Time Series Forecasting

Mini Project :5

SUPRASANNA PRADHAN
PGPBABI-O DEC'18, GROUP: 5

TABLE OF CONTENTS

1. The Assignment

1.1 Case Study: Consumable item A and item B

The attached data shows monthly demand of two different types of consumable items in a certain store from January 2002 to July 2017. The ultimate objective of this exercise is to predict sales for the period August 2017 to December 2018.

1. Read the data as time series objects in R. Plot the data. What are the major features you notice in the series? How do the two series differ?
2. Before a formal extraction of time series components is done, can you check for seasonal changes in the data for the two series separately? Particularly whether there are more variability in a season compared to the others, whether seasonal variations are changing across years etc. Compare the behavior of the two series.

- Decompose each series to extract trend and seasonality, if there are any. Which seasonality is more appropriate – additive or multiplicative? Explain the seasonal indices. In which month(s) do you see higher sales and which month(s) you see lower sales? Any difference in the nature of demand of the two items?

1. Can you extract the residuals for the two decomposition exercises and check if they form a stationary series? Do a formal test for stationarity writing down the null and alternative hypothesis. What is your conclusion in each case?
2. Before the final forecast is undertaken one would like to compare a few models. Use the last 21 months as hold-out sample fit a suitable exponential smoothing model to the rest of the data and calculate MAPE. What are the values of α, β and γ? What role do they play in the modeling? For the same hold-out period compare forecast by decomposition and compute MAPE. Which model gives smaller MAPE? Give a comparison for the two demands.
3. Use the 'best' model obtained from above to forecast demand for the period Oct 2017 to December 2018 for both items. Provide forecasted values as well as their upper and lower confidence limits. If you are the store manager what decisions would you make after looking at the demand of the two items over years?

2. Understanding the data

Time series data has several characteristics that make it unique. each observation is expected to depend on the past observations

Time series data is observed on the same variable over a given period of time with fixed and regular time intervals. Though data can be collected at various intervals such as yearly, monthly, weekly, daily, hour
Here in the data set we have provided monthly data for both Item A or Item B .

We would be using decomposition method to find out the Trend , seasonal variation and random movements, and regression method will be used for past observations
The data set consist of two variable calling Item A and Item B . Monthly data is collected till 2017 July and form 2002 Jan.
Time series include trend (***Tt***), cycles, seasonality (***St***) Irregularity (**I*t***) When choosing a forecasting method, we will first need to identify the time series patterns in the data, and then choose a method that is able to capture the patterns properly.

2.1 Approach

In the given dataset we have demand data for Item A and B for the period January 2002 to July 2017 (as against Sep 2017 mentioned in the problem statement). The data is continuous monthly data for the whole period without any breaks. This qualifies for a time series analysis on the demand for Item A & B, subject to other assumptions being valid.

The plan is to do the following:
1. Convert to time series for the 2 types of data
2. Perform Exploratory analysis of data by visual check of trend, seasonality
3. Time series decomposition into trend, seasonality and residuals
4. Check assumptions for stationary time series. In case of non-stationary, convert to stationary
5. Create various models of Forecasting the time series value
6. Interpretation of result

2.2 Importing data into R studio

```
> ##Importa the data file
> Consumable_data<- read_excel("C:/Users/SuprasannaPradhan/Documents/My Files/Great Lakes Projects/Demand.xlsx")
> str(Consumable_data)
Classes ‘tbl_df’, ‘tbl’ and 'data.frame':       187 obs. of  4 variables:
 $ Year  : num  2002 2002 2002 2002 2002 ...
 $ Month : num  1 2 3 4 5 6 7 8 9 10 ...
 $ Item A: num  1954 2302 3054 2414 2226 ...
 $ Item B: num  2585 3368 3210 3111 3756 ...
```

Demand.xlsx

```
> ts(Consumable_data)
Time Series:
Start = 1
End = 187
Frequency = 1
    Year Month Item A Item B
  1 2002     1   1954   2585
  2 2002     2   2302   3368
  3 2002     3   3054   3210
  4 2002     4   2414   3111
  5 2002     5   2226   3756
  6 2002     6   2725   4216
```

Importing the libraries

```
> library(readxl)# read excel files
> library(ggfortify)
> library(data.table)# For converting data into time series format
> library(ggplot2)# To plot various plots
> library(fpp2)# For examining seasonality graphically
> library(forecast)# For various functions related to Time serie
> library(stats)# For applying tests like acf, Ljung-Box Tests
> library(tseries)  # For applying Dickey Fuller test
>
```

2.3 Checking the pattern of the sale series:

```
> ##convert data into time seires format
> Consumable_ItemA<-ts(Consumable_data[,3],start = c(2002,1),frequency = 12 )
> Consumable_ItemB<-ts(Consumable_data[,4],start = c(2002,1),frequency = 12 )
> autoplot(ts(cbind(Consumable_ItemA,Consumable_ItemB),start = c(2002,1),frequency = 12 ),
+          facets = FALSE)
>
```

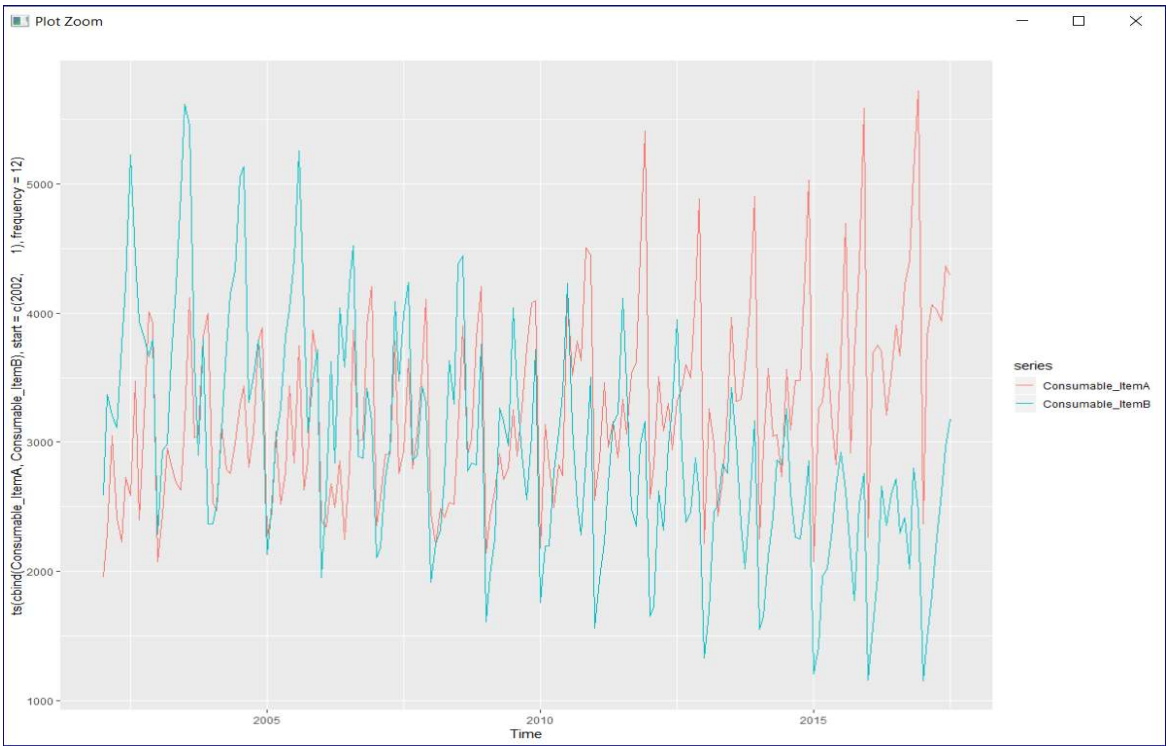

Figure 1:Comparison of Item A and Item B forecasting

We have converted whole data set with time series data and observations are plotted against the time of observation for Item A and Item B with consecutive observations together . Above shows monthly plot from January 2002 to July 2017 ,Item A is growing the sales over the period whereas Item B is leaning towards downcast .

