannot be used in the forecasting process as the result may not be very good.

Hypothesis of Box-Pierce test

1. H_o : The ARIMA model does not show a lack of fit.
2. H_1 : The ARIMA model does show a lack of fit.

```
> Box.test(arimaTrain$residuals)

        Box-Pierce test

data:  arimaTrain$residuals
X-squared = 0.063261, df = 1, p-value = 0.8014
```

Figure 6.2.3: Result of Box-Pierce test

From Figure 6.2.3, we can see that the p-value is 0.8014 which is larger than the 0.05, so we fail to reject the null hypothesis and conclude that the model does not show lack of fit. Therefore, we

can proceed to the forecasting process with this ARIMA model suggested by the auto.arima function.

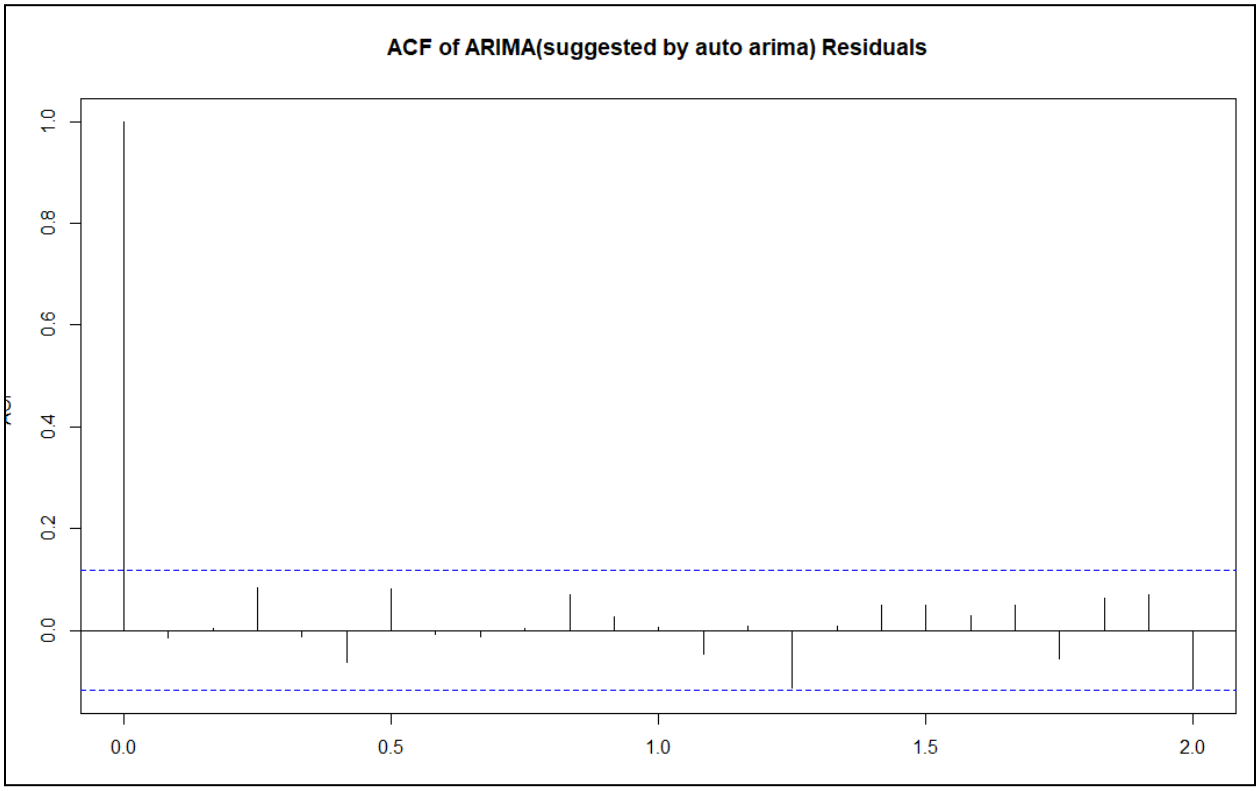

Figure 6.2.4: ACF of the residual of ARIMA(2,0,1)(0,1,1)[12] with drift

Furthermore, from the ACF, we can clearly see that the residuals show a white noise pattern as there is no spike exceed the blue dotted line, which means that the model does not show a lack of fit and is fine to make forecasting for the time series data.

```
> forecast_arimaTrain <- forecast(arimaTrain,h=120)
> print(forecast_arimaTrain)
         Point Forecast     Lo 80     Hi 80     Lo 95     Hi 95
Jan 2008      113.70178 111.01745 116.38610 109.59646 117.80709
Feb 2008      109.43887 106.39411 112.48363 104.78231 114.09543
Mar 2008      102.46096  99.37011 105.55182  97.73391 107.18802
Apr 2008       92.96810  89.86211  96.07410  88.21790  97.71831
May 2008       94.26786  91.16015  97.37557  89.51502  99.02070
Jun 2008      103.28095 100.17253 106.38937  98.52703 108.03486
Jul 2008      111.43822 108.32974 114.54670 106.68421 116.19223
Aug 2008      113.81558 110.70707 116.92409 109.06152 118.56964
Sep 2008      102.32523  99.21672 105.43375  97.57117 107.07930
Oct 2008       95.09929  91.99077  98.20781  90.34522  99.85336
Nov 2008       96.65994  93.55142  99.76846  91.90587 101.41401
Dec 2008      110.05355 106.94503 113.16206 105.29948 114.80761
Jan 2009      116.19290 112.98489 119.40092 111.28667 121.09914
Feb 2009      111.54322 108.30723 114.77920 106.59421 116.49222
Mar 2009      104.40067 101.16088 107.64046  99.44584 109.35550
Apr 2009       94.82733  91.58628  98.06838  89.87057  99.78409
May 2009       96.09377  92.85258  99.33497  91.13679 101.05076
Jun 2009      105.09003 101.84877 108.33128 100.13295 110.04710
Jul 2009      113.24061 109.99935 116.48187 108.28353 118.19769
Aug 2009      115.61442 112.37316 118.85569 110.65734 120.57151
Sep 2009      104.12274 100.88148 107.36401  99.16566 109.07983
Oct 2009       96.89605  93.65478 100.13731  91.93896 101.85313
Nov 2009       98.45644  95.21518 101.69770  93.49935 103.41353
Dec 2009      111.84988 108.60862 115.09115 106.89280 116.80697
Jan 2010      117.98919 114.65240 121.32598 112.88601 123.09238
Feb 2010      113.33947 109.97579 116.70314 108.19517 118.48376
Mar 2010      106.19692 102.82958 109.56425 101.04702 111.34682
Apr 2010       96.62356  93.25501  99.99212  91.47181 101.77532
May 2010       97.89001  94.52131 101.25870  92.73804 103.04198
Jun 2010      106.88626 103.51751 110.25501 101.73420 112.03831
Jul 2010      115.03684 111.66809 118.40560 109.88478 120.18891
Aug 2010      117.41065 114.04190 120.77941 112.25858 122.56272
Sep 2010      105.91897 102.55022 109.28773 100.76691 111.07104
Oct 2010       98.69228  95.32352 102.06103  93.54021 103.84435
Nov 2010      100.25267  96.88391 103.62143  95.10060 105.40474
Dec 2010      113.64611 110.27736 117.01487 108.49404 118.79818
```

Figure 6.2.5: Forecast results by ARIMA(2,0,1)(0,1,1)[12] with drift from 2008-2010

Jan 2011	119.78542	116.32465	123.24619	114.49263	125.07821
Feb 2011	115.13570	111.64900	118.62239	109.80326	120.46814
Mar 2011	107.99315	104.50292	111.48337	102.65530	113.33099
Apr 2011	98.41980	94.92839	101.91120	93.08016	103.75943
May 2011	99.68624	96.19470	103.17777	94.34640	105.02608
Jun 2011	108.68249	105.19090	112.17408	103.34256	114.02241
Jul 2011	116.83307	113.34148	120.32467	111.49314	122.17300
Aug 2011	119.20688	115.71529	122.69848	113.86695	124.54682
Sep 2011	107.71521	104.22361	111.20680	102.37527	113.05514
Oct 2011	100.48851	96.99691	103.98010	95.14857	105.82844
Nov 2011	102.04890	98.55730	105.54050	96.70896	107.38884
Dec 2011	115.44234	111.95075	118.93394	110.10241	120.78228
Jan 2012	121.58165	118.00120	125.16210	116.10582	127.05748
Feb 2012	116.93193	113.32641	120.53745	111.41776	122.44609
Mar 2012	109.78938	106.18044	113.39831	104.26998	115.30877
Apr 2012	100.21603	96.60596	103.82610	94.69490	105.73715
May 2012	101.48247	97.87227	105.09267	95.96114	107.00379
Jun 2012	110.47872	106.86847	114.08897	104.95731	116.00012
Jul 2012	118.62930	115.01905	122.23956	113.10789	124.15071
Aug 2012	121.00311	117.39286	124.61337	115.48170	126.52453
Sep 2012	109.51144	105.90118	113.12170	103.99002	115.03285
Oct 2012	102.28474	98.67448	105.89500	96.76332	107.80615
Nov 2012	103.84513	100.23487	107.45539	98.32372	109.36655
Dec 2012	117.23857	113.62832	120.84883	111.71716	122.75999
Jan 2013	123.37788	119.68162	127.07415	117.72494	129.03083
Feb 2013	118.72816	115.00761	122.44871	113.03807	124.41825
Mar 2013	111.58561	107.86175	115.30947	105.89045	117.28076
Apr 2013	102.01226	98.28730	105.73722	96.31542	107.70909
May 2013	103.27870	99.55361	107.00378	97.58167	108.97573
Jun 2013	112.27495	108.54981	116.00009	106.57784	117.97205
Jul 2013	120.42553	116.70039	124.15067	114.72842	126.12265
Aug 2013	122.79935	119.07420	126.52449	117.10223	128.49646
Sep 2013	111.30767	107.58252	115.03281	105.61055	117.00478
Oct 2013	104.08097	100.35582	107.80611	98.38385	109.77809
Nov 2013	105.64136	101.91622	109.36651	99.94425	111.33848
Dec 2013	119.03481	115.30966	122.75995	113.33769	124.73192

Figure 6.2.6: Forecast results by ARIMA(2,0,1)(0,1,1)[12] with drift from 2011-2013

Jan 2014	125.17411	121.36556	128.98267	119.34943	130.99880
Feb 2014	120.52439	116.69226	124.35652	114.66365	126.38512
Mar 2014	113.38184	109.54649	117.21718	107.51618	119.24749
Apr 2014	103.80849	99.97207	107.64490	97.94120	109.67577
May 2014	105.07493	101.23840	108.91146	99.20746	110.94240
Jun 2014	114.07118	110.23460	117.90776	108.20363	119.93873
Jul 2014	122.22176	118.38518	126.05835	116.35421	128.08932
Aug 2014	124.59558	120.75899	128.43217	118.72802	130.46313
Sep 2014	113.10390	109.26731	116.94049	107.23634	118.97145
Oct 2014	105.87720	102.04061	109.71379	100.00964	111.74476
Nov 2014	107.43759	103.60100	111.27418	101.57003	113.30515
Dec 2014	120.83104	116.99445	124.66763	114.96348	126.69859
Jan 2015	126.97034	123.05272	130.88797	120.97885	132.96184
Feb 2015	122.32062	118.38007	126.26117	116.29407	128.34717
Mar 2015	115.17807	111.23439	119.12175	109.14673	121.20940
Apr 2015	105.60472	101.66000	109.54943	99.57180	111.63764
May 2015	106.87116	102.92633	110.81599	100.83806	112.90426
Jun 2015	115.86741	111.92253	119.81229	109.83423	121.90059
Jul 2015	124.01799	120.07311	127.96288	117.98481	130.05118
Aug 2015	126.39181	122.44692	130.33670	120.35862	132.42499
Sep 2015	114.90013	110.95524	118.84502	108.86694	120.93331
Oct 2015	107.67343	103.72854	111.61832	101.64024	113.70662
Nov 2015	109.23382	105.28893	113.17871	103.20064	115.26701
Dec 2015	122.62727	118.68238	126.57216	116.59408	128.66045
Jan 2016	128.76657	124.74283	132.79032	122.61278	134.92036
Feb 2016	124.11685	120.07078	128.16292	117.92892	130.30478
Mar 2016	116.97430	112.92519	121.02341	110.78171	123.16688
Apr 2016	107.40095	103.35082	111.45107	101.20682	113.59508
May 2016	108.66739	104.61715	112.71763	102.47308	114.86170
Jun 2016	117.66364	113.61335	121.71393	111.46926	123.85802
Jul 2016	125.81422	121.76393	129.86452	119.61984	132.00861
Aug 2016	128.18804	124.13774	132.23833	121.99365	134.38243
Sep 2016	116.69636	112.64606	120.74665	110.50197	122.89075
Oct 2016	109.46966	105.41937	113.51995	103.27527	115.66405
Nov 2016	111.03005	106.97976	115.08035	104.83567	117.22444
Dec 2016	124.42350	120.37320	128.47379	118.22911	130.61789

Figure 6.2.7: Forecast results by ARIMA(2,0,1)(0,1,1)[12] with drift from 2014-2016

Jan 2017	130.56280	126.43567	134.68994	124.25089	136.87472
Feb 2017	125.91308	121.76418	130.06198	119.56788	132.25828
Mar 2017	118.77053	114.61866	122.92240	112.42079	125.12027
Apr 2017	109.19718	105.04432	113.35004	102.84593	115.54843
May 2017	110.46362	106.31065	114.61659	104.11220	116.81504
Jun 2017	119.45987	115.30685	123.61289	113.10838	125.81136
Jul 2017	127.61046	123.45743	131.76348	121.25896	133.96195
Aug 2017	129.98427	125.83124	134.13729	123.63277	136.33577
Sep 2017	118.49259	114.33957	122.64561	112.14109	124.84409
Oct 2017	111.26589	107.11287	115.41891	104.91439	117.61739
Nov 2017	112.82628	108.67326	116.97931	106.47478	119.17779
Dec 2017	126.21973	122.06670	130.37275	119.86823	132.57123

Figure 6.2.8: Forecast results by ARIMA(2,0,1)(0,1,1)[12] with drift for 2017

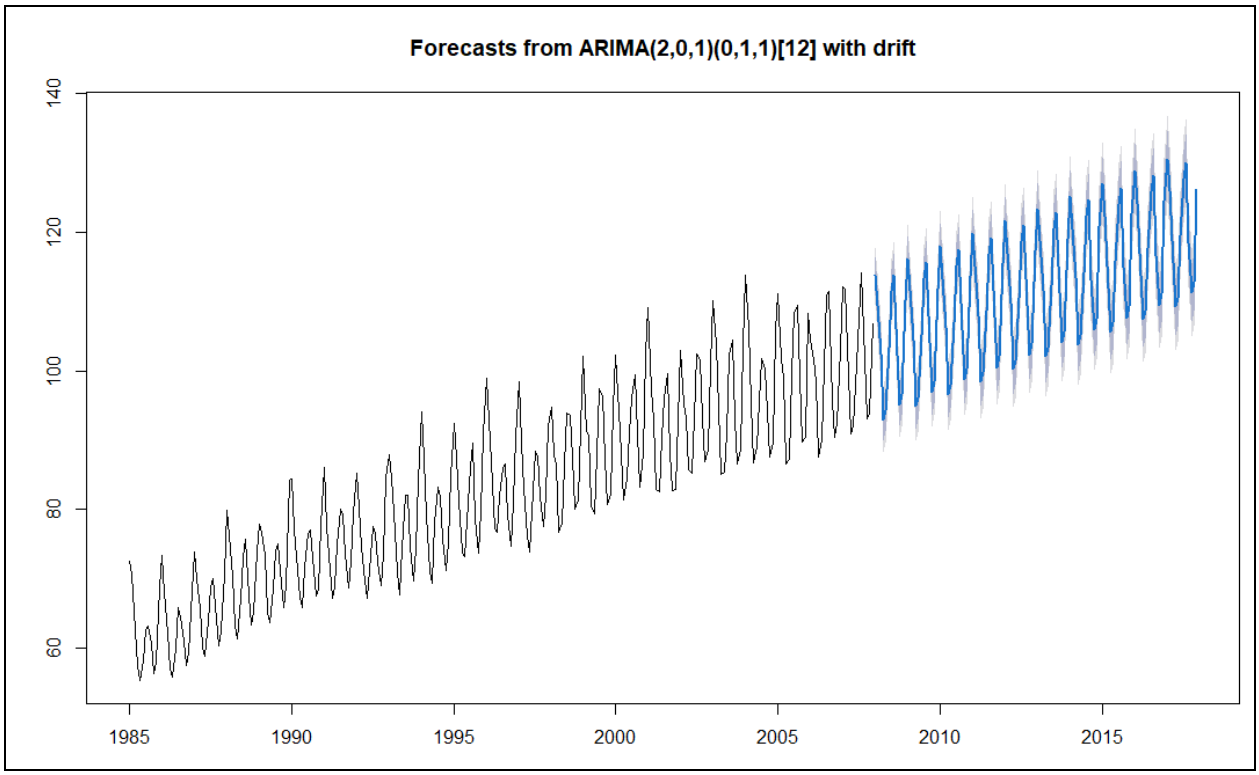

Figure 6.2.9: Forecast results by ARIMA(2,0,1)(0,1,1)[12] with drift

Since our test data is from 2008 - 2017,which is 10 years, then we will use the forecasting model to forecast for 10 years by using h =120 which means we are doing forecasting for 120 months (10 years).

```
> error=test-forecast_arimaTrain$mean
> #(test)
> #print(error)
> mse = mean(error*error)
> print(mse)
[1] 132.9156
> rmse = sqrt(mse)
> print(rmse)
[1] 11.5289
```