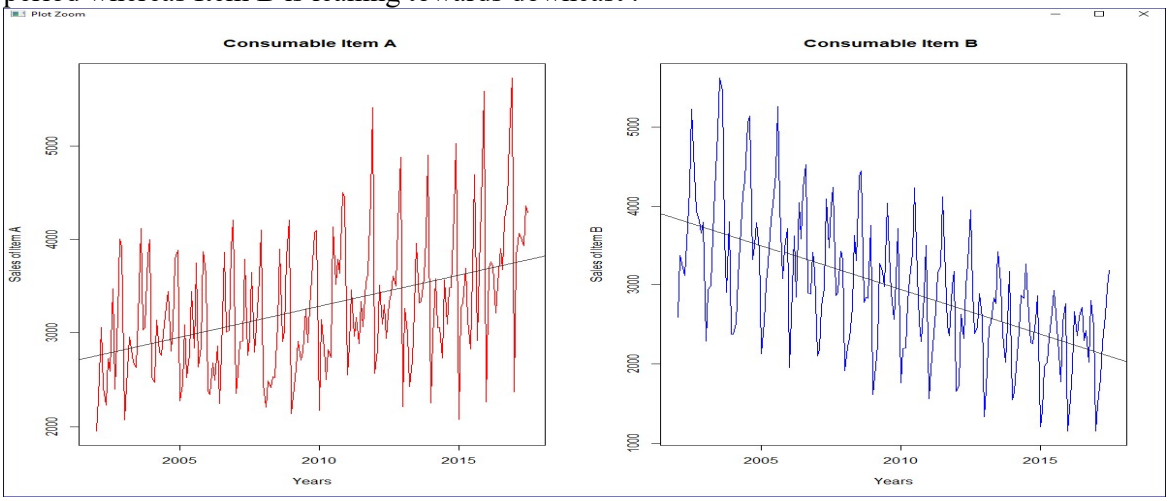

Figure 2: Increased or Decreased item wise yearly trend

```
> ## Plot the Time series data
> par(mfrow = c(1,2))
> plot(Consumable_ItemA, xlab="Years", ylab = "Sales of Item A",col="red", main = "Consumable Item A ")
> abline(reg=lm(Consumable_ItemA~time(Consumable_ItemA)))
> plot(Consumable_ItemB, xlab="Years", ylab = "Sales of Item B",col="blue",main = "Consumable Item B ")
> abline(reg=lm(Consumable_ItemB~time(Consumable_ItemB)))
>
```

Following are the observations obtained from Figure 2.

We have plotted both Item A and B separately to check their sale , found Item A has increased the sale after 2011 and picked up more in 2017 , Item B is decreased the sales constantly from 2006 to till 2017 .

- Data values are stored in correct time order and no data is missing.
- The sales are increasing in numbers, implying presence of trend component.
- Intra-year stable fluctuations are indicative of seasonal component. As trend increases, fluctuations are also increasing. This is indicative of multiplicative seasonality

Following three plots help to identify the seasonality fluctuations better.

```
> cycle(Consumable_ItemA)
     Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec
2002   1   2   3   4   5   6   7   8   9  10  11  12
2003   1   2   3   4   5   6   7   8   9  10  11  12
2004   1   2   3   4   5   6   7   8   9  10  11  12
2005   1   2   3   4   5   6   7   8   9  10  11  12
2006   1   2   3   4   5   6   7   8   9  10  11  12
2007   1   2   3   4   5   6   7   8   9  10  11  12
2008   1   2   3   4   5   6   7   8   9  10  11  12
2009   1   2   3   4   5   6   7   8   9  10  11  12
2010   1   2   3   4   5   6   7   8   9  10  11  12
2011   1   2   3   4   5   6   7   8   9  10  11  12
2012   1   2   3   4   5   6   7   8   9  10  11  12
2013   1   2   3   4   5   6   7   8   9  10  11  12
2014   1   2   3   4   5   6   7   8   9  10  11  12
2015   1   2   3   4   5   6   7   8   9  10  11  12
2016   1   2   3   4   5   6   7   8   9  10  11  12
2017   1   2   3   4   5   6   7
> cycle(Consumable_ItemB)
     Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec
2002   1   2   3   4   5   6   7   8   9  10  11  12
2003   1   2   3   4   5   6   7   8   9  10  11  12
2004   1   2   3   4   5   6   7   8   9  10  11  12
2005   1   2   3   4   5   6   7   8   9  10  11  12
2006   1   2   3   4   5   6   7   8   9  10  11  12
2007   1   2   3   4   5   6   7   8   9  10  11  12
2008   1   2   3   4   5   6   7   8   9  10  11  12
2009   1   2   3   4   5   6   7   8   9  10  11  12
2010   1   2   3   4   5   6   7   8   9  10  11  12
2011   1   2   3   4   5   6   7   8   9  10  11  12
2012   1   2   3   4   5   6   7   8   9  10  11  12
2013   1   2   3   4   5   6   7   8   9  10  11  12
2014   1   2   3   4   5   6   7   8   9  10  11  12
2015   1   2   3   4   5   6   7   8   9  10  11  12
2016   1   2   3   4   5   6   7   8   9  10  11  12
2017   1   2   3   4   5   6   7
>
> par(mfrow = c(2,2))
> plot(aggregate(Consumable_ItemA,FUN=mean))
> plot(aggregate(Consumable_ItemB,FUN=mean))
> boxplot(Consumable_ItemA~cycle(Consumable_ItemA))
> boxplot(Consumable_ItemB~cycle(Consumable_ItemB))
>
```

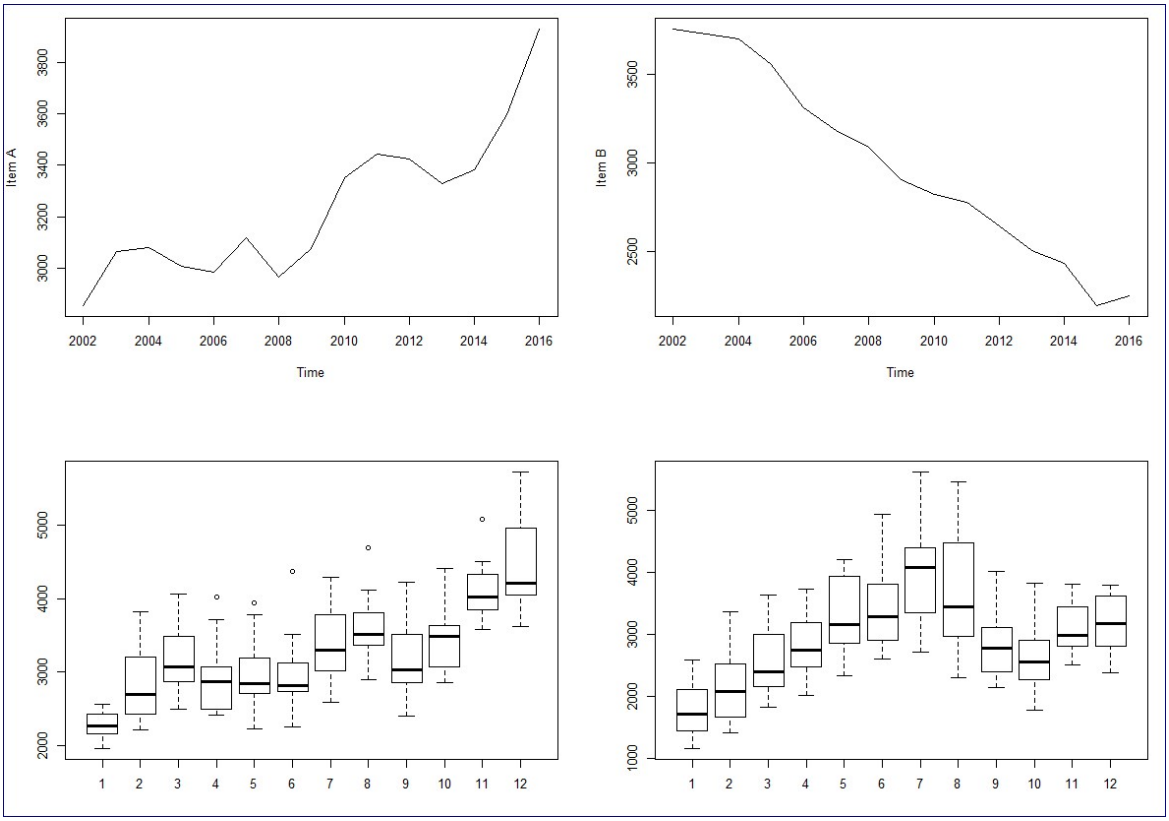

Figure 3:Box plot and aggregate

Important Inferences

The year on year trend clearly shows that the Item A have been increasing without fail.

From the above plots, we can see Item A has an increasing demand, whereas Item B has fall in demand. Also, there is some seasonality and trend in demands. Both Item A and B doesn't seem to have cyclic in nature. Item A variation increases with time whereas Item B variation decreases.

The variance and the mean value item A is in Dec and item B is in July much higher than rest of the months.

Even though the mean value of each month is quite different their variance is small. Hence, we have strong seasonal effect with a cycle of 12 months or less.

```
> ## Plot 1: Seasonal plot Year-wise
> ggseasonplot(Consumable_ItemA, year.labels=TRUE, year.labels.left=TRUE ) + ylab("degree") +
+             ggtitle("Seasonal plot: Sales Data of Item A")
> ggseasonplot(Consumable_ItemB, year.labels=TRUE, year.labels.left=TRUE ) + ylab("degree") +
+             ggtitle("Seasonal plot: Sales Data of Item B")
>
```

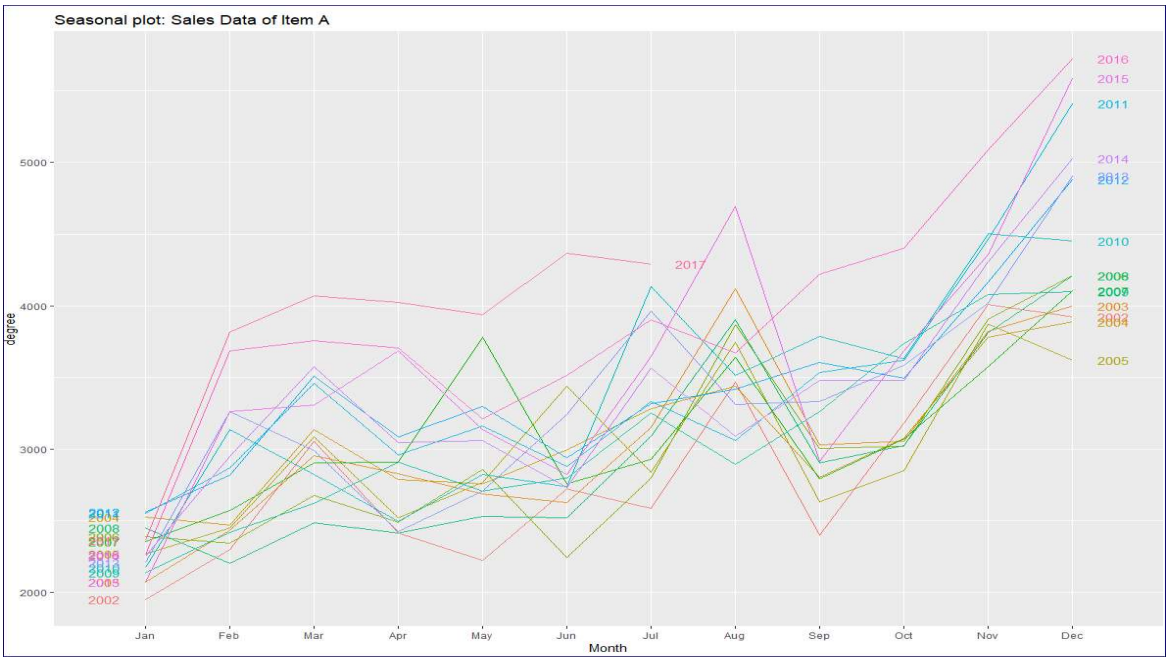

Figure :4 Seasonal plot year-wise Item A

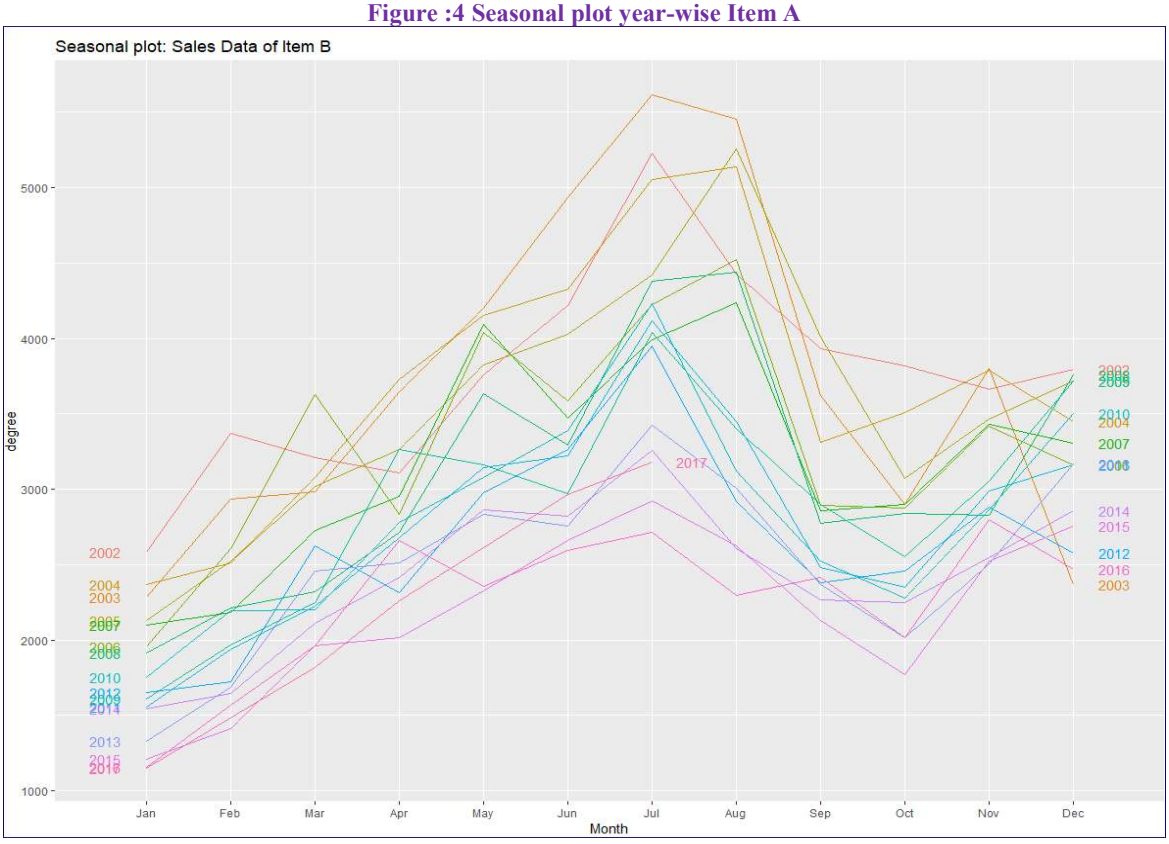

Figure :5 Seasonal plot year-wise Item B

```
> ## Plot 2: Polar Seasonal plot Year-wise
> ggseasonplot(Consumable_ItemA, polar=TRUE) +   ylab("degree") +
+            ggtitle("Polar seasonal plot: Sales Data A")
>
> ggseasonplot(Consumable_ItemB, polar=TRUE) +   ylab("degree") +
+             ggtitle("Polar seasonal plot: Sales Data B")
>
```