Figure 6.2.10: Root Mean Square Error of ARIMA

Figure 6.2.10 shows how we calculate the Root Mean Square Error (RMSE) for the ARIMA model, we will use the test data to minus the forecasted value by the model to get the value of error. Then we will calculate the Mean Square Error (MSE) by getting the mean of error multiplied by error (error^2). Lastly we will square root the MSE and the value of RMSE of the model is 11.5289. We will not use the accuracy(),a built-in function in R, to evaluate the

performance of the model as the accuracy as the function evaluates the model based on the train data. By right, to avoid overfitting, we will evaluate the model based on test data, hence we are comparing the predicted test data and actual value of test data to calculate the RMSE as the evaluation of the model.

```
> summary(arimaTrain)
Series: train
ARIMA(2,0,1)(0,1,1)[12] with drift

Coefficients:
          ar1     ar2     ma1     sma1   drift
      -0.0910  0.2468  0.6263  -0.7047  0.1497
s.e.   0.2503  0.1452  0.2371   0.0483  0.0068
```

Figure 6.2.11: Coefficients of ARIMA

The model's coefficient information is as below :

ar1, ϕ_1 = -0.091

ar2, ϕ_2 = 0.2468

ma1, θ_1 = -0.6263

sma1, Θ_1 = -0.5427

drift, c = 0.1497

```
> coeftest(arimaTrain)

z test of coefficients:

        Estimate Std. Error  z value  Pr(>|z|)
ar1   -0.0909999  0.2503210  -0.3635  0.716207
ar2    0.2468256  0.1452199   1.6997  0.089193 .
ma1    0.6263322  0.2371447   2.6411  0.008263 **
sma1  -0.7046962  0.0482527 -14.6043 < 2.2e-16 ***
drift  0.1496859  0.0068144  21.9661 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Figure 6.2.12: Results of coefficients test

After identifying the coefficient of the model we would like to test the significance of the coefficient of the model.

1. H_o : Coefficient is not significant
2. H_1 : Coefficient is significant

From the figure above, we can see that the p-value of ϕ_1, ϕ_2 , θ_1,Θ_1 and c are 0.716207, 0.089193, 0.008263, 2.2e-16 (0.00000000000000022) and 2.2e-16 respectively. Based on the figure above, it shows that the 3 p-values are lower than $\alpha = 0.05$ which are θ_1,Θ_1 and c, therefore we successfully reject H0 and conclude that these 3 coefficients are significant. For ϕ_1 and ϕ_2 , which having p-value larger than 0.05, we do not reject the null hypothesis and conclude that these 2 coefficients are not significant. This means that only θ_1,Θ_1 and c are important in the model to do forecasting.

6.3 Exponential Smoothing(ETS)

```
> fit <- ets(train)
> summary(fit)
ETS(M,Ad,M)

Call:
 ets(y = train)

  Smoothing parameters:
    alpha = 0.6715
    beta  = 1e-04
    gamma = 1e-04
    phi   = 0.9796

  Initial states:
    l = 62.3361
    b = 0.2821
    s = 1.0704 0.9362 0.9037 0.9701 1.0617 1.0542
           0.9747 0.8976 0.906 1.0039 1.0731 1.1483

  sigma:  0.0242

     AIC     AICc      BIC
1945.578 1948.239 2010.745
```

Figure 6.3.1 Results of ETS Model

The third model we will be using is Exponential Smoothing(ETS). After fitting the train data to the model, we print the summary of the model. We can see that our data is a (M,Ad,M) which means that the model is a Multiplicative Error, Addictive Trend and Multiplicative Seasonal model. The suggested smoothing parameters is stated as below:

alpha: 0.6715
beta: 1e-04 (0.0001)
gamma: 1e-04 (0.0001)
Phi: 0.9706

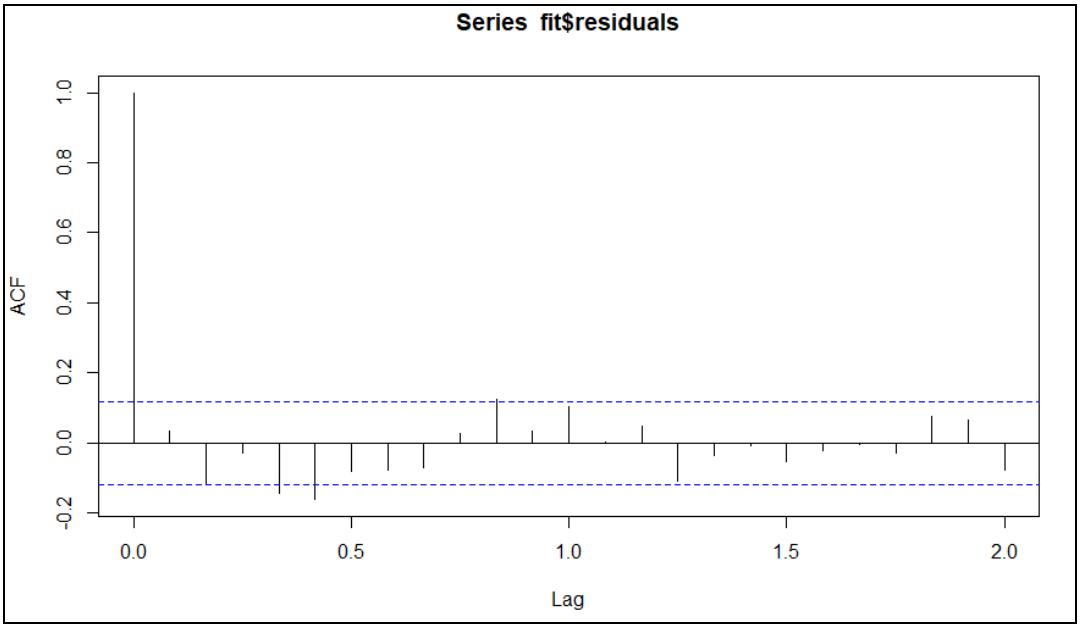

Figure 6.3.2: ACF of residuals of ETS model

```
> Box.test(fit$residuals, lag =12)

        Box-Pierce test

data:  fit$residuals
X-squared = 29.587, df = 12, p-value = 0.003221
```

Figure 6.3.3Box Test of ETS Model

From Figure 6.3.2, we can see that the ACF of ETS model shows a white noise pattern as there are only two spikes before lag 0.5 slightly exceed the blue dotted line. Hence, we can ignore them as they only slightly exceed the blue dotted line and conclude that the ACF of ETS model shows a white noise pattern. Besides, we will use the Box-Pierce test to determine whether the ETS model's residual is white noise or not. This is due the fact that if there are any signals in the model's residuals, we must update the model.

Hypothesis of Box-Pierce Test

1. H_o : The ETS model does not show a lack of fit.
2. H_1 : The ETS model does show a lack of fit.

Figure 6.3.3 shows that the p-value is equal to 0.003221, which is smaller than 0.05 and indicates that we reject the null hypothesis and conclude that the ETS model does show a lack of fit and its residual is not showing white noise pattern. Although the ACF shows a white noise pattern, we will still stick with the result of the box test which states that there are still some signals in the residuals of the model. After testing the residuals of the model, we will proceed to forecasting with the ETS model.

```
> forecast <- forecast(fit,h=120)
> print(forecast)
         Point Forecast     Lo 80     Hi 80     Lo 95     Hi 95
Jan 2008      115.15690 111.59062 118.72318 109.70274 120.61105
Feb 2008      107.61570 103.60082 111.63059 101.47546 113.75594
Mar 2008      100.68043  96.37951 104.98135  94.10274 107.25811
Apr 2008       90.85854  86.54097  95.17610  84.25539  97.46168
May 2008       90.01977  85.34951  94.69004  82.87722  97.16232
Jun 2008       97.74981  92.28520 103.21442  89.39241 106.10721
Jul 2008      105.73090  99.42313 112.03868  96.08399 115.37782
Aug 2008      106.47819  99.74957 113.20682  96.18765 116.76874
Sep 2008       97.29016  90.81642 103.76389  87.38943 107.19088
Oct 2008       90.63621  84.31620  96.95622  80.97059 100.30183
Nov 2008       93.89881  87.06484 100.73279  83.44715 104.35048
Dec 2008      107.35607  99.22819 115.48395  94.92555 119.78659
Jan 2009      115.17582 106.13085 124.22080 101.34272 129.00892
Feb 2009      107.63303  98.88712 116.37894  94.25731 121.00874
Mar 2009      100.69630  92.24845 109.14416  87.77643 113.61618
Apr 2009       90.87257  83.01639  98.72875  78.85758 102.88756
May 2009       90.03340  82.02595  98.04085  77.78706 102.27973
Jun 2009       97.76430  88.83274 106.69586  84.10466 111.42394
Jul 2009      105.74626  95.83613 115.65639  90.59002 120.90250
Aug 2009      106.49334  96.26814 116.71855  90.85524 122.13144
Sep 2009       97.30371  87.74210 106.86533  82.68049 111.92694
Oct 2009       90.64858  81.54160  99.75556  76.72066 104.57650
Nov 2009       93.91137  84.27442 103.54832  79.17293 108.64981
Dec 2009      107.37014  96.12559 118.61469  90.17308 124.56719
Jan 2010      115.19060 102.88867 127.49254  96.37642 134.00479
Feb 2010      107.64656  95.93175 119.36137  89.73030 125.56282
Mar 2010      100.70871  89.54811 111.86930  83.64004 117.77737
Apr 2010       90.88354  80.63368 101.13339  75.20773 106.55934
May 2010       90.04404  79.71535 100.37272  74.24768 105.84040
Jun 2010       97.77562  86.37470 109.17653  80.33942 115.21181
Jul 2010      105.75825  93.22916 118.28734  86.59666 124.91984
Aug 2010      106.50517  93.69183 119.31852  86.90885 126.10149
Sep 2010       97.31430  85.43043 109.19817  79.13949 115.48911
Oct 2010       90.65825  79.42534 101.89115  73.47900 107.83749
Nov 2010       93.92118  82.11861 105.72375  75.87070 111.97166
Dec 2010      107.38112  93.70057 121.06167  86.45852 128.30372
```

Figure 6.3.4: Forecast results by ETS(M,Ad,M) for 2008-2010

Jan 2011	115.20215	100.32761	130.07668	92.45351	137.95079
Feb 2011	107.65713	93.57455	121.73970	86.11969	129.19457
Mar 2011	100.71839	87.37510	114.06168	80.31158	121.12520
Apr 2011	90.89210	78.70049	103.08370	72.24665	109.53755
May 2011	90.05235	77.82641	102.27829	71.35439	108.75031
Jun 2011	97.78446	84.35101	111.21790	77.23977	118.32914
Jul 2011	105.76762	91.06874	120.46650	83.28762	128.24762
Aug 2011	106.51441	91.54370	121.48513	83.61868	129.41015
Sep 2011	97.32257	83.49188	111.15327	76.17035	118.47479
Oct 2011	90.66579	77.64108	103.69051	70.74621	110.58537
Nov 2011	93.92884	80.29179	107.56589	73.07277	114.78491
Dec 2011	107.38970	91.63583	123.14357	83.29624	131.48316
Jan 2012	115.21117	98.13723	132.28510	89.09883	141.32350
Feb 2012	107.66538	91.54994	123.78082	83.01894	132.31182
Mar 2012	100.72596	85.50116	115.95075	77.44164	124.01027
Apr 2012	90.89879	77.02700	104.77057	69.68373	112.11384
May 2012	90.05884	76.18528	103.93239	68.84107	111.27661
Jun 2012	97.79136	82.58676	112.99597	74.53792	121.04480
Jul 2012	105.77493	89.17909	122.37077	80.39379	131.15608
Aug 2012	106.52163	89.65893	123.38433	80.73235	132.31091
Sep 2012	97.32903	81.78591	112.87216	73.55788	121.10019
Oct 2012	90.67169	76.06641	105.27696	68.33485	113.00853
Nov 2012	93.93482	78.67517	109.19448	70.59719	117.27245
Dec 2012	107.39640	89.80389	124.98891	80.49098	134.30183
Jan 2013	115.21821	96.18893	134.24748	86.11544	144.32097
Feb 2013	107.67183	89.74478	125.59888	80.25477	135.08889
Mar 2013	100.73186	83.82649	117.63724	74.87732	126.58641
Apr 2013	90.90401	75.52815	106.27987	67.38867	114.41935
May 2013	90.06391	74.71229	105.41553	66.58563	113.54219
Jun 2013	97.79675	81.00000	114.59351	72.10834	123.48517
Jul 2013	105.78065	87.47621	124.08508	77.78643	133.77487
Aug 2013	106.52727	87.95721	125.09733	78.12681	134.92773
Sep 2013	97.33408	80.24281	114.42536	71.19523	123.47293
Oct 2013	90.67629	74.63955	106.71303	66.15022	115.20237
Nov 2013	93.93950	77.20779	110.67121	68.35055	119.52844
Dec 2013	107.40164	88.13832	126.66495	77.94093	136.86234

Figure 6.3.5: Forecast results by ETS(M,Ad,M) for 2011-2013

Jan 2014	115.22371	94.41475	136.03267	83.39915	147.04826
Feb 2014	107.67686	88.09840	127.25533	77.73418	137.61955
Mar 2014	100.73648	82.29684	119.17612	72.53549	128.93747
Apr 2014	90.90809	74.15712	107.65906	65.28969	116.52649
May 2014	90.06787	73.36302	106.77272	64.52001	115.61573
Jun 2014	97.80097	79.54455	116.05738	69.88019	125.72174
Jul 2014	105.78511	85.91219	125.65803	75.39210	136.17812
Aug 2014	106.53167	86.39225	126.67109	75.73109	137.33225
Sep 2014	97.33802	78.82196	115.85409	69.02014	125.65590
Oct 2014	90.67989	73.32416	108.03562	64.13659	117.22319
Nov 2014	93.94315	75.85345	112.03284	66.27735	121.60895
Dec 2014	107.40572	86.59932	128.21213	75.58507	139.22638
Jan 2015	115.22801	92.77358	137.68243	80.88692	149.56909
Feb 2015	107.68080	86.57381	128.78779	75.40044	139.96116
Mar 2015	100.74008	80.87887	120.60130	70.36497	131.11520
Apr 2015	90.91128	72.88488	108.93767	63.34229	118.48027
May 2015	90.07096	72.10974	108.03219	62.60164	117.54029
Jun 2015	97.80426	78.19134	117.41717	67.80889	127.79963
Jul 2015	105.78860	84.45666	127.12054	73.16421	138.41299
Aug 2015	106.53511	84.93451	128.13571	73.49985	139.57037
Sep 2015	97.34110	77.49727	117.18493	66.99257	127.68963
Oct 2015	90.68270	72.09672	109.26868	62.25789	119.10751
Nov 2015	93.94600	74.58859	113.30340	64.34140	123.55060
Dec 2015	107.40892	85.16079	129.65704	73.38335	141.43449
Jan 2016	115.23136	91.23831	139.22442	78.53716	151.92557
Feb 2016	107.68387	85.14646	130.22128	73.21588	142.15186
Mar 2016	100.74290	79.55031	121.93549	68.33163	133.15417
Apr 2016	90.91377	71.69198	110.13556	61.51657	120.31096
May 2016	90.07338	70.93373	109.21303	60.80181	119.34495
Jun 2016	97.80683	76.92064	118.69301	65.86416	129.74949
Jul 2016	105.79132	83.08891	128.49373	71.07098	140.51166
Aug 2016	106.53780	83.56375	129.51184	71.40203	141.67357
Sep 2016	97.34351	76.25077	118.43624	65.08495	129.60206
Oct 2016	90.68490	70.94096	110.42883	60.48915	120.88064
Nov 2016	93.94823	73.39684	114.49962	62.51759	125.37887
Dec 2016	107.41141	83.80455	131.01828	71.30783	143.51499

Figure 6.3.6: Forecast results by ETS(M,Ad,M) for 2014-2016

Jan 2017	115.23399	89.78995	140.67802	76.32069	154.14728
Feb 2017	107.68627	83.79910	131.57344	71.15400	144.21854
Mar 2017	100.74510	78.29545	123.19475	66.41132	135.07888
Apr 2017	90.91571	70.56458	111.26684	59.79134	122.04008
May 2017	90.07527	69.82167	110.32886	59.10006	121.05047
Jun 2017	97.80884	75.71836	119.89931	64.02438	131.59329
Jul 2017	105.79345	81.79410	129.79280	69.08961	142.49729
Aug 2017	106.53990	82.26538	130.81441	69.41523	143.66457
Sep 2017	97.34538	75.06947	119.62130	63.27732	131.41345
Oct 2017	90.68661	69.84509	111.52813	58.81225	122.56097
Nov 2017	93.94997	72.26625	115.63368	60.78759	127.11235
Dec 2017	107.41336	82.51728	132.30945	69.33809	145.48864

Figure 6.3.7: Forecast results by ETS(M,Ad,M) for 2007

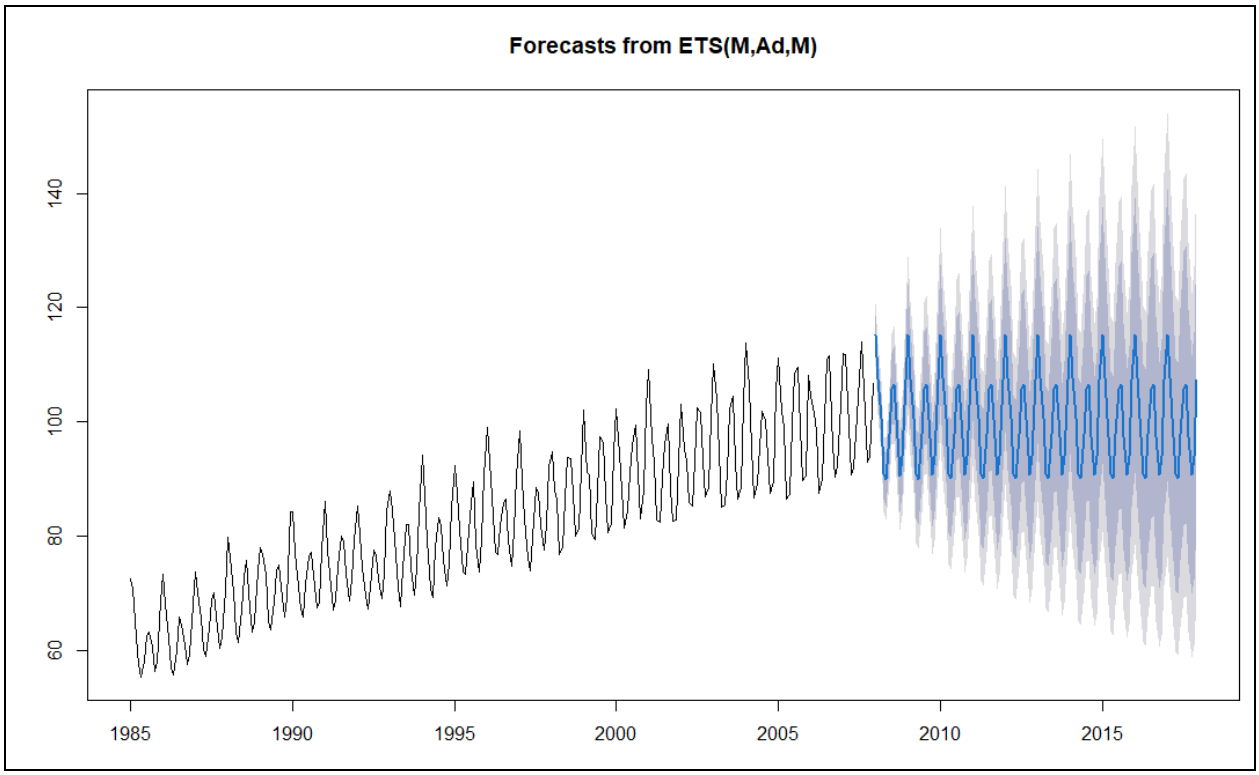

Figure 6.3.8: Forecast results by ETS(M,Ad,M)

Since our test data is from 2008 - 2017,which is 10 years, then we will use the forecasting model to forecast for 10 years by using h =120 which means we are doing forecasting for 120 months (10 years).