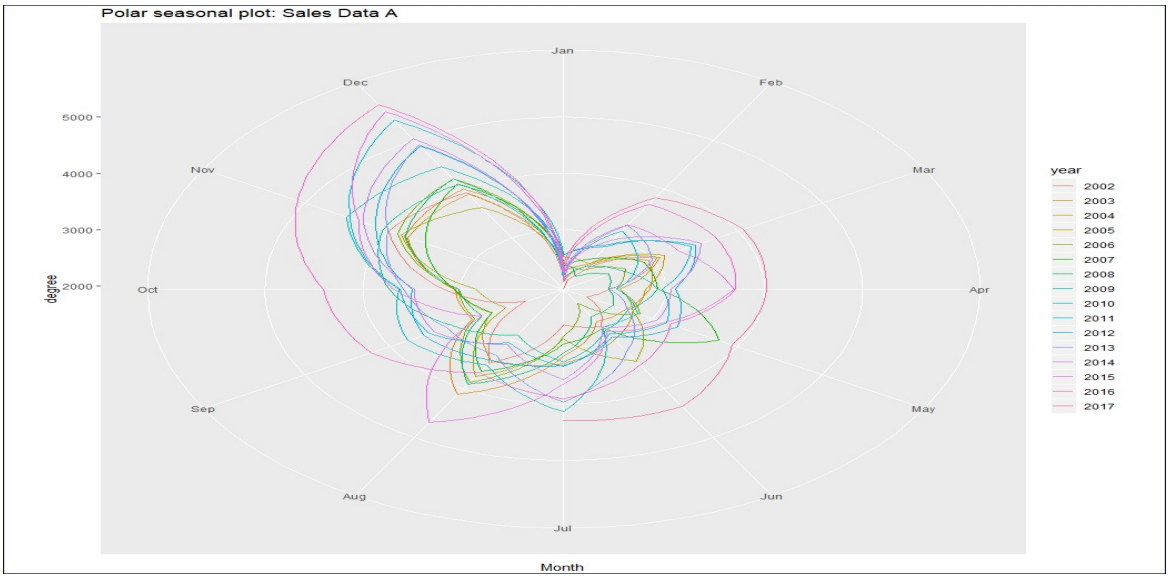

Figure 6: Seasonal plot year-month-wise Item A

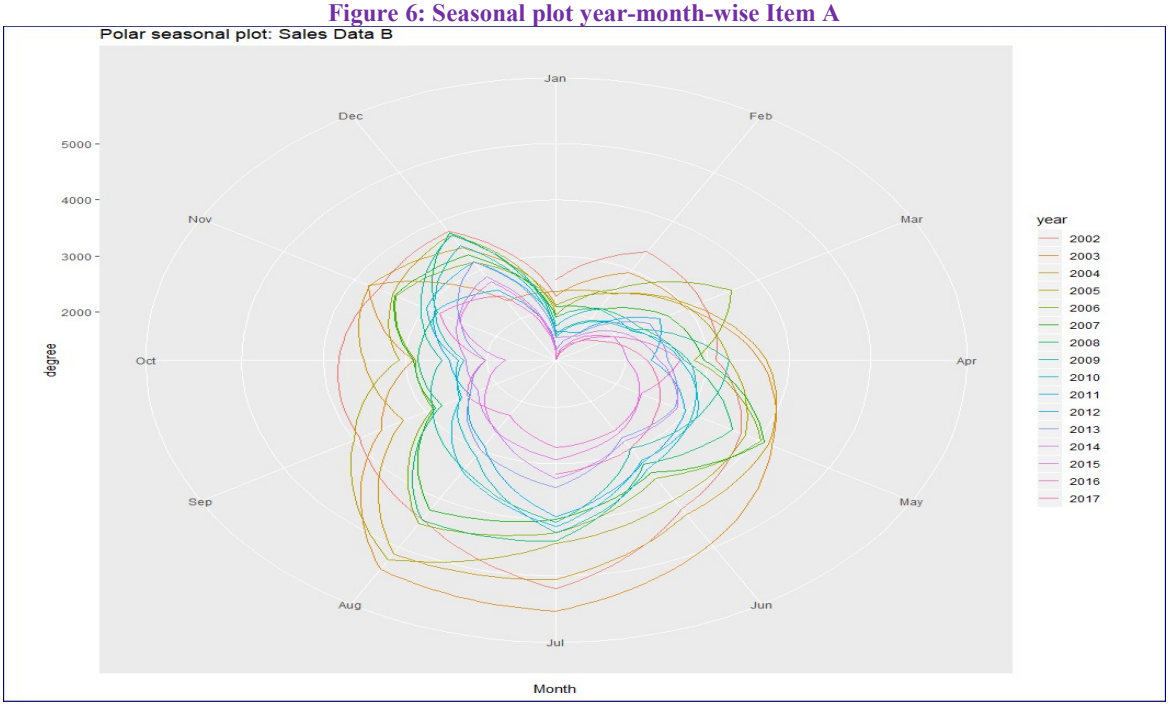

Figure 7: Seasonal plot year-month-wise Item B

```
> ## Seasonal plot Month-wise
> monthplot(Consumable_ItemA)
> monthplot(Consumable_ItemB)
> par(mfrow = c(1,2))
> monthplot(Consumable_ItemA)
> monthplot(Consumable_ItemB)
>
```

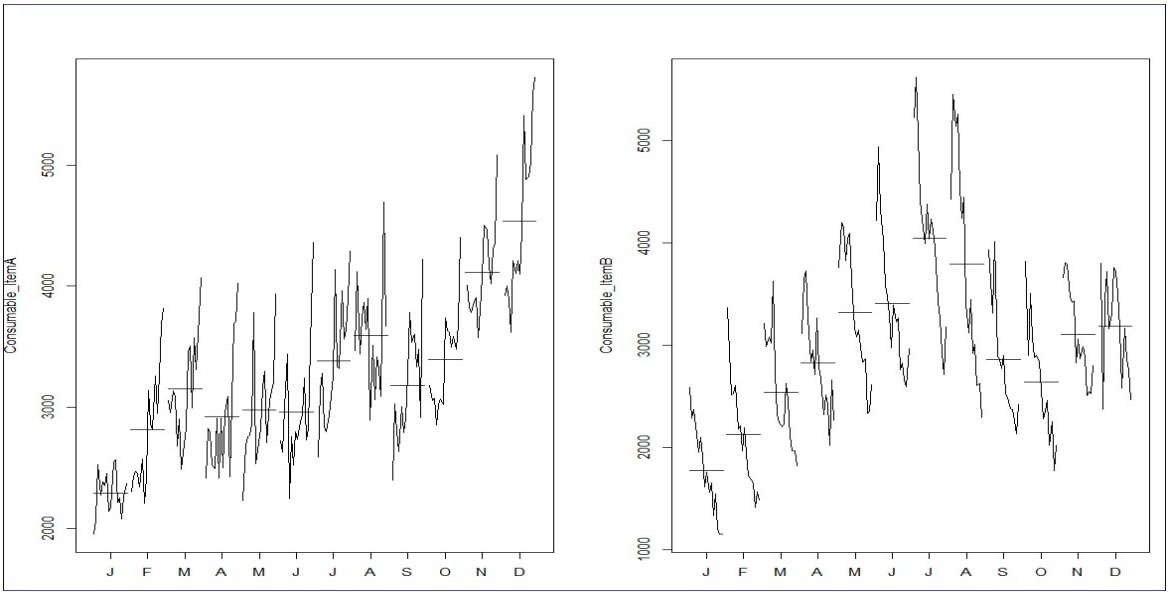

Figure 8: Seasonal plot month-wise

Figure 4 and Figure 5 graphs show that the sales of item A are increasing every year in number.

Figure 8 emphasizes that; the vertical lines represent monthly sales and the horizontal lines represent average sales of the given month. Here, it can be observed that average sale are higher in December for Item A and August for Item B as compered other months

In all these above plots the increasing lines that represent sales have seasonal fluctuations along with a trend. Thus, we can confirm that it is multiplicative seasonality.

3. Identify the components of the given sales data

Now decomposition method is applied to identify and separate out the three components (i.e. trend, seasonality and irregular components) from the given series to observe their independent properties

A time series decomposition is procedure which transform a time series into multiple different series. The original time series is often computed (decompose) into 3 sub-time series:

1. **Seasonal:** patterns that repeat with fixed period of time.
2. **Trend:** the underlying trend of the metrics.
3. **Random:** (also call "noise", "Irregular" or "Remainder") Is the residuals of the time series after allocation into the seasonal and trends time series.

Other than above three component there is **Cyclic** component which occurs after long period of time.

Additive or multiplicative decomposition? To get a successful decomposition, it is important to choose between the **additive** or **multiplicative** model. To choose the right model we need to look at the time series.

a. The **additive model** is useful when the seasonal variation is relatively constant over time.
b. The **multiplicative model** is useful when the seasonal variation increases over time.

3.1 Decomposition of TS using decompose()

```
> ItemA<-decompose(Consumable_ItemA, type = "multiplicative")
> plot(ItemA)
>
```

```
> ItemB<-decompose(Consumable_ItemB, type = "multiplicative")
> plot(ItemB)
>
```

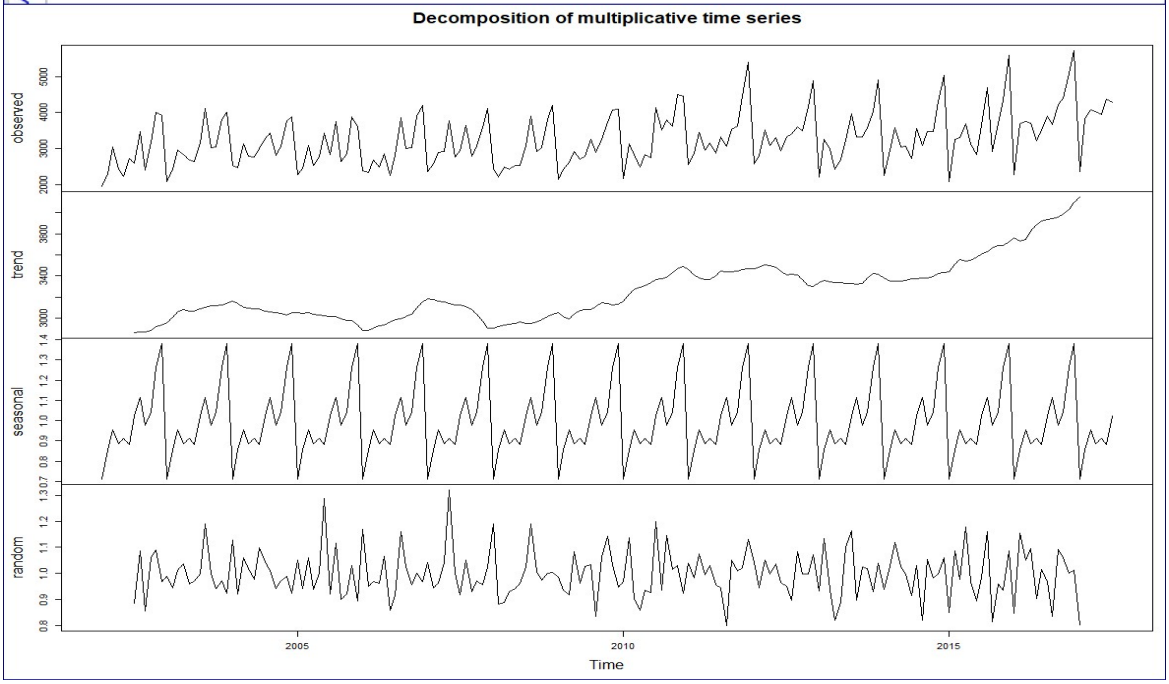

Figure 9: Decomposed Item A time series into components using decompose()

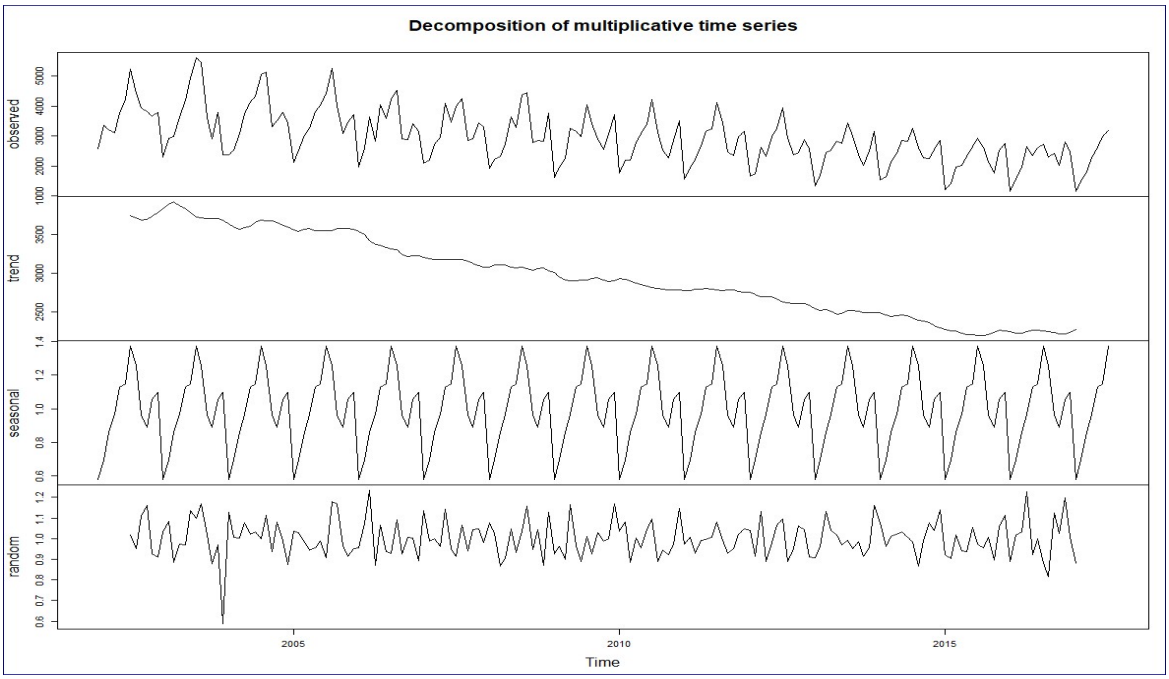

Figure 10: Decomposed Item B time series into components using decompose()

```
> ## Observing the seasonal indices of "TSDecmpose"
> Seasonal_A=round(t(ItemA$figure),3)
> colnames(Seasonal_A) <-c("Jan","Feb","Mar","Apr","May","Jun","Jul", "Aug","Sep","Oct","Nov","Dec")
> Seasonal_A
      Jan   Feb   Mar   Apr   May   Jun   Jul   Aug   Sep   Oct   Nov  Dec
[1,] 0.71 0.856 0.954 0.885 0.914 0.884 1.024 1.114 0.976 1.041 1.262 1.38
>
```

```
> Seasonal_B=round(t(ItemB$figure),4)
> colnames(Seasonal_B) <-c("Jan","Feb","Mar","Apr","May","Jun","Jul", "Aug","Sep","Oct","Nov","Dec")
> Seasonal_B
        Jan    Feb    Mar    Apr    May    Jun    Jul    Aug    Sep   Oct    Nov    Dec
[1,] 0.5777 0.694 0.8586 0.9659 1.1276 1.1476 1.3715 1.2541 0.9603 0.888 1.0575 1.0974
>
```

Figure 9 Item A indicates that trend is increasing linearly . Since this is monthly data, there are 12 seasonal indices. Sum of the monthly indices is 12. In December sales is the highest among all months in the same year, as borne by the highest value of the seasonal component whereas in January (lowest seasonality) sales is the lowest

Figure 10 Item B similarity indicates for sum of 12 months ,June is carried the highest sale value and lowest seasonality found in January

3.2 Decomposition of TS Using stl()

STL is an acronym for "Seasonal and Trend decomposition using Loess", while Loess is a method for estimating nonlinear relationships. Owing to some limitations in decompose() function, stl() has been proposed. This is more versatile but does not admit multiplicative seasonality. Hence log transformation is used to convert multiplicative seasonality into additive seasonality

```
> ## Decomposition of TS
> ItemA_se <-stl(log10(Consumable_ItemA[,1]), s.window='p')
> ItemB_se <-stl(log10(Consumable_ItemB[,1]), s.window='p')
> plot(ItemA_se,col='orange')
> plot(ItemB_se,col='blue')
>
```