```
> error_ets=test-forecast$mean
> mse_ets = mean(error_ets*error_ets)
> print(mse_ets)
[1] 14.95901
> rmse_ets = sqrt(mse_ets)
> print(rmse_ets)
[1] 3.867688
>
```

Figure 6.3.10: Root Mean Square Error of ETS(M,Ad,M)

Figure 6.3.10 shows how we calculate the Root Mean Square Error (RMSE) for the ETS(M,Ad,M) model, we will use the test data to minus the forecasted value by the model to get the value of error. Then we will calculate the Mean Square Error (MSE) by getting the mean of error multiplied by error (error^2). Lastly we will square root the MSE and the value of RMSE of the model is 3.867688. We will not use the accuracy(),a built-in function in R, to evaluate the performance of the model as the accuracy as the function evaluates the model based on the train data. By right, to avoid overfitting, we will evaluate the model based on test data, hence we are

comparing the predicted test data and actual value of test data to calculate the RMSE as the evaluation of the model.

6.4 Holt Winters

```
> hwTrain <-HoltWinters(train,seasonal="mult")
> summary(hwTrain)
             Length Class  Mode
fitted       1056   mts    numeric
x             276   ts     numeric
alpha           1   -none- numeric
beta            1   -none- numeric
gamma           1   -none- numeric
coefficients   14   -none- numeric
seasonal        1   -none- character
SSE             1   -none- numeric
call            3   -none- call
```

Figure 6.4.1: Holt Winters Multiplicative Model

The fourth model we will be using is Holt Winter Multiplicative Model. This is because our model is seasonal multiplicative so we will choose Holt Winter Multiplicative Model over Holt Winter Additive Model. Hence, when fitting the data, we will set the seasonal to mult which means multiplicative. From the summary we can see that if we do not set the other parameters, it will use the default parameters where alpha,beta and gamma will be 1.

```
> Box.test(forecast_hw$residuals,lag=12)

        Box-Pierce test

data:  forecast_hw$residuals
X-squared = 32.305, df = 12, p-value = 0.001241
```

Figure 6.4.2: Box Test of Holt Winters Multiplicative Model

Furthermore, we will use the Box-Pierce test to determine whether the Holt Winters Multiplicative model residual shows a white noise pattern or not. This is due the fact that if there are any signals in the model's residuals, we must update the model otherwise the forecasting process would not be successful.

Hypothesis of Box-Pierce Test

1. H_o : The Holt Winters Multiplicative model does not show a lack of fit.
2. H_1 : The Holt Winters Multiplicative model does show a lack of fit.

Figure 6.4.2 shows that the p-value is equal to 0.001241, which is smaller than 0.05 and indicates that we reject the null hypothesis and conclude that the Holt Winters Multiplicative model does show a lack of fit and its residuals does not show a white noise pattern.

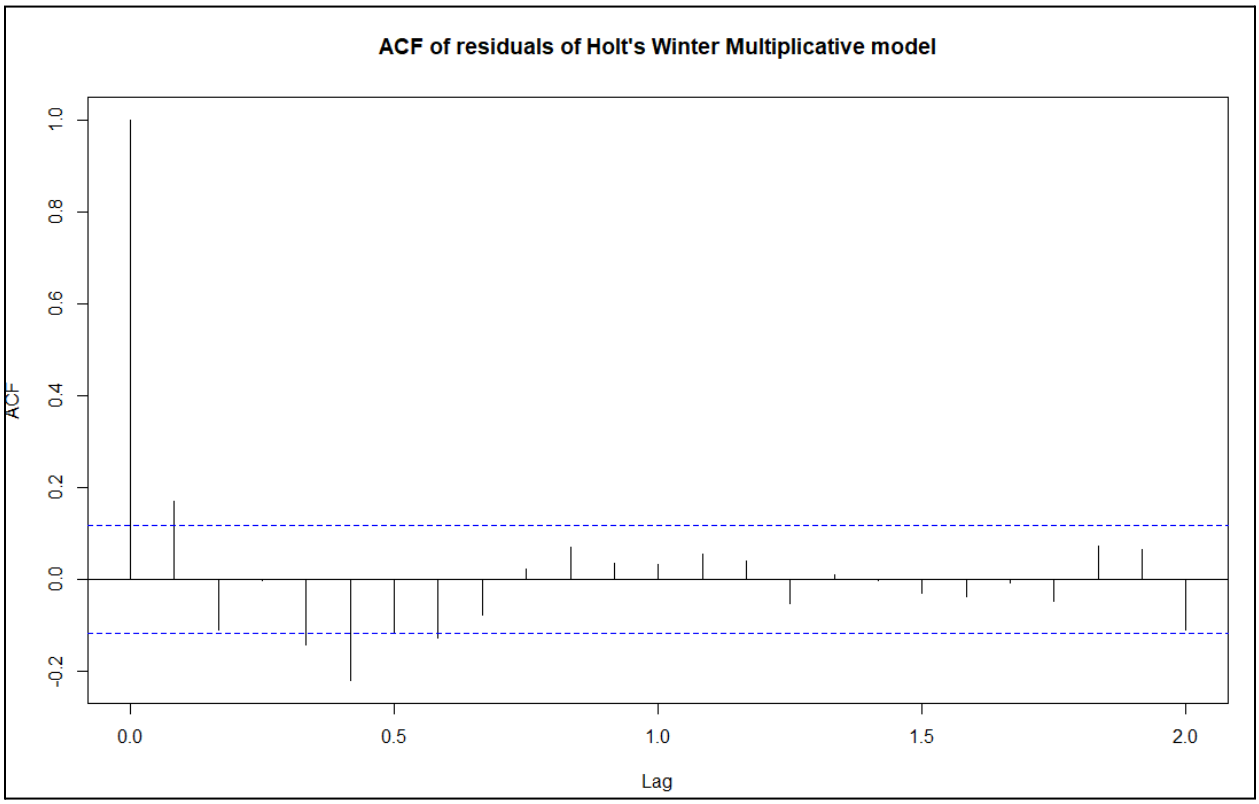

Figure 6.4.3: ACF of Holt Winters Multiplicative Model residuals

From Figure 6.4.3, we notice that there are spikes exceeding the blue dotted line before Lag 0.5 which means that the residuals of Holt Winters Multiplicative Model does not show a white noise pattern. Hence, we can get the same conclusion from the ACF of residuals and the Box-Test, both showing that the residuals do not show a white noise pattern. After determining whether the residuals is white noise or not, we will proceed to forecasting with Holt Winters Multiplicative Model.