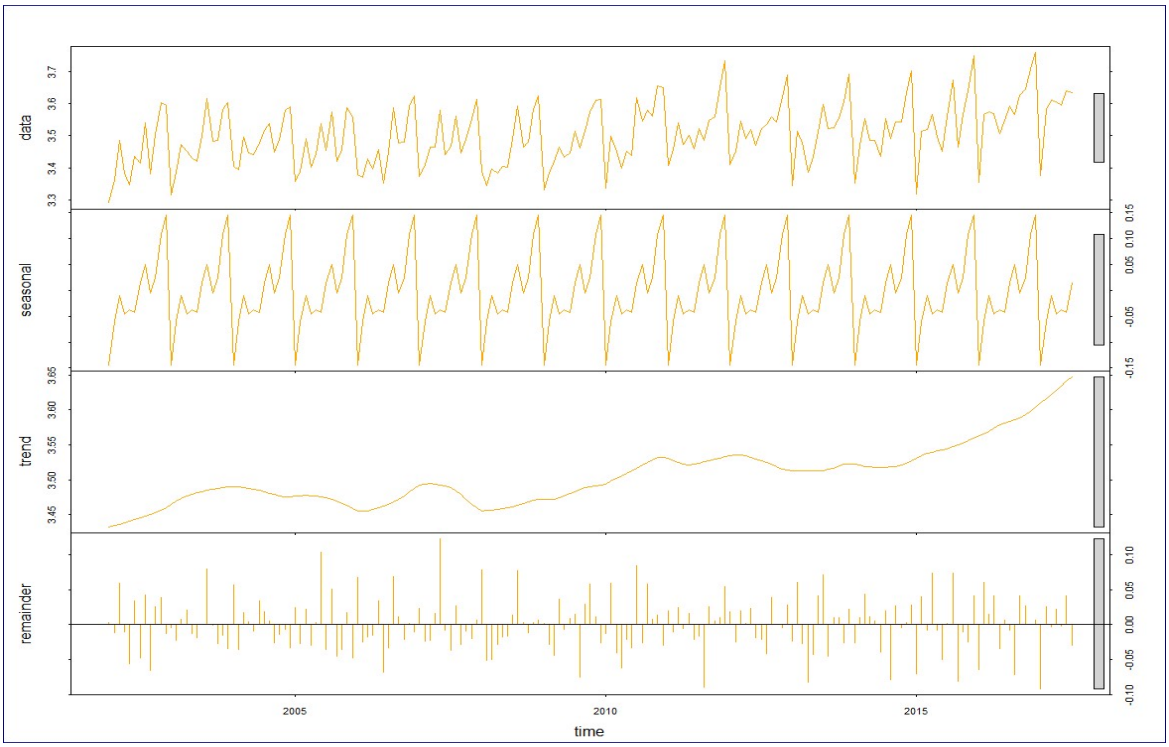

Figure 10: Decomposed time series into components using stl()-Item A

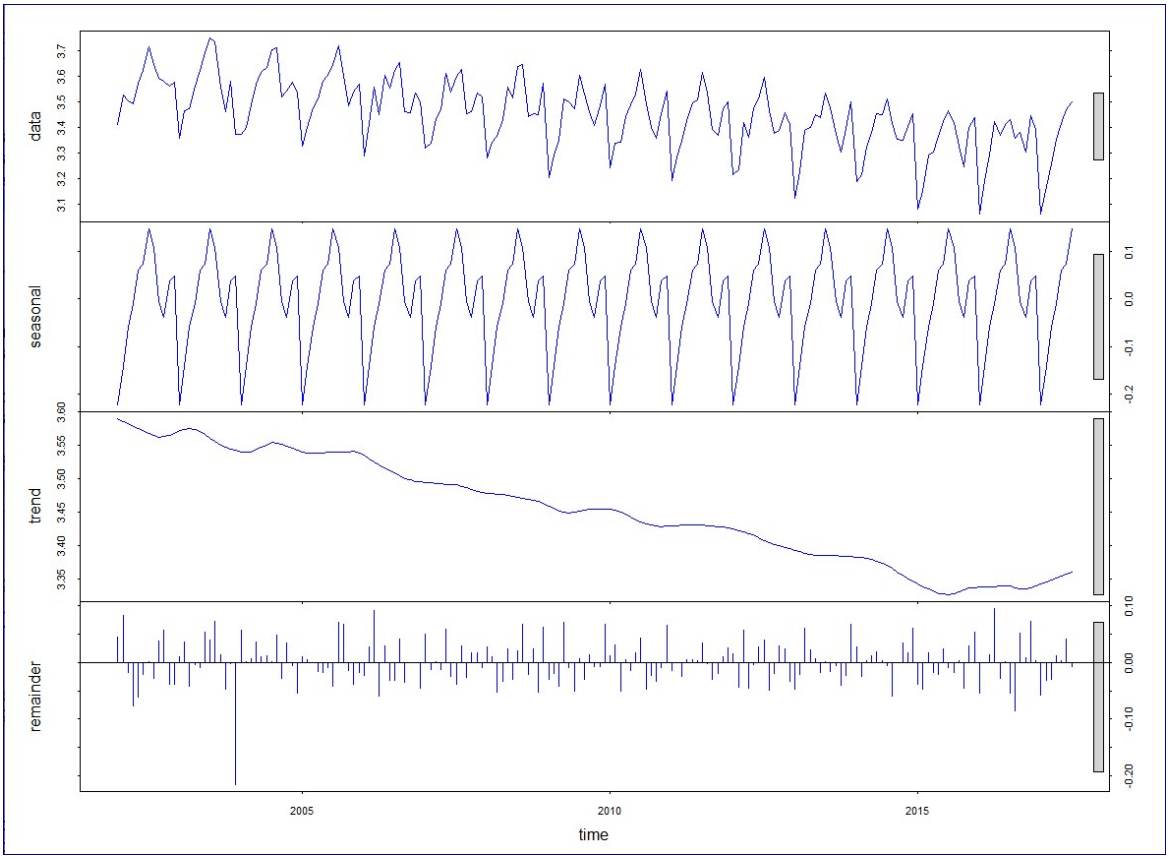

Figure 10: Decomposed time series into components using stl()- Item B

From the above decomposed details, we can see that there is continuous increase in demand for Item A, but on contrary similar drop pattern observed for Item B.

Decompose the time series and plot the deseasoned series
If the focus is on figuring out whether the general trend of demand is up, we depersonalize, and possibly forget about the seasonal component. However, if we need to forecast the demand in next month, then we need take into account both the secular trend and seasonality.

In Figure 9 and 10 the small bars , represent the magnitude of the particular component in the given series. Smaller the bar; greater the significance of the component.

3.3 Data split

```
> ##Spliting data into training and test data sets
> ## Spliting data set of Item A
> ItemA_Train <- window(Consumable_ItemA, start=c(2002,1), end=c(2015,10), freq=12)
> ItemA_Test <- window(Consumable_ItemA, start=c(2015,11),end=c(2017,7),freq=12)
> autoplot(ts(cbind(ItemA_Train,ItemA_Test),start = c(2002,1),frequency = 12 ),
+          facets = FALSE)
>
```

```
##Spliting data set for Item B
ItemB_Train <- window(Consumable_ItemB, start=c(2002,1), end=c(2015,10), freq=12)
ItemB_Test <- window(Consumable_ItemB, start=c(2015,11),freq=12)
autoplot(ts(cbind(ItemB_Train,ItemB_Test),start = c(2002,1),frequency = 12 ),
         facets = FALSE)
```

Before a forecast method is proposed, the method needs to be validated. For that purpose, data has to be split into two sets i.e. training and testing. Training data helps in identifying and fitting right model(s) and test data is used to validate the same.
In case of time series data, the test data is the most recent part of the series so that the ordering in the data is preserved.

Propose best model for the sales data
It is important to evaluate forecast accuracy using genuine forecasts. That is, it is invalid to look at how well a model fits the historical data; the accuracy of forecasts can only be determined by considering how well a model performs on new data that were not used when estimating the model .

3.4 Measures of Forecast Accuracy

Forecasting accuracy measures compare the predicted values against the observed values to quantify the predictive power of the proposed model. Mathematically, it can be defined as
Forecast error et for period t is given by: $et = Y\ t\hat{}-Y\ t$
Where $Y\ t\hat{}$= forecast value for time period
t $Y\ t$ = actual value in time period
$t\ n$ = No. of observations

Major Various measures of forecasting
Mean Absolute Deviation (MAD): $MAD = \sum |et|/n\ n\ t$=1
Mean Absolute Percentage Error (MAPE):

This method is used extensively in time series because it acts as a unit free criteria; therefore, performance of forecasted values can be easily compared. $MAPE = \sum (|et|/Y\ t)/nn\ t=1\ *100$. MAPE is usually expressed as a percentage.

Mean Square Error (MSE): $MSE = \sum |et|/n\ n\ t=1$

Root Mean Square Error (RMSE): $RMSE = \sqrt{\sum}\ (et)2/n\ n\ t=1$

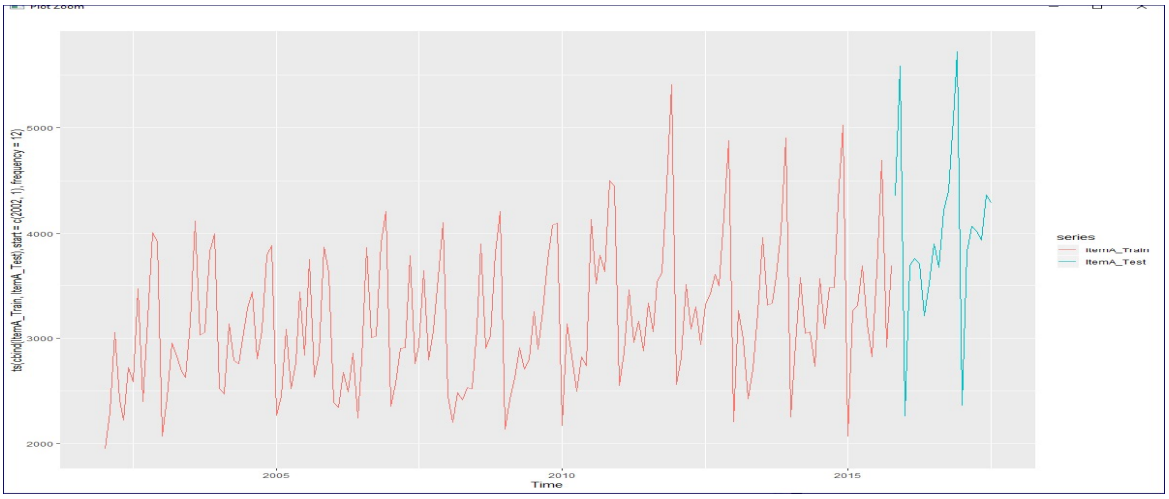

Figure 11: Data (split into Training and test for Item A

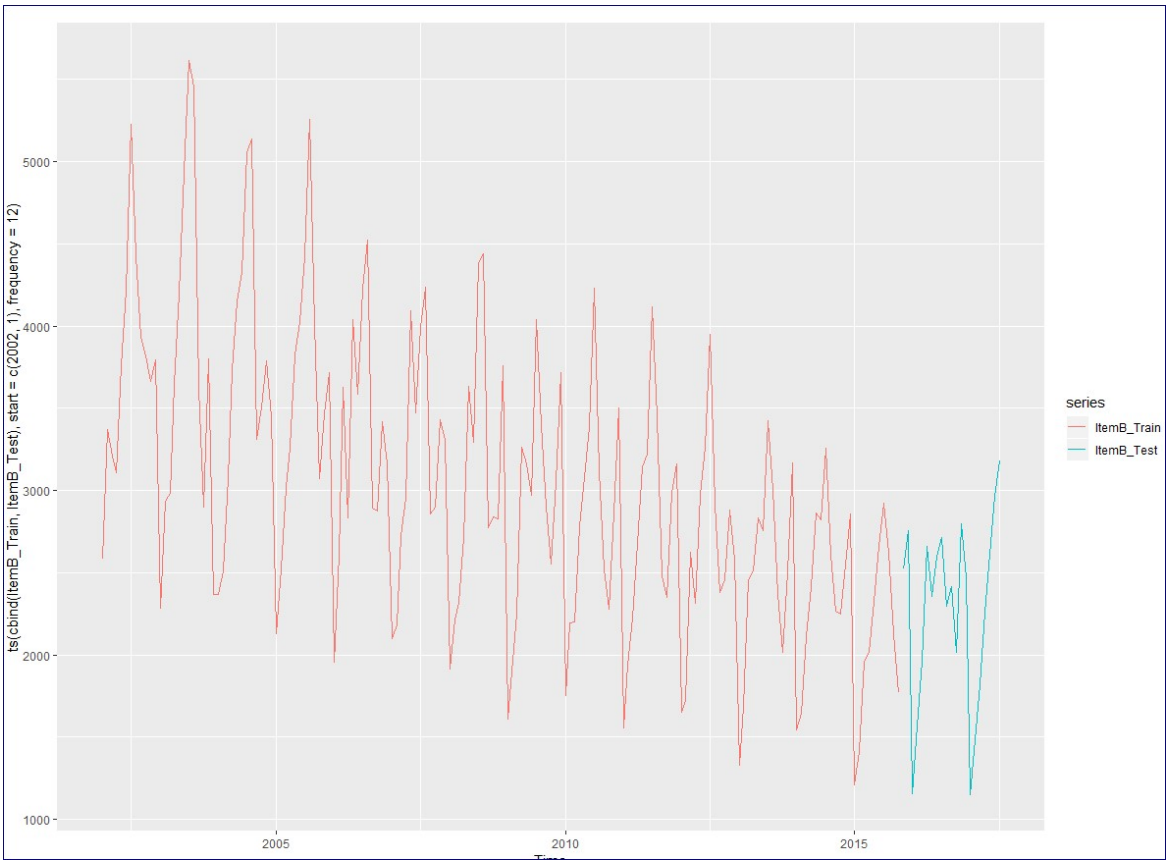

Figure 12: Data (split into Training and test for Item B

4. Forecasting Methods

4.1 Random walk with drift

This method forecasts next period value as per the amount of change over time (called the drift) is evaluated the average change seen in past data.

This method can be applied along with decomposition.

```
~/My Files/R/R project files/
> ##Forecasting Methods
> ##Forcasting Item A  for 21 months
> ItemA_Train_log <- log(ItemA_Train)
> library(forecast)
> ItemADecmpose_Train_Log <-stl(ItemA_Train[,1],s.window="period")
> ItemA_Train_stl<-forecast(ItemADecmpose_Train_Log,method="rwdrift",h=21)
> plot(ItemA_Train_stl)
>
```

```
> ##Forecasting Methods
> ##Forcasting Item B  for 21 months
> ItemB_Train_log <- log(ItemB_Train[,1])
> library(forecast)
> ItemBDecmpose_Train_Log <-stl(ItemB_Train[,1],s.window="period")
> ItemB_Train_stl<-forecast(ItemBDecmpose_Train_Log,method="rwdrift",h=21)
> plot(ItemB_Train_stl)
>
```

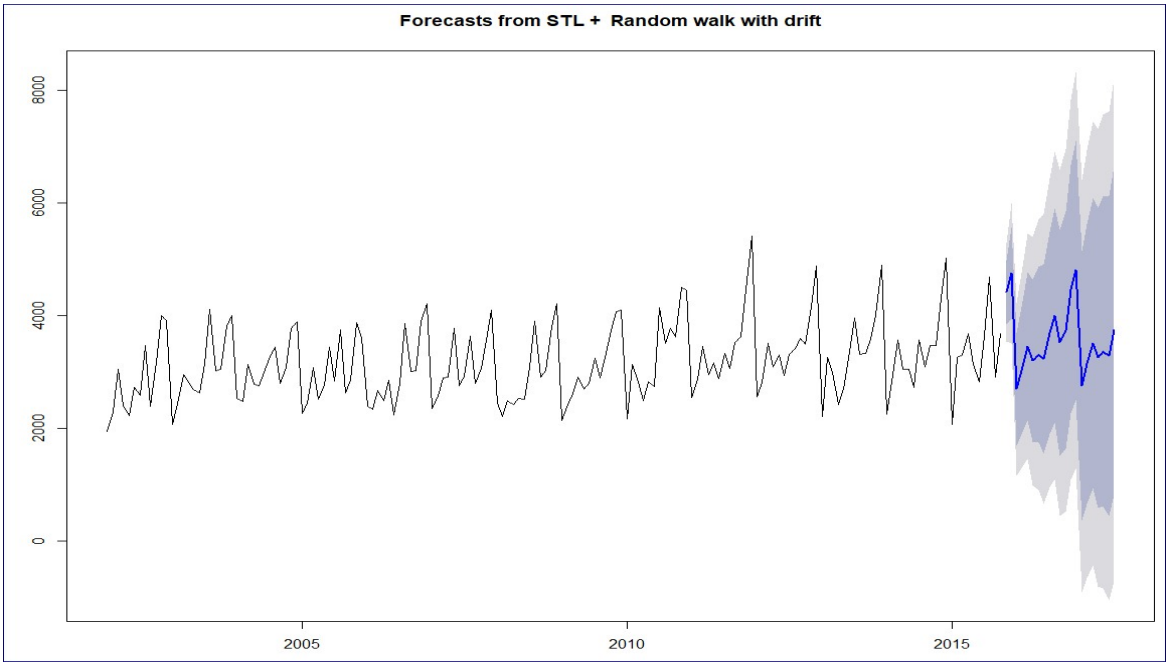

Figure 13: Forecasted 21 months using RWD method for Item A

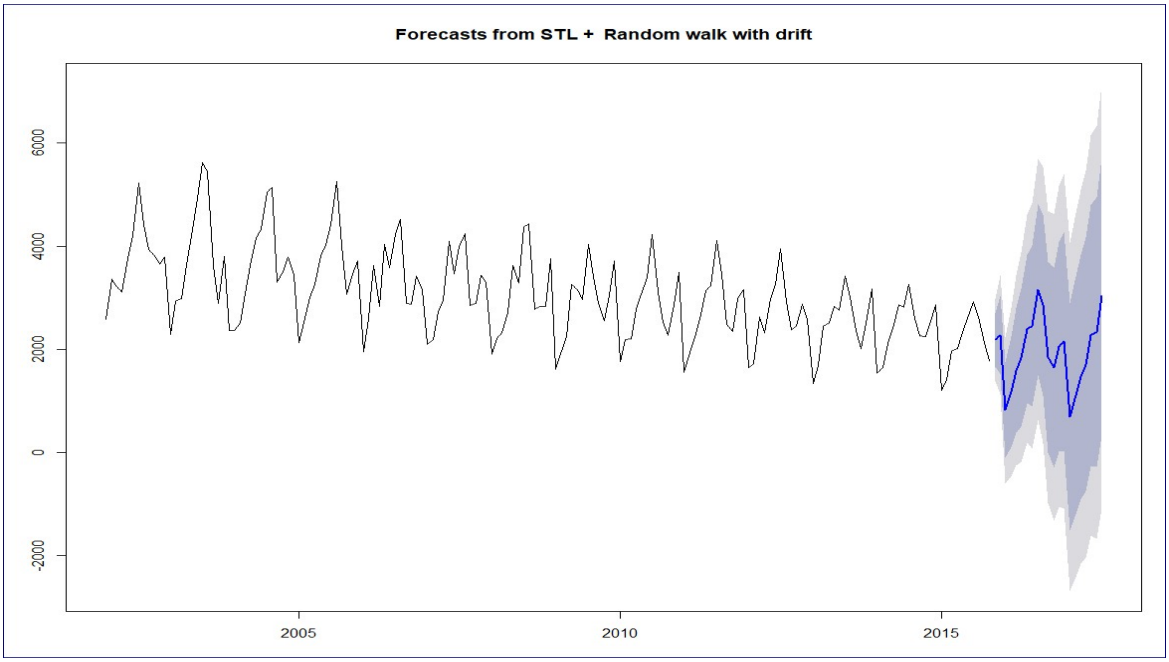

Figure 13: Forecasted 21 months using RWD method for Item B

4.2 Accuracy measures using RWD method

```
> ## Accuracy measures: RMSE and MAPE using RWD Item A
> ItemAVec2 <-cbind(ItemA_Test,ItemA_Train_stl$mean)
> ts.plot(ItemAVec2 , col=c("blue", "red"), main="Item A sales: Actual vs F orecast")
```

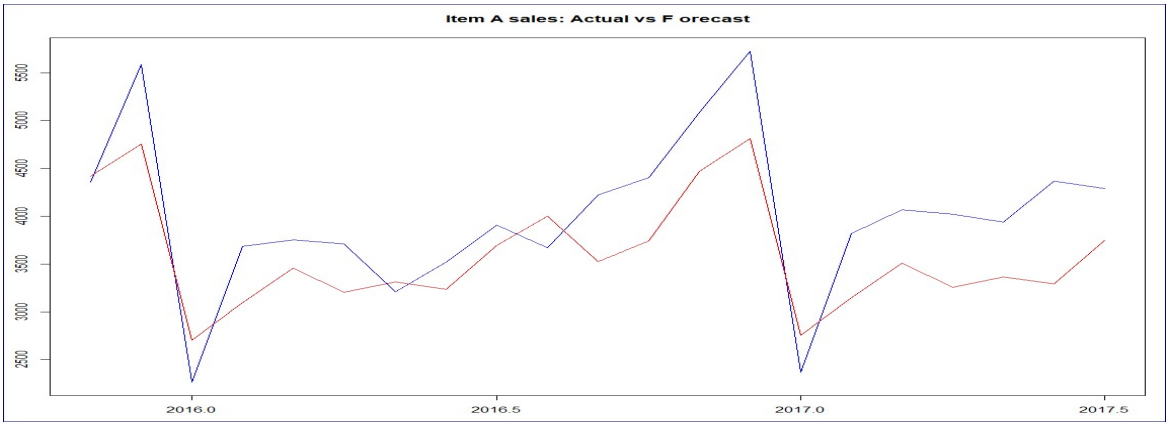

Figure 14: Plot Actual vs. Forecasted sales using RWD's method for 2016-2017 years

```
> ## Accuracy measures: RMSE and MAPE using RWD Item B
> ItemBVec2 <-cbind(ItemB_Test,ItemB_Train_stl$mean)
> ts.plot(ItemBVec2 , col=c("blue", "red"), main="Item B sales: Actual vs F orecast")
```