```
> forecast_hw <- forecast(hwTrain,h=120)
> print(forecast_hw)
         Point Forecast     Lo 80     Hi 80     Lo 95     Hi 95
Jan 2008      115.09388 113.21113 116.97663 112.21447 117.97330
Feb 2008      110.16236 107.96831 112.35641 106.80685 113.51787
Mar 2008      102.37675  99.96040 104.79310  98.68126 106.07223
Apr 2008       92.47438  89.91522  95.03354  88.56048  96.38828
May 2008       93.86618  91.02505  96.70732  89.52104  98.21132
Jun 2008      103.08519  99.81665 106.35373  98.08638 108.08399
Jul 2008      111.18597 107.51547 114.85646 105.57243 116.79950
Aug 2008      112.87138 108.97040 116.77236 106.90535 118.83741
Sep 2008      100.17751  96.45679 103.89822  94.48716 105.86785
Oct 2008       92.59887  88.91917  96.27858  86.97125  98.22649
Nov 2008       94.43620  90.51242  98.35998  88.43530 100.43710
Dec 2008      108.91648  87.86460 129.96837  76.72041 141.11256
Jan 2009      116.41002 101.71693 131.10310  93.93888 138.88115
Feb 2009      111.42090  97.28584 125.55597  89.80318 133.03862
Mar 2009      103.54523  90.32526 116.76519  83.32703 123.76342
Apr 2009       93.52884  81.48845 105.56923  75.11465 111.94303
May 2009       94.93550  82.65468 107.21631  76.15361 113.71738
Jun 2009      104.25841  90.74446 117.77236  83.59061 124.92621
Jul 2009      112.45018  97.84150 127.05886  90.10814 134.79223
Aug 2009      114.15354  99.27143 129.03566  91.39331 136.91378
Sep 2009      101.31440  88.01012 114.61868  80.96726 121.66153
Oct 2009       93.64877  81.26113 106.03641  74.70350 112.59403
Nov 2009       95.50591  82.81643 108.19539  76.09902 114.91280
Dec 2009      110.14905  80.18804 140.11007  64.32764 155.97047
Jan 2010      117.72615  96.82997 138.62233  85.76820 149.68410
Feb 2010      112.67944  92.61699 132.74190  81.99656 143.36232
Mar 2010      104.71371  85.99573 123.43169  76.08703 133.34039
Apr 2010       94.58330  77.58810 111.57849  68.59139 120.57520
May 2010       96.00481  78.70824 113.30137  69.55199 122.45762
Jun 2010      105.43163  86.42313 124.44013  76.36064 134.50262
Jul 2010      113.71440  93.19117 134.23763  82.32683 145.10197
Aug 2010      115.43571  94.55936 136.31206  83.50808 147.36333
Sep 2010      102.45129  83.83577 121.06681  73.98130 130.92128
Oct 2010       94.69866  77.41235 111.98496  68.26154 121.13578
Nov 2010       96.57562  78.90320 114.24805  69.54798 123.60326
Dec 2010      111.38163  74.45486 148.30839  54.90702 167.85624
```

Figure 6.4.4: Forecast results by Holt Winter Multiplicative Model for 2008-2010

Jan 2011	119.04228	93.23532	144.84925	79.57394	158.51063
Feb 2011	113.93798	89.17773	138.69824	76.07044	151.80552
Mar 2011	105.88219	82.80073	128.96365	70.58214	141.18224
Apr 2011	95.63775	74.70331	116.57220	63.62128	127.65423
May 2011	97.07412	75.78472	118.36352	64.51479	129.63345
Jun 2011	106.60485	83.21890	129.99080	70.83912	142.37058
Jul 2011	114.97862	89.73991	140.21732	76.37935	153.57788
Aug 2011	116.71787	91.05842	142.37733	77.47513	155.96062
Sep 2011	103.58818	80.72844	126.44793	68.62722	138.54915
Oct 2011	95.74855	74.54165	116.95545	63.31539	128.18170
Nov 2011	97.64534	75.98017	119.31050	64.51132	130.77935
Dec 2011	112.61420	69.70621	155.52219	46.99209	178.23631
Jan 2012	120.35842	90.29019	150.42664	74.37303	166.34380
Feb 2012	115.19652	86.35810	144.03494	71.09197	159.30107
Mar 2012	107.05067	80.17931	133.92203	65.95447	148.14687
Apr 2012	96.69221	72.33405	121.05038	59.43961	133.94481
May 2012	98.14343	73.38199	122.90487	60.27408	136.01279
Jun 2012	107.77807	80.58439	134.97175	66.18892	149.36722
Jul 2012	116.24283	86.90118	145.58449	71.36864	161.11702
Aug 2012	118.00004	88.17741	147.82267	72.39026	163.60982
Sep 2012	104.72508	78.16915	131.28100	64.11129	145.33886
Oct 2012	96.79844	72.17514	121.42174	59.14035	134.45653
Nov 2012	98.71505	73.56896	123.86114	60.25743	137.17267
Dec 2012	113.84677	65.57321	162.12033	40.01873	187.67481
Jan 2013	121.67455	87.74652	155.60258	69.78611	173.56299
Feb 2013	116.45506	83.92193	148.98819	66.69993	166.21019
Mar 2013	108.21915	77.91336	138.52495	61.87045	154.56786
Apr 2013	97.74667	70.28489	125.20845	55.74751	139.74583
May 2013	99.21274	71.30306	127.12243	56.52856	141.89693
Jun 2013	108.95129	78.30436	139.59822	62.08086	155.82172
Jul 2013	117.50705	84.44383	150.57026	66.94123	168.07287
Aug 2013	119.28220	85.68275	152.88166	67.89627	170.66814
Sep 2013	105.86197	75.95198	135.77196	60.11858	151.60535
Oct 2013	97.84833	70.12390	125.57276	55.44747	140.24919
Nov 2013	99.78476	71.47819	128.09133	56.49359	143.07593
Dec 2013	115.07934	61.86795	168.29074	33.69954	196.45915

Figure 6.4.5: Forecast results by Holt Winter Multiplicative Model for 2011-2013

Jan 2014	122.99068	85.47799	160.50337	65.61997	180.36139
Feb 2014	117.71360	81.74874	153.67846	62.71009	172.71711
Mar 2014	109.38764	75.89143	142.88384	58.15961	160.61566
Apr 2014	98.80113	68.45573	129.14653	52.39184	145.21041
May 2014	100.28206	69.44681	131.11730	53.12362	147.44049
Jun 2014	110.12451	76.26824	143.98078	58.34582	161.90320
Jul 2014	118.77126	82.24903	155.29350	62.91532	174.62721
Aug 2014	120.56437	83.45418	157.67456	63.80924	177.31950
Sep 2014	106.99886	73.97066	140.02706	56.48659	157.51113
Oct 2014	98.89822	68.29025	129.50620	52.08736	145.70908
Nov 2014	100.85447	69.60872	132.10022	53.06822	148.64073
Dec 2014	116.31192	58.47951	174.14432	27.86489	204.75895
Jan 2015	124.30681	83.41018	165.20345	61.76080	186.85283
Feb 2015	118.97214	79.76750	158.17679	59.01381	178.93047
Mar 2015	110.55612	74.04770	147.06453	54.72132	166.39092
Apr 2015	99.85558	66.78735	132.92382	49.28208	150.42909
May 2015	101.35137	67.75342	134.94931	49.96774	152.73499
Jun 2015	111.29773	74.41054	148.18492	54.88364	167.71181
Jul 2015	120.03548	80.24632	159.82464	59.18321	180.88775
Aug 2015	121.84653	81.42040	162.27267	60.02010	183.67297
Sep 2015	108.13575	72.16212	144.10939	53.11882	163.15269
Oct 2015	99.94811	66.61609	133.28014	48.97118	150.92505
Nov 2015	101.92419	67.90156	135.94681	49.89108	153.95729
Dec 2015	117.54449	55.33610	179.75288	22.40496	212.68401
Jan 2016	125.62295	81.49501	169.75088	58.13509	193.11080
Feb 2016	120.23068	77.93229	162.52907	55.54088	184.92049
Mar 2016	111.72460	72.33964	151.10956	51.49050	171.95870
Apr 2016	100.91004	65.24144	136.57864	46.35963	155.46046
May 2016	102.42068	66.18412	138.65724	47.00165	157.83971
Jun 2016	112.47095	72.68882	152.25308	51.62943	173.31247
Jul 2016	121.29970	78.39005	164.20935	55.67505	186.92434
Aug 2016	123.12870	79.53515	166.72225	56.45811	189.79929
Sep 2016	109.27265	70.48538	148.05991	49.95264	168.59265
Oct 2016	100.99801	65.06368	136.93233	46.04120	155.95481
Nov 2016	102.99390	66.31835	139.66945	46.90349	159.08431
Dec 2016	118.77706	52.38815	185.16597	17.24398	220.31014

Figure 6.4.6: Forecast results by Holt Winter Multiplicative Model for 2014-2016

Jan 2017	126.93908	79.69943	174.17873	54.69227	199.18589
Feb 2017	121.48922	76.21153	166.76692	52.24296	190.73549
Mar 2017	112.89308	70.73791	155.04826	48.42231	177.36386
Apr 2017	101.96450	63.79159	140.13741	43.58407	160.34493
May 2017	103.48999	64.71218	142.26781	44.18444	162.79554
Jun 2017	113.64417	71.07378	156.21455	48.53838	178.74995
Jul 2017	122.56391	76.64869	168.47914	52.34264	192.78519
Aug 2017	124.41087	77.76647	171.05526	53.07442	195.74731
Sep 2017	110.40954	68.91215	151.90692	46.94477	173.87431
Oct 2017	102.04790	63.60693	140.48886	43.25752	160.83828
Nov 2017	104.06361	64.83254	143.29468	44.06487	164.06235
Dec 2017	120.00963	49.59982	190.41945	12.32711	227.69215

Figure 6.4.7: Forecast results by Holt Winter Multiplicative Model for 2007

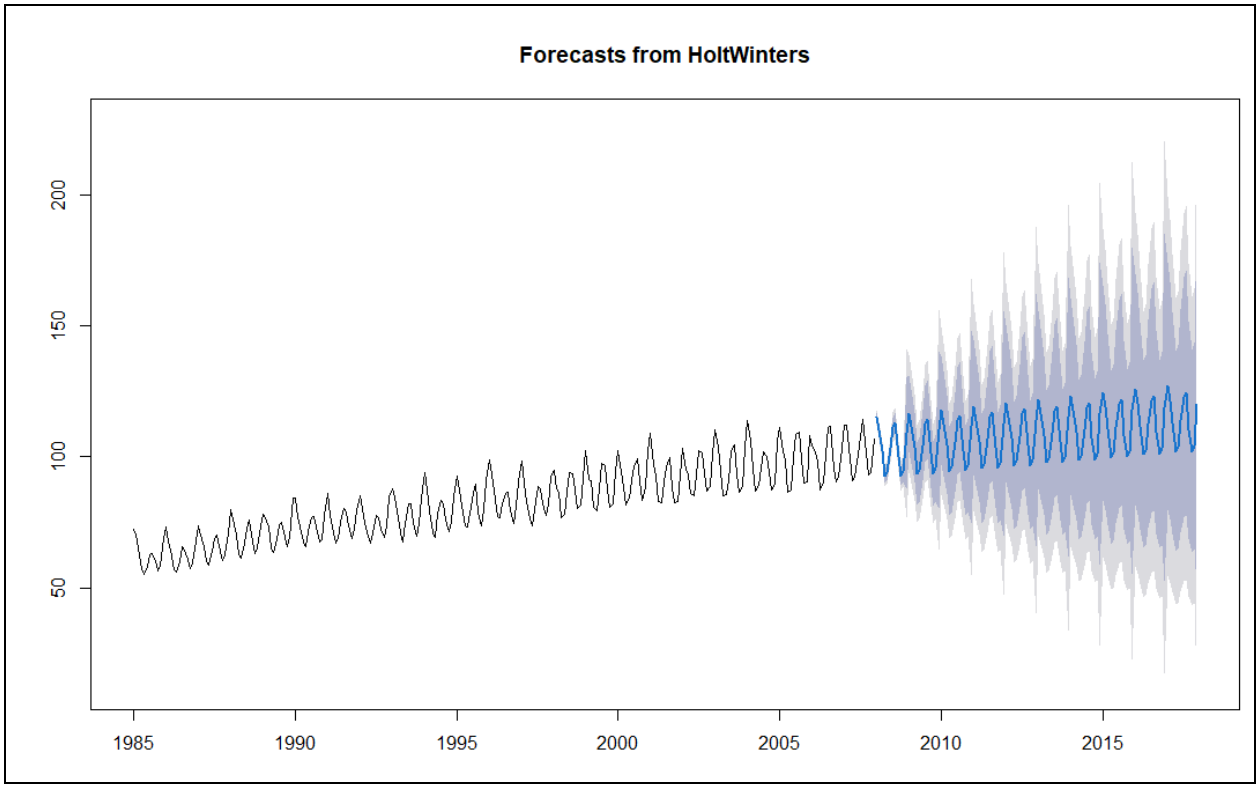

Figure 6.4.8: Forecast results by Holt Winter Multiplicative Model

Since our test data is from 2008 - 2017,which is 10 years, then we will use the forecasting model to forecast for 10 years by using h =120 which means we are doing forecasting for 120 months (10 years).

```
> error_hw=test-forecast_hw$mean
> #print(error_hw)
> mse_hw = mean(error_hw*error_hw)
> print(mse_hw)
[1] 59.28459
> rmse_hw = sqrt(mse_hw)
> print(rmse_hw)
[1] 7.699649
>
```

Figure 6.4.9: Root Mean Square Error of Holt Winter Multiplicative Model

Figure 6.3.9 shows how we calculate the Root Mean Square Error (RMSE) for the Holt Winter Multiplicative Model model, we will use the test data to minus the forecasted value by the model to get the value of error. Then we will calculate the Mean Square Error (MSE) by getting the mean of error multiplied by error (error^2). Lastly we will square root the MSE and the value of RMSE of the model is 7.699649. We will not use the accuracy(),a built-in function in R, to

evaluate the performance of the model as the accuracy as the function evaluates the model based on the train data. By right, to avoid overfitting, we will evaluate the model based on test data, hence we are comparing the predicted test data and actual value of test data to calculate the RMSE as the evaluation of the model.

6.5 Holt's Method Exponential Smoothing

```
> holtTrain <-holt(train,h=120)
> summary(holtTrain)

Forecast method: Holt's method

Model Information:
Holt's method

Call:
 holt(y = train, h = 120)

  Smoothing parameters:
    alpha = 0.0446
    beta  = 1e-04

  Initial states:
    l = 68.4506
    b = 0.1369

  sigma:  7.2146

     AIC     AICc      BIC
2648.013 2648.235 2666.115

Error measures:
                     ME     RMSE      MAE        MPE     MAPE     MASE       ACF1
Training set -0.1785191 7.162134 6.053852 -1.041053 7.384864 2.213981 0.5256191
```

Figure 6.5.1 Result of Holt's Method Exponential Smoothing

The fifth method we are using is Holt's Method Exponential Smoothing. We are using the default parameters to fit the training data into the medel and Figure 6.5.1 shows the result of Holt's Method Exponential Smoothing model. The default alpha will be 0.0446 and beta, 1e-04.

Next, we do the forecasting using the training time series data. We do not manually set the α and β for our model and we forecast using h = 120 to view the forecasting for the next ten years from the year of 2008 to 2017. The result is shown as below :

```
> forecast_holt <- forecast(holtTrain,h=120)
> print(forecast_holt)
         Point Forecast     Lo 80    Hi 80     Lo 95    Hi 95
Jan 2008       102.2612  93.01535 111.5071  88.12087 116.4016
Feb 2008       102.3932  93.13811 111.6484  88.23874 116.5477
Mar 2008       102.5252  93.26084 111.7897  88.35656 116.6939
Apr 2008       102.6573  93.38354 111.9310  88.47433 116.8402
May 2008       102.7893  93.50621 112.0723  88.59206 116.9865
Jun 2008       102.9213  93.62884 112.2137  88.70973 117.1328
Jul 2008       103.0533  93.75145 112.3551  88.82735 117.2792
Aug 2008       103.1853  93.87402 112.4966  88.94493 117.4256
Sep 2008       103.3173  93.99656 112.6380  89.06246 117.5721
Oct 2008       103.4493  94.11907 112.7795  89.17994 117.7187
Nov 2008       103.5813  94.24154 112.9211  89.29736 117.8653
Dec 2008       103.7133  94.36398 113.0627  89.41474 118.0119
Jan 2009       103.8453  94.48640 113.2043  89.53208 118.1586
Feb 2009       103.9773  94.60877 113.3459  89.64936 118.3053
Mar 2009       104.1093  94.73112 113.4876  89.76659 118.4521
Apr 2009       104.2414  94.85344 113.6293  89.88377 118.5989
May 2009       104.3734  94.97572 113.7710  90.00091 118.7458
Jun 2009       104.5054  95.09797 113.9128  90.11800 118.8927
Jul 2009       104.6374  95.22019 114.0546  90.23503 119.0397
Aug 2009       104.7694  95.34238 114.1964  90.35202 119.1867
Sep 2009       104.9014  95.46453 114.3383  90.46896 119.3338
Oct 2009       105.0334  95.58666 114.4801  90.58585 119.4809
Nov 2009       105.1654  95.70875 114.6221  90.70269 119.6281
Dec 2009       105.2974  95.83081 114.7640  90.81949 119.7753
Jan 2010       105.4294  95.95283 114.9060  90.93623 119.9226
Feb 2010       105.5614  96.07483 115.0480  91.05292 120.0699
Mar 2010       105.6934  96.19679 115.1901  91.16957 120.2173
Apr 2010       105.8254  96.31872 115.3322  91.28617 120.3647
May 2010       105.9575  96.44062 115.4743  91.40272 120.5122
Jun 2010       106.0895  96.56249 115.6164  91.51922 120.6597
Jul 2010       106.2215  96.68433 115.7586  91.63567 120.8073
Aug 2010       106.3535  96.80613 115.9008  91.75207 120.9549
Sep 2010       106.4855  96.92790 116.0431  91.86842 121.1026
Oct 2010       106.6175  97.04964 116.1853  91.98473 121.2503
Nov 2010       106.7495  97.17135 116.3277  92.10099 121.3980
Dec 2010       106.8815  97.29303 116.4700  92.21719 121.5458
Jan 2011       107.0135  97.41468 116.6124  92.33335 121.6937
Feb 2011       107.1455  97.53629 116.7548  92.44946 121.8416
Mar 2011       107.2775  97.65787 116.8972  92.56553 121.9895
Apr 2011       107.4095  97.77942 117.0397  92.68154 122.1375
May 2011       107.5416  97.90094 117.1822  92.79751 122.2856
Jun 2011       107.6736  98.02242 117.3247  92.91342 122.4337
Jul 2011       107.8056  98.14388 117.4673  93.02929 122.5818
Aug 2011       107.9376  98.26530 117.6098  93.14511 122.7300
Sep 2011       108.0696  98.38669 117.7525  93.26088 122.8783
Oct 2011       108.2016  98.50805 117.8951  93.37660 123.0266
Nov 2011       108.3336  98.62938 118.0378  93.49228 123.1749
Dec 2011       108.4656  98.75068 118.1805  93.60790 123.3233
```

Figure 6.5.2: Forecast results by Holt's Method Exponential Smoothing Model for 2008-2011

Jan 2012	108.5976	98.87194	118.3233	93.72348	123.4717
Feb 2012	108.7296	98.99317	118.4661	93.83901	123.6202
Mar 2012	108.8616	99.11438	118.6089	93.95449	123.7688
Apr 2012	108.9936	99.23555	118.7517	94.06992	123.9174
May 2012	109.1256	99.35668	118.8946	94.18531	124.0660
Jun 2012	109.2577	99.47779	119.0375	94.30065	124.2147
Jul 2012	109.3897	99.59887	119.1805	94.41593	124.3634
Aug 2012	109.5217	99.71991	119.3234	94.53117	124.5122
Sep 2012	109.6537	99.84092	119.4664	94.64637	124.6610
Oct 2012	109.7857	99.96191	119.6095	94.76151	124.8099
Nov 2012	109.9177	100.08285	119.7525	94.87660	124.9588
Dec 2012	110.0497	100.20377	119.8956	94.99165	125.1078
Jan 2013	110.1817	100.32466	120.0388	95.10665	125.2568
Feb 2013	110.3137	100.44552	120.1819	95.22160	125.4058
Mar 2013	110.4457	100.56634	120.3251	95.33651	125.5549
Apr 2013	110.5777	100.68713	120.4683	95.45136	125.7041
May 2013	110.7097	100.80789	120.6116	95.56617	125.8533
Jun 2013	110.8418	100.92862	120.7549	95.68093	126.0026
Jul 2013	110.9738	101.04932	120.8982	95.79564	126.1519
Aug 2013	111.1058	101.16999	121.0415	95.91031	126.3012
Sep 2013	111.2378	101.29063	121.1849	96.02493	126.4506
Oct 2013	111.3698	101.41123	121.3283	96.13949	126.6001
Nov 2013	111.5018	101.53181	121.4718	96.25402	126.7496
Dec 2013	111.6338	101.65235	121.6152	96.36849	126.8991
Jan 2014	111.7658	101.77286	121.7588	96.48292	127.0487
Feb 2014	111.8978	101.89334	121.9023	96.59729	127.1983
Mar 2014	112.0298	102.01379	122.0459	96.71163	127.3480
Apr 2014	112.1618	102.13421	122.1894	96.82591	127.4978
May 2014	112.2938	102.25460	122.3331	96.94014	127.6475
Jun 2014	112.4258	102.37495	122.4767	97.05433	127.7974
Jul 2014	112.5579	102.49528	122.6204	97.16847	127.9472
Aug 2014	112.6899	102.61557	122.7641	97.28257	128.0972
Sep 2014	112.8219	102.73584	122.9079	97.39661	128.2471
Oct 2014	112.9539	102.85607	123.0517	97.51061	128.3971
Nov 2014	113.0859	102.97627	123.1955	97.62457	128.5472
Dec 2014	113.2179	103.09644	123.3393	97.73847	128.6973
Jan 2015	113.3499	103.21658	123.4832	97.85233	128.8475
Feb 2015	113.4819	103.33669	123.6271	97.96614	128.9977
Mar 2015	113.6139	103.45677	123.7711	98.07990	129.1479
Apr 2015	113.7459	103.57682	123.9150	98.19362	129.2982
May 2015	113.8779	103.69683	124.0590	98.30728	129.4486
Jun 2015	114.0099	103.81682	124.2031	98.42091	129.5990
Jul 2015	114.1419	103.93677	124.3471	98.53448	129.7494
Aug 2015	114.2740	104.05670	124.4912	98.64801	129.8999
Sep 2015	114.4060	104.17659	124.6353	98.76149	130.0504
Oct 2015	114.5380	104.29646	124.7795	98.87492	130.2010
Nov 2015	114.6700	104.41629	124.9237	98.98831	130.3517
Dec 2015	114.8020	104.53609	125.0679	99.10165	130.5023

Figure 6.5.3: Forecast results by Holt's Method Exponential Smoothing Model for 2012-2015

Jan 2016	114.9340	104.65586	125.2121	99.21495	130.6530
Feb 2016	115.0660	104.77560	125.3564	99.32819	130.8038
Mar 2016	115.1980	104.89531	125.5007	99.44139	130.9546
Apr 2016	115.3300	105.01499	125.6450	99.55455	131.1055
May 2016	115.4620	105.13464	125.7894	99.66765	131.2564
Jun 2016	115.5940	105.25426	125.9338	99.78072	131.4074
Jul 2016	115.7260	105.37385	126.0782	99.89373	131.5584
Aug 2016	115.8581	105.49341	126.2227	100.00670	131.7094
Sep 2016	115.9901	105.61294	126.3672	100.11962	131.8605
Oct 2016	116.1221	105.73243	126.5117	100.23249	132.0116
Nov 2016	116.2541	105.85190	126.6563	100.34532	132.1628
Dec 2016	116.3861	105.97134	126.8008	100.45810	132.3141
Jan 2017	116.5181	106.09074	126.9454	100.57084	132.4653
Feb 2017	116.6501	106.21012	127.0901	100.68353	132.6167
Mar 2017	116.7821	106.32947	127.2347	100.79617	132.7680
Apr 2017	116.9141	106.44878	127.3794	100.90877	132.9195
May 2017	117.0461	106.56807	127.5242	101.02132	133.0709
Jun 2017	117.1781	106.68733	127.6689	101.13383	133.2224
Jul 2017	117.3101	106.80655	127.8137	101.24629	133.3740
Aug 2017	117.4421	106.92575	127.9585	101.35870	133.5256
Sep 2017	117.5742	107.04491	128.1034	101.47107	133.6772
Oct 2017	117.7062	107.16405	128.2483	101.58339	133.8289
Nov 2017	117.8382	107.28315	128.3932	101.69566	133.9807
Dec 2017	117.9702	107.40223	128.5381	101.80789	134.1325

Figure 6.5.4: Forecast results by Holt's Method Exponential Smoothing Model for 2016-2017

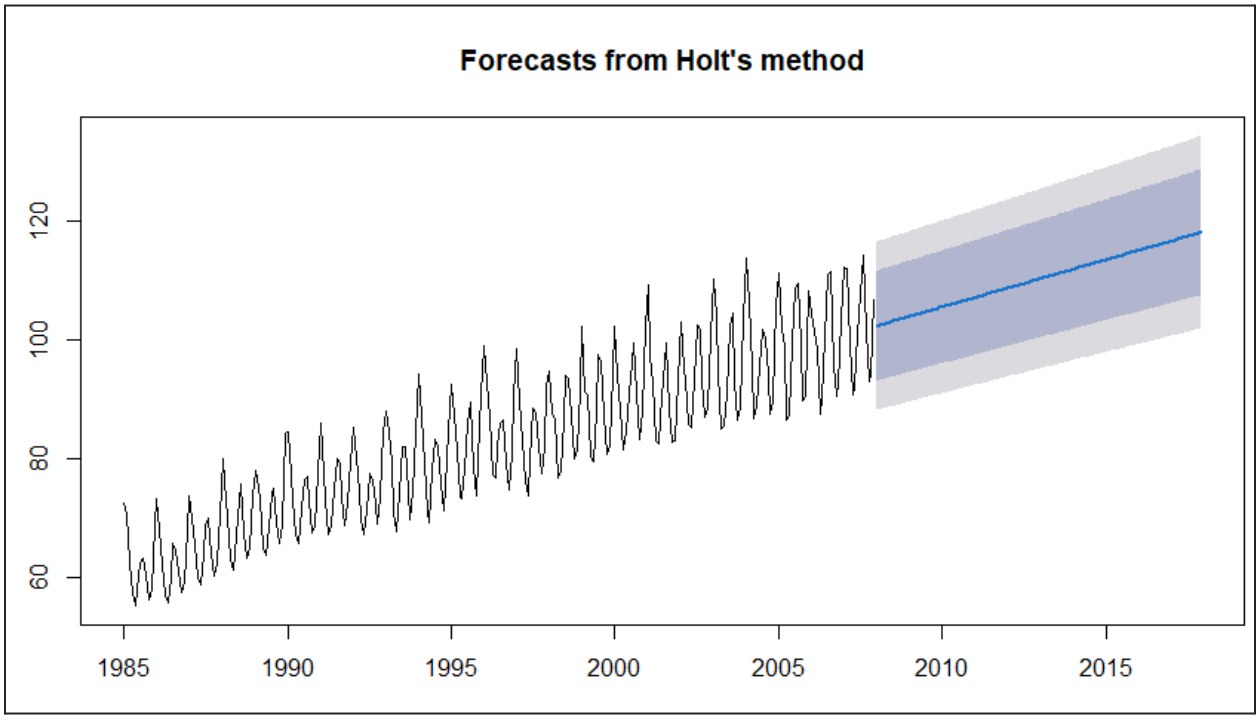

Figure 6.5.5: Forecast using Holt's Method and Capturing a positive trend in data

```
> error_holt = test - forecast_holt$mean
> mse_holt = mean(error_holt*error_holt)
> print(mse_holt)
[1] 171.7895
> rmse_holt = sqrt(mse_holt)
> print(rmse_holt)
[1] 13.10685
```

Figure 6.5.5: Root Mean Square Error of Holt's Method

Figure 6.5.5 shows how we calculate the Root Mean Square Error (RMSE) for the Holt's Method Exponential Smoothing model, we will use the test data to minus the forecasted value by the model to get the value of error. Then we will calculate the Mean Square Error (MSE) by getting the mean of error multiplied by error (error^2). Lastly we will square root the MSE and the value of RMSE of the model is 13.10685. We will not use the accuracy(),a built-in function in R, to evaluate the performance of the model as the accuracy as the function evaluates the model based on the train data. By right, to avoid overfitting, we will evaluate the model based on test data, hence we are comparing the predicted test data and actual value of test data to calculate the RMSE as the evaluation of the model.

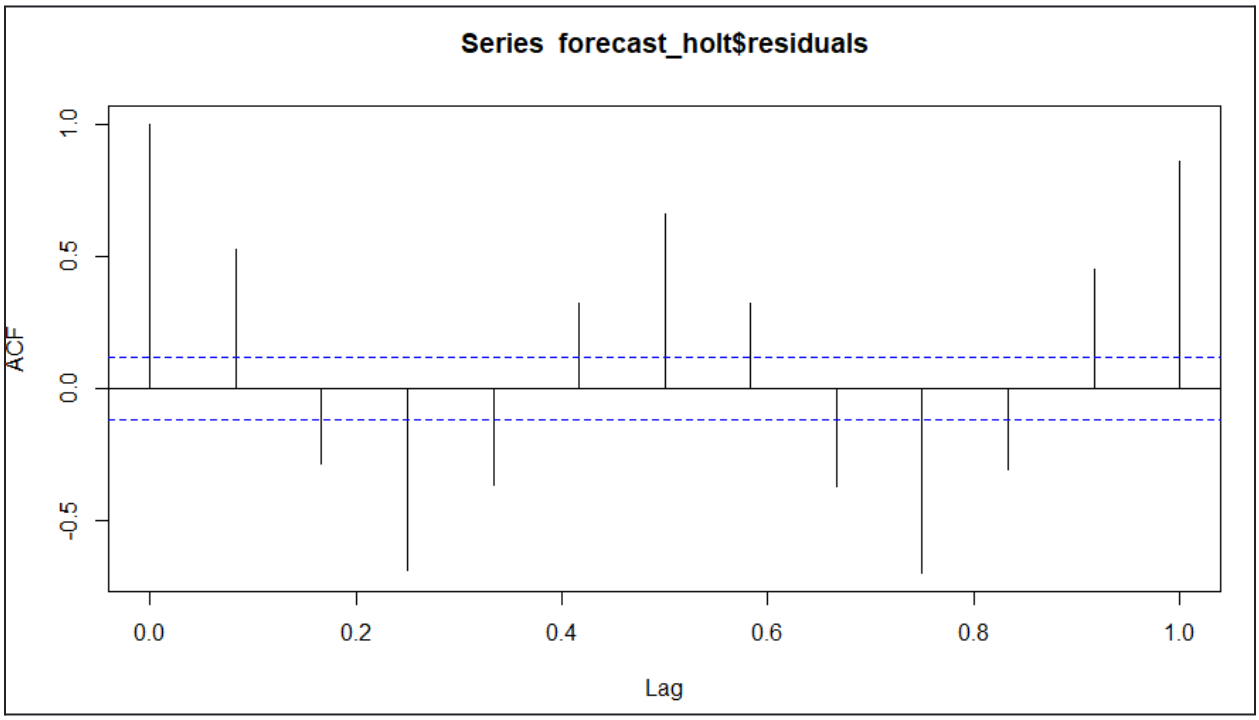

Figure 6.5.6 ACF Correlogram of Residuals of the Forecast with Holt's Method Model

We run the Holt's method model forecasting and we can see that at lag 0.0 has a spike that exceeds the 95% confidence interval. The 95% confidence interval is indicated by the blue dotted lines. Since all the acf values at all lag above lag 0 have no significant different from the value of 0, the forecast SES model is stationary.It shows that there is spikes before lag 0.5 which means that residuals of Holt's Method Exponential Smoothing does not shows a white noise pattern. Hence, we can get the conclusion from the ACF of residuals and the Box-Test that both showing that the residuals do not shows a white noise pattern.

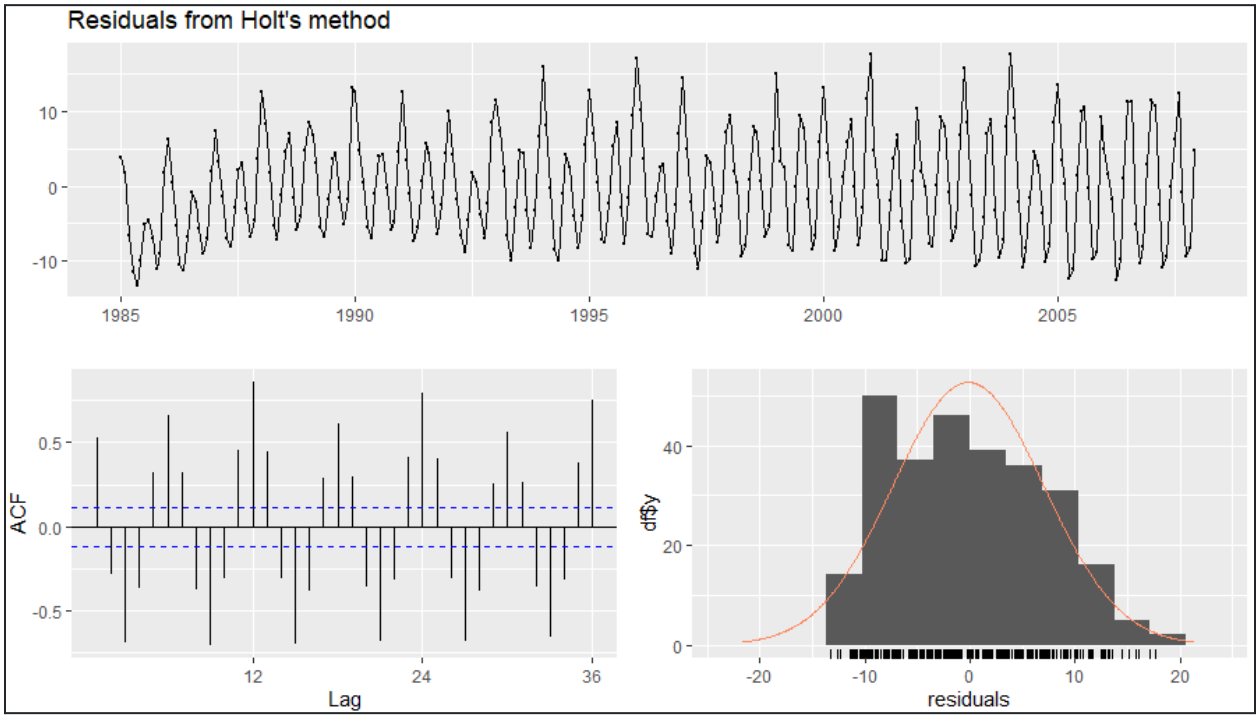

Figure 6.5.7 Holt's Method Residuals Detailed Charts

The figure above shows the details of Holt's Method model residuals using check residuals() function. The top of the chart shows that the value of the residuals changes at different times and we can see that between 1985 to 2007,there are 29 residuals that exceed the value of 10 and -10 while the rest are within the range of 10 to -10. At the bottom part on the right side, there is a chart that shows the normal distribution of the residuals where it shows the distribution of the residuals is not following a normal distribution with zero means and constant variance.

```
> Box.test(forecast_holt$residuals,lag=12)

        Box-Pierce test

data:  forecast_holt$residuals
X-squared = 904.41, df = 12, p-value < 2.2e-16

> Box.test(forecast_holt$residuals,lag=12,type = "Ljung-Box")

        Box-Ljung test

data:  forecast_holt$residuals
X-squared = 935.77, df = 12, p-value < 2.2e-16
```

Figure 6.5.8 Box Test and Ljung Box Test of Holt's Method Exponential Smoothing Model

Furthermore, we will use the Box-Pierce and Ljung Box test to determine whether the Holt's Method Exponential Smoothing model residual shows a white noise pattern or not. This is due the fact that if there are any signals in the model's residuals, we must update the model otherwise the forecasting process would not be successful.

Hypothesis of Box-Pierce Test

1. H_o : The Holt's Method Exponential Smoothing model does not show a lack of fit.
2. H_1: The Holt's Method Exponential Smoothing model does show a lack of fit.

Figure 6.5.8 shows that the p-value is equal to 2.2e-16, which is smaller than 0.05 and indicates that we reject the null hypothesis and conclude that the Holt's Method Exponential Smoothing model does show a lack of fit and its residuals does not show a white noise pattern.

6.6 TBATS

```
> model_tbats <- tbats(train)
> summary(model_tbats)
                  Length Class  Mode
lambda                 1 -none- numeric
alpha                  1 -none- numeric
beta                   1 -none- numeric
damping.parameter      1 -none- numeric
gamma.one.values       1 -none- numeric
gamma.two.values       1 -none- numeric
ar.coefficients        1 -none- numeric
ma.coefficients        2 -none- numeric
likelihood             1 -none- numeric
optim.return.code      1 -none- numeric
variance               1 -none- numeric
AIC                    1 -none- numeric
parameters             2 -none- list
seed.states           15 -none- numeric
fitted.values        276 ts     numeric
errors               276 ts     numeric
x                   4140 -none- numeric
seasonal.periods       1 -none- numeric
k.vector               1 -none- numeric
y                    276 ts     numeric
p                      1 -none- numeric
q                      1 -none- numeric
call                   2 -none- call
series                 1 -none- character
method                 1 -none- character
```

Figure 6.6.1 Summary information for TBATS model

The last model that we use for this assignment is the TBATS model. By using the summary() function, we can get a variety of information from the TBATS model such as lambda value, alpha value, beta value, AIC value, variance and so on. From figure 6.6.1, we can conclude that the value of alpha, beta and lambda is 1. Akaike information criterion(AIC) is 1.

```
> Box.test(for_tbats$residuals, lag =12)

        Box-Pierce test

data:  for_tbats$residuals
X-squared = 13.363, df = 12, p-value = 0.3432
```

Figure 6.6.2 Result of Box-Pierce Test for TBATS

```
> Box.test(fit$residuals, lag =12)

        Box-Pierce test

data:  fit$residuals
X-squared = 29.587, df = 12, p-value = 0.003221
```

Figure 6.6.3 Result of Box Test using fit$residuals for ETS model

```
> Box.test(forecast$residuals, lag =12)

        Box-Pierce test

data:  forecast$residuals
X-squared = 29.587, df = 12, p-value = 0.003221
```

Figure 6.6.4 Result of Box Test using forecast$residuals for ETS model

Figure 6.6.2 shows the result of the Box-Pierce Test for TBATS model. The figure shows that we are using forecasting values in doing the Box-Pierce test. This is because we are unable to get fit data's residuals from the model but only from forecasts data. Nevertheless, the p-value of the box test will be the same either using fit data or forecast values. Figure 6.6.3 and Figure 6.6.4 will show that the p-value from Box-Pierce test for ETS model are the same using fit$residuals and forecast$residuals. Therefore, we are using residuals in the forecast to calculate the p-value for determining whether the model has a white noise pattern or not.

Hypothesis of Box-Pierce Test

1. H_o : The TBATS model does not show a lack of fit.
2. H_1 : The TBATS model does show a lack of fit.

Figure 6.6.2 shows the p-value is 0.3435 which is larger than 0.05. Thus, we fail to reject null hypothesis and conclude that the TBATS model does not show a lack of fit. Since, the TBATS model passed the box test, we will proceed to the next step which is forecasting with the TBATS model.

```
> for_tbats <- forecast::forecast(model_tbats,h=120)
> df_tbats = as.data.frame(for_tbats)
> print(for_tbats)
         Point Forecast     Lo 80     Hi 80     Lo 95     Hi 95
Jan 2008      115.38686 112.10885 118.76072 110.41144 120.58649
Feb 2008      108.79101 105.18843 112.51698 103.32989 114.54077
Mar 2008      101.90496  98.31016 105.63121  96.45881 107.65862
Apr 2008       92.63448  89.26799  96.12792  87.53568  98.03026
May 2008       91.72878  88.34071  95.24679  86.59813  97.16340
Jun 2008      100.11881  96.38278 103.99965  94.46182 106.11457
Jul 2008      108.19116 104.12306 112.41820 102.03183 114.72231
Aug 2008      109.38413 105.24369 113.68747 103.11570 116.03363
Sep 2008       99.71337  95.91434 103.66288  93.96220 105.81656
Oct 2008       93.25505  89.67727  96.97558  87.83921  99.00482
Nov 2008       96.31846  92.59447 100.19221  90.68176 102.30553
Dec 2008      110.43773 106.13053 114.91973 103.91886 117.36553
Jan 2009      118.34952 113.68775 123.20245 111.29476 125.85147
Feb 2009      111.02374 106.60122 115.62975 104.33183 118.14488
Mar 2009      103.58915  99.41122 107.94267  97.26820 110.32086
Apr 2009       93.88537  90.04752  97.88680  88.07980 100.07360
May 2009       92.75511  88.90796  96.76875  86.93641  98.96327
Jun 2009      101.05930  96.80246 105.50333  94.62210 107.93443
Jul 2009      109.05483 104.38609 113.93238 101.99607 116.60209
Aug 2009      110.13429 105.33881 115.14807 102.88535 117.89396
Sep 2009      100.30650  95.86224 104.95681  93.58986 107.50518
Oct 2009       93.74019  89.51203  98.16806  87.35153 100.59609
Nov 2009       96.75977  92.31554 101.41795  90.04614 103.97394
Dec 2009      110.88603 105.69902 116.32757 103.05214 119.31543
```

Figure 6.6.5 Forecast results by TBATS for 2008-2009

Jan 2010	118.77723	113.11767	124.71995	110.23167	127.98527
Feb 2010	111.38240	105.97589	117.06472	103.22092	120.18919
Mar 2010	103.88926	98.75256	109.29314	96.13698	112.26665
Apr 2010	94.12994	89.39029	99.12091	86.97865	101.86922
May 2010	92.97285	88.20625	97.99703	85.78270	100.76566
Jun 2010	101.27344	95.98811	106.84979	93.30281	109.92497
Jul 2010	109.26370	103.46073	115.39214	100.51464	118.77429
Aug 2010	110.32516	104.36388	116.62694	101.33969	120.10733
Sep 2010	100.46394	94.94300	106.30593	92.14429	109.53477
Oct 2010	93.87353	88.62835	99.42912	85.97142	102.50196
Nov 2010	96.88458	91.38230	102.71816	88.59721	105.94714
Dec 2010	111.01578	104.60975	117.81409	101.36963	121.57984
Jan 2011	118.90336	111.93393	126.30673	108.41146	130.41065
Feb 2011	111.48976	104.85351	118.54602	101.50192	122.46041
Mar 2011	103.98018	97.69706	110.66738	94.52615	114.37975
Apr 2011	94.20476	88.42835	100.35849	85.51530	103.77717
May 2011	93.03996	87.25262	99.21115	84.33617	102.64200
Jun 2011	101.33983	94.94739	108.16266	91.72831	111.95847
Jul 2011	109.32877	102.33754	116.79761	98.81944	120.95575
Aug 2011	110.38485	103.23100	118.03445	99.63362	122.29622
Sep 2011	100.51332	93.91368	107.57675	90.59730	111.51468
Oct 2011	93.91545	87.66970	100.60616	84.53334	104.33886
Nov 2011	96.92389	90.39700	103.92203	87.12173	107.82890
Dec 2011	111.05671	103.48628	119.18094	99.68993	123.71953
Jan 2012	118.94319	110.73740	127.75704	106.62522	132.68420
Feb 2012	111.52370	103.73858	119.89305	99.83985	124.57485
Mar 2012	104.00894	96.66421	111.91172	92.98849	116.33546
Apr 2012	94.22843	87.49924	101.47514	84.13378	105.53426
May 2012	93.06120	86.34183	100.30349	82.98349	104.36278
Jun 2012	101.36086	93.96302	109.34114	90.26797	113.81693
Jul 2012	109.34938	101.28404	118.05698	97.25817	122.94378
Aug 2012	110.40376	102.17605	119.29400	98.07176	124.28644
Sep 2012	100.52897	92.96109	108.71294	89.18832	113.31163
Oct 2012	93.92874	86.78722	101.65791	83.22923	106.00372
Nov 2012	96.93635	89.49411	104.99747	85.78859	109.53269
Dec 2012	111.06968	102.46074	120.40196	98.17693	125.65552

Figure 6.6.6 Forecast results by TBATS for 2010-2012

Jan 2013	118.95581	109.64869	129.05293	105.02030	134.74048
Feb 2013	111.53445	102.72687	121.09717	98.34956	126.48693
Mar 2013	104.01805	95.72910	113.02472	91.61202	118.10410
Apr 2013	94.23593	86.65958	102.47466	82.89868	107.12367
May 2013	93.06794	85.51995	101.28211	81.77534	105.91996
Jun 2013	101.36753	93.07578	110.39795	88.96458	115.49962
Jul 2013	109.35592	100.33532	119.18751	95.86533	124.74496
Aug 2013	110.40975	101.22662	120.42596	96.67871	126.09098
Sep 2013	100.53393	92.10414	109.73525	87.93170	114.94229
Oct 2013	93.93295	85.99352	102.60539	82.06600	107.51589
Nov 2013	96.94030	88.68209	105.96752	84.59917	111.08172
Dec 2013	111.07379	101.53831	121.50475	96.82651	127.41744
Jan 2014	118.95981	108.66923	130.22488	103.58713	136.61386
Feb 2014	111.53786	101.81636	122.18758	97.01794	128.23086
Mar 2014	104.02094	94.88713	114.03397	90.38126	119.71901
Apr 2014	94.23831	85.90316	103.38222	81.79351	108.57658
May 2014	93.07007	84.77909	102.17187	80.69341	107.34504
Jun 2014	101.36964	92.27546	111.36008	87.79637	117.04133
Jul 2014	109.35799	99.47893	120.21811	94.61583	126.39713
Aug 2014	110.41165	100.36894	121.45922	95.42787	127.74814
Sep 2014	100.53550	91.32938	110.66962	86.80227	116.44150
Oct 2014	93.93428	85.27532	103.47249	81.01949	108.90774
Nov 2014	96.94155	87.94667	106.85638	83.52801	112.50912
Dec 2014	111.07509	100.70216	122.51650	95.60914	129.04285
Jan 2015	118.96108	107.78057	131.30140	102.29382	138.34402
Feb 2015	111.53894	100.98947	123.19042	95.81504	129.84325
Mar 2015	104.02186	94.12178	114.96326	89.26830	121.21375
Apr 2015	94.23907	85.21492	104.21886	80.79307	109.92280
May 2015	93.07075	84.10436	102.99304	79.71300	108.66689
Jun 2015	101.37031	91.54589	112.24905	86.73667	118.47283
Jul 2015	109.35864	98.69750	121.17138	93.48123	127.93277
Aug 2015	110.41226	99.58558	122.41598	94.29089	129.28997
Sep 2015	100.53600	90.62108	111.53572	85.77462	117.83775
Oct 2015	93.93471	84.61812	104.27706	80.06631	110.20526
Nov 2015	96.94194	87.27309	107.68199	82.55142	113.84104
Dec 2015	111.07551	99.93561	123.45718	94.49815	130.56095

Figure 6.6.7 Forecast results by TBATS for 2013-2015

Jan 2016	118.96149	106.96512	132.30326	101.11238	139.96144
Feb 2016	111.53928	100.23002	124.12460	94.71511	131.35192
Mar 2016	104.02215	93.41820	115.82975	88.24965	122.61359
Apr 2016	94.23930	84.58167	104.99966	79.87656	111.18464
May 2016	93.07096	83.48301	103.76008	78.81400	109.90691
Jun 2016	101.37052	90.87345	113.08014	85.76408	119.81685
Jul 2016	109.35885	97.97664	122.06336	92.43897	129.37572
Aug 2016	110.41245	98.86233	123.31197	93.24551	130.73989
Sep 2016	100.53616	89.96660	112.34747	84.82894	119.15178
Oct 2016	93.93484	84.01036	105.03174	79.18843	111.42731
Nov 2016	96.94207	86.64968	108.45701	81.65123	115.09643
Dec 2016	111.07564	99.22559	124.34088	93.47322	131.99285
Jan 2017	118.96161	106.20922	133.24516	100.02158	141.48812
Feb 2017	111.53939	99.52549	125.00352	93.69875	132.77697
Mar 2017	104.02224	92.76500	116.64556	87.30765	123.93674
Apr 2017	94.23938	83.99333	105.73531	79.02835	112.37816
May 2017	93.07103	82.90529	104.48328	77.98137	111.08058
Jun 2017	101.37059	90.24779	113.86424	84.86263	121.08976
Jul 2017	109.35892	97.30547	122.90546	91.47223	130.74320
Aug 2017	110.41251	98.18846	124.15840	92.27520	132.11482
Sep 2017	100.53621	89.35638	113.11480	83.95055	120.39860
Oct 2017	93.93488	83.44333	105.74557	78.37245	112.58755
Nov 2017	96.94211	86.06766	109.19052	80.81393	116.28902
Dec 2017	111.07568	98.56229	125.17776	92.51927	133.35390

Figure 6.6.8 Forecast results by TBATS for 2016-2017

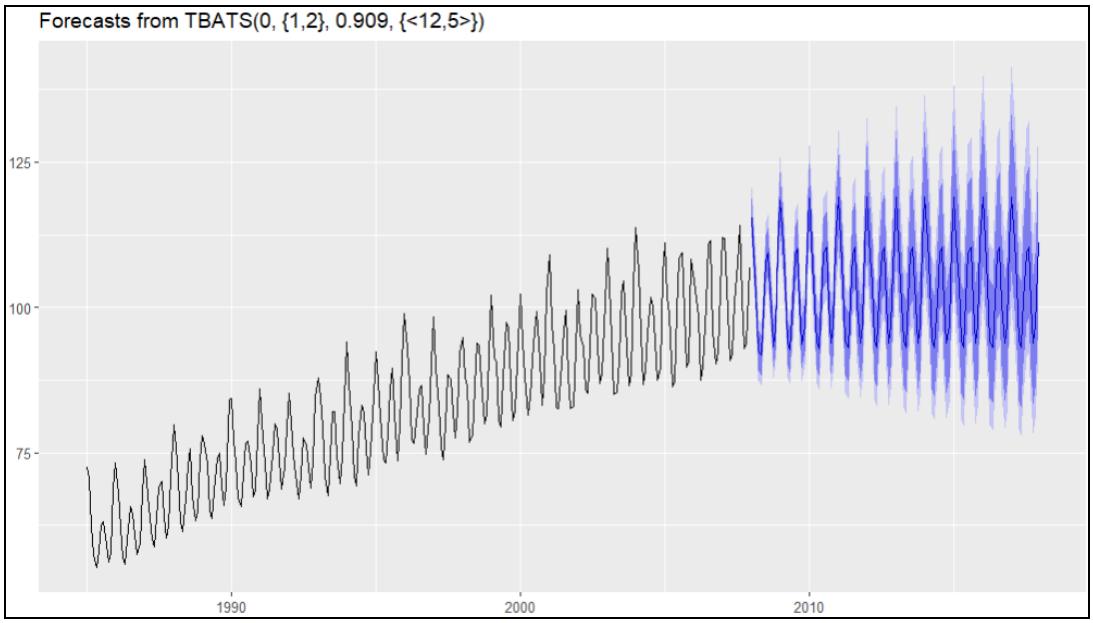

Figure 6.6.10 Forecast plot by TBATS model

Our test data for this assignment is from the years 2008 to 2017. Thus, we set h value equal to 120 which means 10 years(120 months) to forecast for 10 years using the forecasting model.

```
> error_tbats=test-for_tbats$mean
> mse_tbats = mean(error_tbats*error_tbats)
> print(mse_tbats)
[1] 14.6889
> rmse_tbats = sqrt(mse_tbats)
> print(rmse_tbats)
[1] 3.83261
```

Figure 6.6.11 Root Mean Square Error of TBATS model

To choose the best model among the forecasting models, we need to calculate the Root Mean Square Error (RMSE) for the TBATS model. We use the test data subtract with the forecasted value to get the error value. Next, to calculate the Mean Square Error(MSE) is equal to 14.6889, we are calculating by the mean of error multiply error which is error^2. Afterwards, we square root the MSE value to get the Root Mean Square Error(RMSE) of the model. For the TBATS model, its RMSE value is 3.83261.

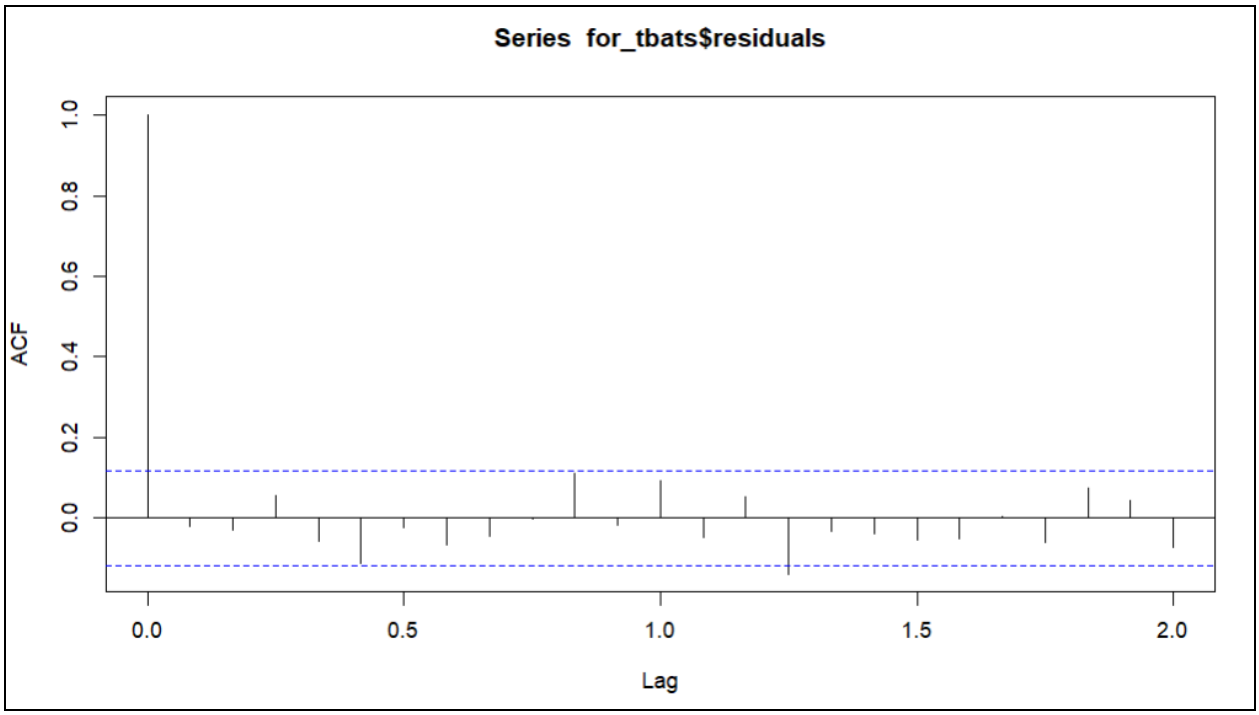

Figure 6.6.12 ACF graph of forecasted TBATS

From Figure 6.6.12 above, we can see that at lag 0.0 has a spike that exceeds the blue dotted line which indicates a 95% confidence interval. So, at lag 0.0, having one spike is exceeding the 95% confidence interval. Besides, we can also observed that there are no spike exceeds the blue dotted line accepts the spike between lag 1.0 and lag 1.5 are exceeds 95% confidence interval with a small value, so we can ignore it as it did not exceed the blue dotted line too much. In a nutshell, from the ACF graph we can conclude that there are having white noise which same as the results of the Box-Pierce Test shown above.

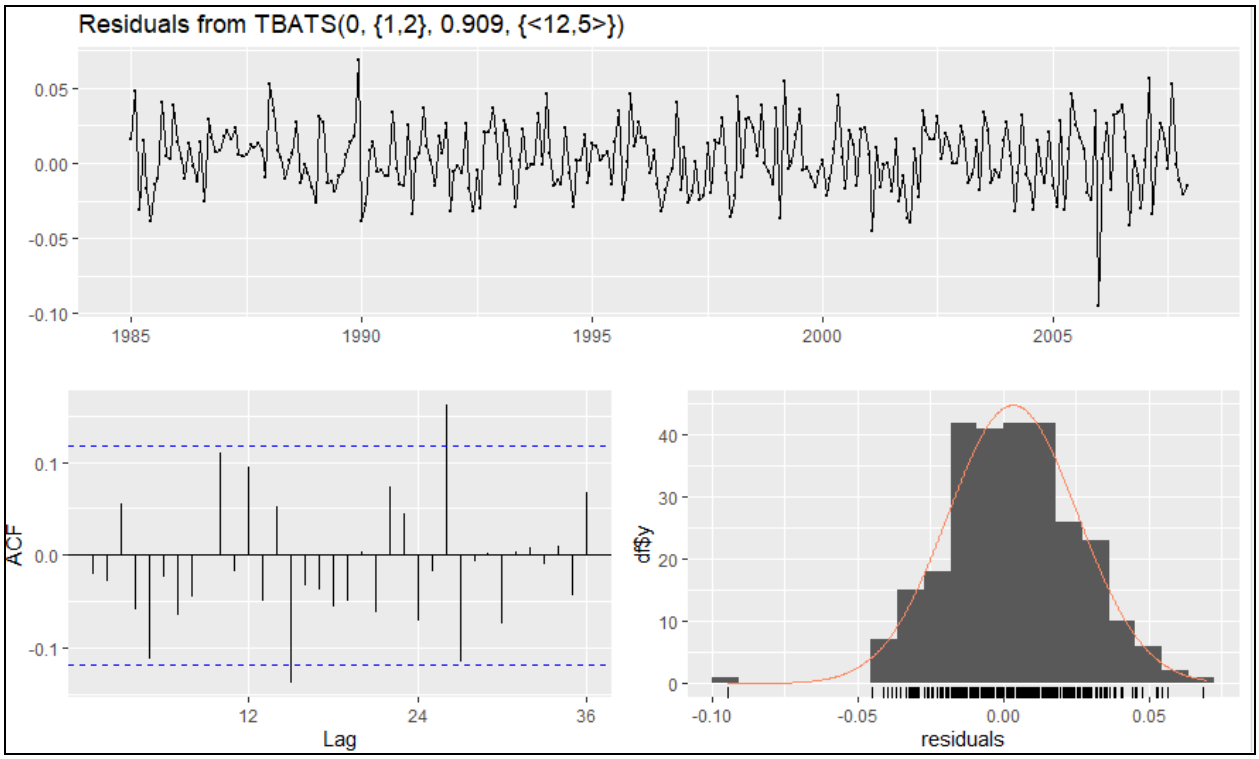

Figure 6.6.13 Residuals Detailed Charts of TBATS model

By using checkresiduals() function, we can obtain the graphs above, we can see that there are 6 residuals exceeds the value of 0.05 and -0.05 while the rest of the residuals are within the range of it shown by the above chart. The ACF graph, we can see that there are 2 residuals that exceed the value of 0.1 and 1.0. Lastly, the bottom right corner chart showing a left skewed distribution of the graph.

6.7 Model Evaluation

All the implemented time series models are evaluated and compared to choose the best time series model based on the metrics such as mean squared error (MSE) and root mean squared error (RMSE).

Time Series Model	MSE	RMSE	Pass Box-Test or not
ETS	14.95901	3.867688	No
ARIMA(2,0,1)(0,1,1)[12] with drift	132.9156	11.5289	Yes
ARIMA(1,0,0)(2,1,0)[12]	14.27835	3.778671	No
Multiplicative Holt Winter	59.28459	7.699649	No
Holt's Method Exponential Smoothing	171.7895	13.10685	No
TBATS	14.6889	3.83261	Yes

Table 1: The evaluation metrics on time series models

Based on Table 1, we can observe that only 2 of the models have passed the Box-Pierce Test which are ARIMA(2,0,1)(0,1,1)[12] with drift and TBATS model. Besides, we can observe that the model with highest RMSE is Holt's Method Exponential Smoothing with the RMSE of 13.10685 while the model with lowest RMSE is ARIMA(1,0,0)(2,1,0)[12] with the RMSE of 3.77861. In this case, we would choose ARIMA(1,0,0)(2,1,0)[12] as the best model to forecast the electrical production, as it has the lowest RMSE. RMSE is a way to measure the error made by a model in predicting the actual value of the data. Hence, we will choose the model with the lowest RMSE value which will minimize the error in predicting data. That's why we choose ARIMA(1,0,0)(2,1,0)[12] as the best model as it has the lowest RMSE value. Although it does not pass the Box-Pierce test due to its residuals is not a white noise pattern as there are still some signals remaining in the residuals, it still can predict the actual value of the data with the lowest

error among the model we have tested. In short, ARIMA(1,0,0)(2,1,0)[12] is the best model to predict the monthly electrical production data.

6.8 Forecasting

Best forecast model : ARIMA(1,0,0)(2,1,0)[12]
Parameters:
p = 1, d = 0, q = 0, P = 2, D = 1, Q = 0

The model's coefficient information is as below :
ar1, $\phi_1 = 0.828$
sar1,$\Phi_1 =$ -0.5233
sar2,$\Phi_2 = -0.3847$

Equation of best model:
$(1-\phi_1 B)(1 - \Phi_1 B^{12} - \Phi_2 B^{24})(1-B^{12})Y_t = \varepsilon_t$
$\phi_1 B$ = AR(1), $\Phi_1 B^{12}$ = SAR(1), $\Phi_2 B^{24}$ = SAR(2), $(1-B^{12})$ = Seasonal Differencing

$(1-0.828B)(1 + 0.5233B^{12} + 0.3847B^{24})(1-B^{12})Y_t = \varepsilon_t$

$(1 + 0.5233B^2 + 0.3847B^{24} - B^{12} - 0.5233B^{14} - 0.3847B^{36} - 0.828B - 0.4332924B^3 - 0.3185316B^{25} + 0.828B^{13} + 0.4332924B^{15} + 0.3185316B^{37})Y_t = \varepsilon_t$

$Y_t + 0.5233Y_{t-2} + 0.3847Y_{t-24} - Y_{t-12} - 0.5233Y_{t-14} - 0.3847Y_{t-36} - 0.828Y_{t-1} - 0.4332924Y_{t-3} - 0.3185316Y_{t-25} + 0.828Y_{t-13} + 0.4332924Y_{t-15} + 0.3185316Y_{t-37} = \varepsilon_t$

Final Equation:
$Y_t = - 0.5233Y_{t-2} - 0.3847Y_{t-24} + Y_{t-12} + 0.5233Y_{t-14} + 0.3847Y_{t-36} + 0.828Y_{t-1} + 0.4332924Y_{t-3} + 0.3185316Y_{t-25} - 0.828Y_{t-13} - 0.4332924Y_{t-15} - 0.3185316Y_{t-37} + \varepsilon_t$

6.8.1 Test Significance of the Coefficients and Stationary Test

```
> coeftest(bestmodel)

z test of coefficients:

      Estimate Std. Error z value  Pr(>|z|)
ar1   0.827981   0.037343 22.1723 < 2.2e-16 ***
sar1 -0.523342   0.064204 -8.1512 3.603e-16 ***
sar2 -0.384684   0.066185 -5.8123 6.163e-09 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Figure 6.8.1.1 : Coefficient test of ARIMA(1,0,0)(2,1,0)[12]

After identifying the coefficient of the model we would like to test the significance of the coefficient of the model.

H_o : Coefficient is not significant

H_1 : Coefficient is significant

From the figure above, we can see that the p-value of ar1, sar1, sar2 and c are 2.2e-16 (0.00000000000000022), 3.603e-19 (0.000000000000000003603) and 6.163e-09 (0.000000006163) respectively. Based on the figure above, it shows that all p-values are lower than $\alpha = 0.05$, therefore we successfully reject H0 and conclude that all 3 coefficients are significant.

```
> Box.test(bestmodel$residuals,lag=12)

        Box-Pierce test

data:  bestmodel$residuals
X-squared = 37.774, df = 12, p-value = 0.0001672
```

Figure 6.8.1.2 : Box test of ARIMA(1,0,0)(2,1,0)[12]

Furthermore, we will use the Box-Pierce test to determine whether the ARIMA(1,0,0)(2,1,0)[12] model's residual shows a white noise pattern or not. This is due the fact that if there are any signals in the model's residuals, we must update the model otherwise the forecasting process may not be successful.

Hypothesis of Box-Pierce test

H_o : The ARIMA(1,0,0)(2,1,0)[12] model does not show a lack of fit.

H_1 : The ARIMA(1,0,0)(2,1,0)[12] model does show a lack of fit.

Figure 6.8.1.2 shows that the p-value is equal to 0.00001672, which is smaller than 0.05 and indicates that we reject the null hypothesis and conclude that the ARIMA(1,0,0)(2,1,0)[12] model does show a lack of fit and does not have a white noise in its residuals.

6.8.2 Forecasting

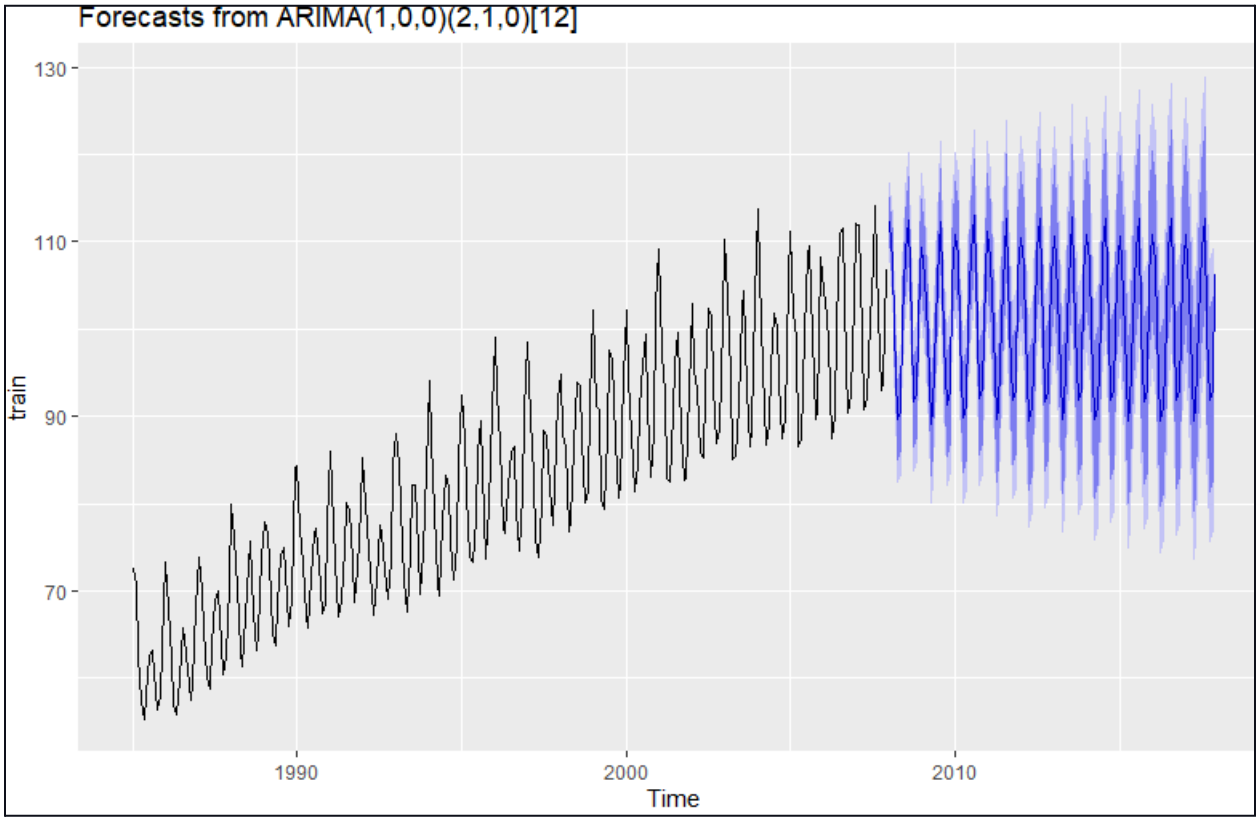

Figure 6.8.2.1 : Graph of Visualizing the Forecast from best model

Since our test data is from 2008 - 2017,which is 10 years, then we will use the forecasting model to forecast for 10 years by using h =120 which means we are doing forecasting for 120 months (10 years).

6.8.3 Evaluation for Best Model

```
> error=test-forecast_bestmodel$mean
> mse = mean(error*error)
> print(mse)
[1] 14.27835
> rmse = sqrt(mse)
> print(rmse)
[1] 3.778671
```

Figure 6.8.3.1 : Evaluation of the best model

Figure 6.8.3.1 shows how we calculate the Root Mean Square Error (RMSE) for the best model, we will use the test data to minus the forecasted value by the model to get the value of error. Then we will calculate the Mean Square Error (MSE) by getting the mean of error multiplied by error (error^2). Lastly we will square root the MSE and the value of RMSE of the model is 3.778671. We will not use the accuracy(),a built-in function in R, to evaluate the performance of the model as the accuracy as the function evaluates the model based on the train data. By right, to avoid overfitting, we will evaluate the model based on test data, hence we are comparing the predicted test data and actual value of test data to calculate the RMSE as the evaluation of the model.

7.0 Discussions And Interpretations

7.1 Discussion

After all the calibration and testing, we decided to fit our dataset with ARIMA(1,0,0)(2,1,0)[12] as the best model to perform forecasting. Our decision is being made with the evaluation of the metrics on all the time series models shown in Table 1, the mean squared error (MSE) and root mean squared error (RMSE) for the ARIMA(1,0,0)(2,1,0)[12] model is the lowest among the models that we have trained.

In Figure 6.8.2.1, we can see that the future movement for the Electricity Production in United States is having an growing trend from the year of 2008 to 2017. With this forecasting result, it inform the investment decisions about power generation and supporting network infrastructure.It

increase the efficiency and revenues for the electrical generating company by helping thm to plan their capacity and operations in order to supply all consumers with the required electricity consumption.

7.2 Limitations And Recommendations

The main limitation of this project is that the time series data does not contain the latest monthly electricity production of the United States as it only ranged from 2008 to 2017. Therefore, it is recommended that the future work of time series analysis on the latest Monthly Electricity Production should be conducted to provide more useful information to the ARIMA(1,0,0)(2,1,0)[12] to forecast the monthly electricity production of the United States in the future.

Besides that, there are only six of the time series models being implemented in this project, so it is still unknown that there is any time series model which has better fit to the time series data than the ARIMA(1,0,0)(2,1,0)[12](best model). Thus, other time series models such as regression, Vector Autoregression (VAR), Vector Autoregression Moving Average (VARMA), and Vector Autoregression Moving Average with Exogenous Regressors (VARMAX) are recommended to be implemented and evaluation before implying that the ARIMA(1,0,0)(2,1,0)[12] (best model) is the best possible time series model for Monthly Electricity Production in United States.

8.0 Conclusion

As a result of this study, we were successful in achieving the objective we had set at the beginning which is to analyze the time-series data from the dataset titled "Monthly Electrical Production," which was obtained from Kaggle. The ARIMA(1,0,0)(2,1,0)[12] model has been successfully identified as the best model for predicting and forecasting the electrical production for the next 10 years, which corresponds to the electrical production years from 2008 to 2017.

We also like to thank our tutor, Dr. Tan Pei Ling, and our lecture, Dr. Chin Wan Yoke. We could not have completed our project successfully without them. They were patient with us as they provided us with advice on how to learn statistics and project guidelines. We believe that this knowledge would be beneficial to us in a variety of ways after graduating from Tunku Abdul Rahman University College.

9.0 Reference

1. Chourasia, A., 2020. Decomposition in time series data. Medium. Available at: https://medium.com/analytics-vidhya/decomposition-in-time-series-data-b20764946d63 [Accessed September 26, 2022].

2. Brownlee, J., 2020. How to decompose time series data into trend and seasonality. Machine Learning Mastery. Available at: https://machinelearningmastery.com/decompose-time-series-data-trend-seasonality/ [Accessed September 27, 2022].

3. Hall, M. & Degen, B., 1980. Forecast. Amazon. Available at: https://docs.aws.amazon.com/forecast/latest/dg/aws-forecast-recipe-ets.html [Accessed September 28, 2022].

4. Jofipasi, C.A., Miftahuddin and Hizir (2018) 'Selection for the best ETS (error, trend, seasonal) model to forecast weather in the Aceh Besar District', IOP Conference Series: Materials Science and Engineering, 352(1). Available at: https://doi.org/10.1088/1757-899X/352/1/012055.

5. Nadeem, 2021. Time Series forecasting using TBATS model. Medium. Available at: https://medium.com/analytics-vidhya/time-series-forecasting-using-tbats-model-ce8c429442a9 [Accessed September 28, 2022].

10.0 Appendix

```
#Import libraries
library(ggplot2)
library(gridExtra)
library(dplyr)
library(astsa)
library(fpp2)
library(forecast)
library(tseries)
library(graphics)
library(MLmetrics)
library(lmtest)
library(lubridate)
library(repr)

################################
#   Descriptive & Preprocessing   #
################################

#Importing Time Series Dataset
print(getwd())
setwd("C:/Users/cecil/Downloads/")
data <- read.csv("Electric_Production.csv")

#Display Summary of Dataset
summary(data)
count(data)

#Listing our some data from each column
str(data)

#Checking for any null/missing value
for(i in colnames(data)){
  cat(sum(is.na(i)),"Null values in the coulumn : ", i, "\n")
}

#Changing data format from char into date
data$DATE <- mdy(data$DATE)
summary(data)
str(data)

#Visualize the density for Value Column
options(repr.plot.width=12, repr.plot.height=12)
valuePlot = ggplot(data, aes(Value)) + geom_histogram(bins = 10, aes(y = ..density..)
                                                      ,col ="red",fill = "red", alpha=0.3) + geom_density() + xlim(c(0, 140))
valuePlot

#Check Duplicate Value
cat("There are ",length(unique(data[["DATE"]])),"  days of record in the dataset")

#plot ori data
y <- data$Value
y<- ts(y, start = c(1985,1), end = c(2017, 12), frequency = 12)
plot(y,xlab = "Year", ylab="Electricity Production", main="Monthly Electricity Production from 1985 to 2017")
```

```r
ggseasonplot(y, year.labels=TRUE, year.labels.left=TRUE) +
  ylab("Electricity Production") +
  ggtitle("Seasonal plot: Monthly Electricity Production in United State")

adf.test(y)

#-----------------------------------------------------------------------------------------------------------

#Split the dataset into training and testing data
## Split univariate time series data into training and testing
# By year
# Use train data from 1985 to 2007 for forecasting
train = window(y, start=1985, end=c(2007,12))
print(train)
# Use remaining data from 2008 to 2018 to test accuracy
test = window(y, start=2008, end=c(2017,12))
print(test)

#Decomposition of time series
components <- decompose(train)
plot(components)

#ACF and PACF train
#correlogram, ACF and PACF
acf(train,main='Autocorrelations')
pacf(train,main='Partial Autocorrelations')
ggtsdisplay(train)

#accessing the stationary of time series data using adf test
#RESULT in our time series data is a seasonal data
#Because our time series is seasonal dataset and adf is only used to
#show non seasonal data,hence we have to do seasonal differencing first before stationary test
adf.test(train)
kpss.test(train)

#doing seasonal differencing
seasonalDiff <- diff(train,lag=12)
plot(seasonalDiff)
kpss.test(seasonalDiff)
#result in p value > 0.05,do not reject H0,time series data is stationary after first differencing
#correlogram
ggtsdisplay(seasonalDiff)
acf(seasonalDiff)
pacf(seasonalDiff)

#prove no need
seasonalDiff2 <- diff(seasonalDiff,lag=12)
acf(seasonalDiff2,main ="Original Time Series")
#ACF and PACF
ggtsdisplay(seasonalDiff)
```

```
#ETS
fit <- ets(train)
summary(fit)

#Box.test(forecast$residuals,lag=12)
Box.test(fit$residuals,lag=12)

plot(fit)
acf(fit$residuals,lag=12,na.action = na.pass)
forecast <- forecast(fit,h=120)
print(forecast)

error_ets=test-forecast$mean
#print(test_ets)
print(error_ets)
mse_ets = mean(error_ets*error_ets)
print(mse_ets)
rmse_ets = sqrt(mse_ets)
print(rmse_ets)
```

```
#ARIMA MODEL

#fit arima with auto arima
arimaTrain <- auto.arima(train,ic="aic",trace=TRUE)
Box.test(arimaTrain$residuals)
acf(arimaTrain$residuals,lag=12,na.action = na.pass)
print(arimaTrain)
#plot(train)
componentsTrain <- decompose(train)
plot(componentsTrain)
kpss.test(train)
#coeftest(train)

#check for stationary of model
options(repr.plot.width = 6,repr.plot.height = 4)
plot(arimaTrain)
summary(arimaTrain)
#the point is between the unit circle ,the model is stationary and invertable

#obtain forecast model to check stationary
forecast_arimaTrain <- forecast(arimaTrain,h=120)
Box.test(forecast_arimaTrain$residuals,lag=12)
print(accuracy(forecast_arimaTrain))
print(forecast_arimaTrain)
error=test-forecast_arimaTrain$mean
print(test)
print(error)
mse = mean(error*error)
print(mse)
rmse = sqrt(mse)
print(rmse)
plot(forecast_arimaTrain)
acf(arimaTrain$residuals)
accuracy(forecast_arimaTrain)
```

```
#Manually fit arima model
#find optimal parameters
#find p,d,q & P,D,Q
sarimaTrain_man <- arima(train,c(1,0,0),seasonal = list(order=c(2,1,0),period=12))
Box.test(sarimaTrain_man$residuals)
print(sarimaTrain_man)
plot(train)
componentsTrain <- decompose(train)
plot(componentsTrain)
kpss.test(train)
coeftest(sarimaTrain_man)

#check for stationary of model
options(repr.plot.width = 6,repr.plot.height = 4)
plot(sarimaTrain_man)
summary(sarimaTrain_man)
#the point is between the unit circle ,the model is stationary and invertable

#obtain forecast model to check stationary
forecast_sarimaTrain <- forecast(sarimaTrain_man,h=120)
Box.test(forecast_sarimaTrain$residuals,lag=12)
#print(accuracy(forecast_sarimaTrain))
#print(forecast_sarimaTrain)
error=test-forecast_sarimaTrain$mean
#print(test)
#print(error)
mse = mean(error*error)
print(mse)
rmse = sqrt(mse)
print(rmse)
plot(forecast_sarimaTrain)
acf(forecast_sarimaTrain$residuals)
accuracy(forecast_sarimaTrain)
```

```
#Holt's Method
holtTrain <-holt(train,h=120)
summary(holtTrain)
Box.test(holtTrain$residuals,lag=12)
acf(holtTrain$residuals,lag=12,na.action = na.pass)

#plot the forecast graph
forecast_holt <- forecast(holtTrain,h=120)
print(forecast_holt)
plot(forecast_holt)

error_holt = test - forecast_holt$mean
mse_holt = mean(error_holt*error_holt)
print(mse_holt)
rmse_holt = sqrt(mse_holt)
print(rmse_holt)

acf(forecast_holt$residuals,lag=12,na.action = na.pass)
checkresiduals(forecast_holt)
Box.test(forecast_holt$residuals,lag=12)
Box.test(forecast_holt$residuals,lag=12,type = "Ljung-Box")
```

```
#Holt Winter Model
hwTrain <-HoltWinters(train,seasonal="mult")
summary(hwTrain)
options(repr.plot.width = 6,repr.plot.height = 4)
plot(hwTrain)
acf(forecast_hw$residuals,na.action = na.pass,main="ACF of residuals of Holt's Winter Multiplicative model")
Box.test(forecast_hw$residuals,lag=12)
forecast_hw <- forecast(hwTrain,h=120)
print(forecast_hw)
plot(forecast_hw)
error_hw=test-forecast_hw$mean
mse_hw = mean(error_hw*error_hw)
print(mse_hw)
rmse_hw = sqrt(mse_hw)
print(rmse_hw)
```

```r
#TBATS Model
model_tbats <- tbats(train)
summary(model_tbats)

for_tbats <- forecast::forecast(model_tbats,h=120)
df_tbats = as.data.frame(for_tbats)
print(for_tbats)
acf(for_tbats$residuals)
autoplot(for_tbats)
checkresiduals(for_tbats)
#Ljung-Box test

#data:  Residuals from TBATS(0, {1,2}, 0.909, {<12,5>})
#Q* = 28.233, df = 15, p-value = 0.02015
#Model df: 9.   Total lags used: 24

Box.test(for_tbats$residuals, lag =12)

print(accuracy(for_tbats))
print(for_tbats)

error_tbats=test-for_tbats$mean
mse_tbats = mean(error_tbats*error_tbats)
print(mse_tbats)
rmse_tbats = sqrt(mse_tbats)
print(rmse_tbats)
#print(test)
#print(error_tbats)

plot(for_tbats)
acf(for_tbats$residuals)
accuracy(for_tbats)

acf(for_tbats$residuals,lag.max=10,na.action = na.pass)
```

```
#MODEL EVALUATION
accuracy(forecast)
accuracy(forecast_arimaTrain)
accuracy(forecast_hw)
accuracy(forecast_holt)
accuracy(for_tbats)

#Best Model >SARIMA
bestmodel <- arima(train,c(1,0,0),seasonal = list(order=c(2,1,0),period=12))
Box.test(bestmodel$residuals,lag=12)
print(bestmodel)
plot(train)
componentsTrain <- decompose(train)
plot(componentsTrain)
kpss.test(train)
coeftest(bestmodel)

#check for stationary of model
options(repr.plot.width = 6,repr.plot.height = 4)
plot(bestmodel)
summary(bestmodel)
#the point is between the unit circle ,the model is stationary and invertable

#obtain forecast model to check stationary
forecast_bestmodel <- forecast(bestmodel,h=120)
autoplot(forecast_bestmodel)
Box.test(forecast_bestmodel$residuals)
print(accuracy(forecast_bestmodel))
print(forecast_bestmodel)
error=test-forecast_bestmodel$mean
mse = mean(error*error)
print(mse)
rmse = sqrt(mse)
print(rmse)
plot(forecast_bestmodel)
acf(bestmodel$residuals)
accuracy(forecast_bestmodel)
```