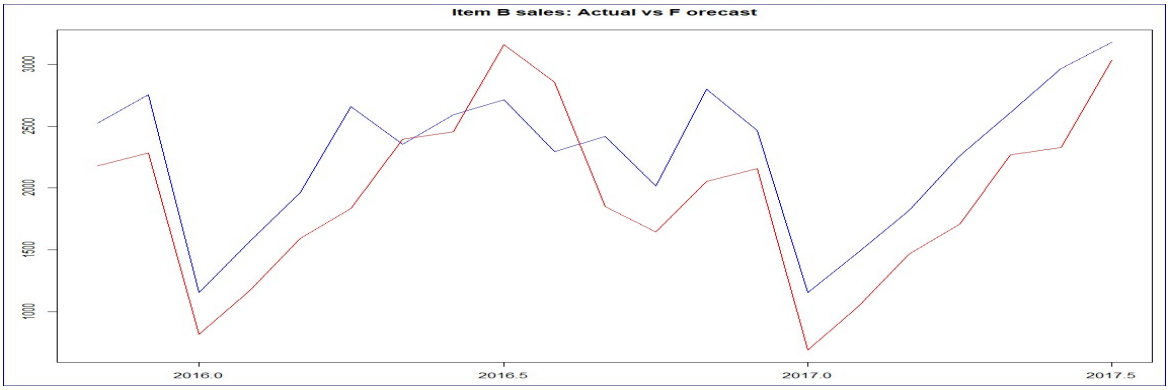

Figure 15: Plot Actual vs. Forecasted sales using RWD's method for 2016-2017 years

```
> RMSEA2 <- round(sqrt(sum(((ItemAVec2[,1]-ItemAVec2[,2])^2)/length(ItemAVec2[,1]))),4)
> MAPEA2 <- round(mean(abs(ItemAVec2[,1]-ItemAVec2[,2])/ItemAVec2[,1]),4)
> paste("Accuracy Measures: RMSE:", RMSEA2, "and MAPE:", MAPEA2)
[1] "Accuracy Measures: RMSE: 587.2886 and MAPE: 0.1318"
```

```
> RMSEB2 <- round(sqrt(sum(((ItemBVec2[,1]-ItemBVec2[,2])^2)/length(ItemBVec2[,1]))),4)
> MAPEB2 <- round(mean(abs(ItemBVec2[,1]-ItemBVec2[,2])/ItemBVec2[,1]),4)
> paste("Accuracy Measures: RMSE:", RMSEB2, "and MAPE:", MAPEB2)
[1] "Accuracy Measures: RMSE: 460.9989 and MAPE: 0.1987"
```

4.3 Exponential Smoothing Method:

This is an extension of moving (rolling) average method where more recent observations get higher weight.

We would be using here Holt Winter's method (Triple Exponential): This is an extension of Holt's method when seasonality is found in the data.

Forecast equation : $\hat{Y}t+1 = lt + bt + st-m(k+1)$
Level Equation : $lt = \alpha(Y\ t - st-m)+\alpha(1-\alpha)Y\ t-1,\ 0 < \alpha < 1$
Trend Equation : $bt = \beta(lt - lt-1)+(1-\beta)bt-1,\ \ 0 < \beta < 1$
Seasonal Equation : $\gamma(Y\ t - lt-1 - bt-1)+(1-\gamma)st-m,\ \ 0 < \gamma < 1$

This is also known as three parameters exponential or triple exponential because of the three smoothing parameters α, β and γ. This is a general method and a true multi-step ahead forecast.

sales series is to be forecasted using Holt-Winters' method since it contains both trend and seasonality.

```
> ##Exponential Smoothing Method
> #Apply Holt winter method Item A
> ItemA_Train_H <- window(Consumable_ItemA, start=c(2002,1), end=c(2015,7), freq=12)
> ItemA_Test_H <- window(Consumable_ItemA, start=c(2015,8),end=c(2017,7),freq=12)
>
> ItemB_Train_H <- window(Consumable_ItemB, start=c(2002,1), end=c(2015,7), freq=12)
> ItemB_Test_H <- window(Consumable_ItemB, start=c(2015,8),end=c(2017,7),freq=12)
>
> ItemA_Train_HW <- hw(ItemA_Train_H, seasonal = "multiplicative")
> plot(ItemA_Train_HW)
```

```
> ItemA_Train_HW$model
Holt-Winters' multiplicative method

Call:
 hw(y = ItemA_Train_H, seasonal = "multiplicative")

  Smoothing parameters:
    alpha = 0.0992
    beta  = 0.0021
    gamma = 0.2864

  Initial states:
    l = 2981.6059
    b = 15.06
    s = 1.3104 1.2791 1.0097 0.9291 1.188 0.9782
           0.9416 0.8968 0.8989 1.005 0.8125 0.7507

  sigma:  0.1048

     AIC     AICc      BIC
2736.815 2741.036 2789.409
```

values of α,β and γ, and can observe the differences in the model

```
> #Forecast data for next 21 months using HW method for Item A
> plot(forecast(ItemA_Train_HW, h=21))
>
```

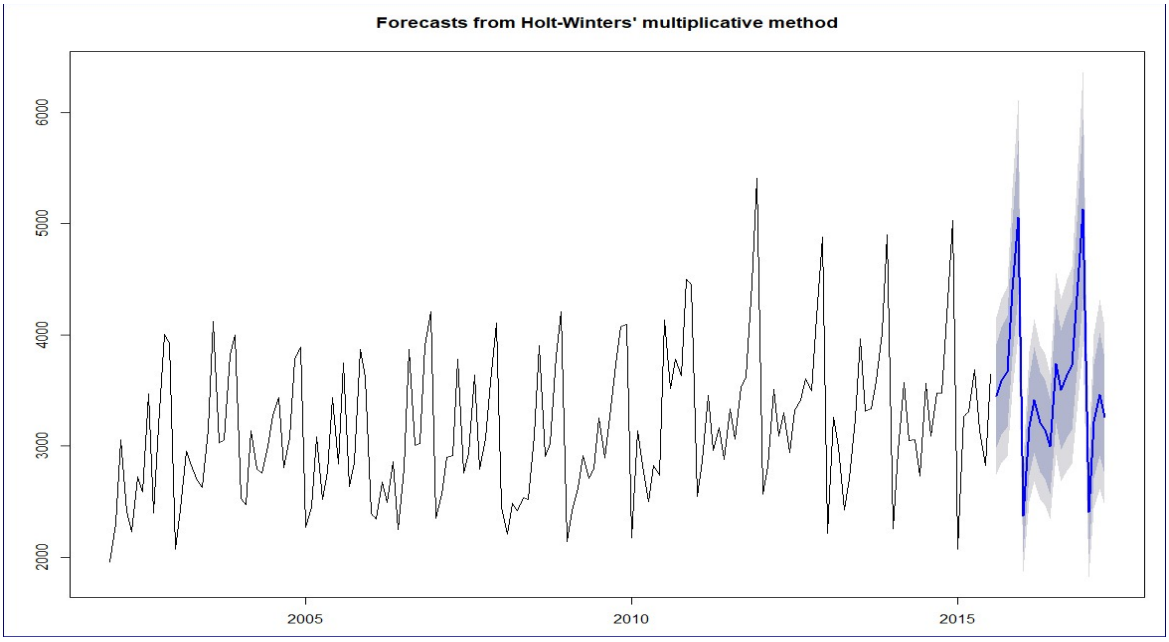

Figure 16: Forecasted sales using HW's method for Item A

```
> #Apply Holt winter method Item B
> ItemB_Train_HW <- hw(ItemB_Train_H, seasonal = "multiplicative")
> plot(ItemB_Train_HW)
> ItemB_Train_HW$model
Holt-Winters' multiplicative method

Call:
 hw(y = ItemB_Train_H, seasonal = "multiplicative")

  Smoothing parameters:
    alpha = 0.0223
    beta  = 0.0015
    gamma = 1e-04

  Initial states:
    l = 4056.6419
    b = -12.5949
    s = 1.0572 1.034 0.8952 0.9635 1.3073 1.3861
           1.148 1.1181 0.9316 0.8521 0.7187 0.5883

  sigma:  0.0991

     AIC     AICc      BIC
2697.148 2701.368 2749.741
```

```
>
> ##Forecast data for next 21 months using HW method for Item B
> plot(forecast(ItemB_Train_HW, h=21))
>
```

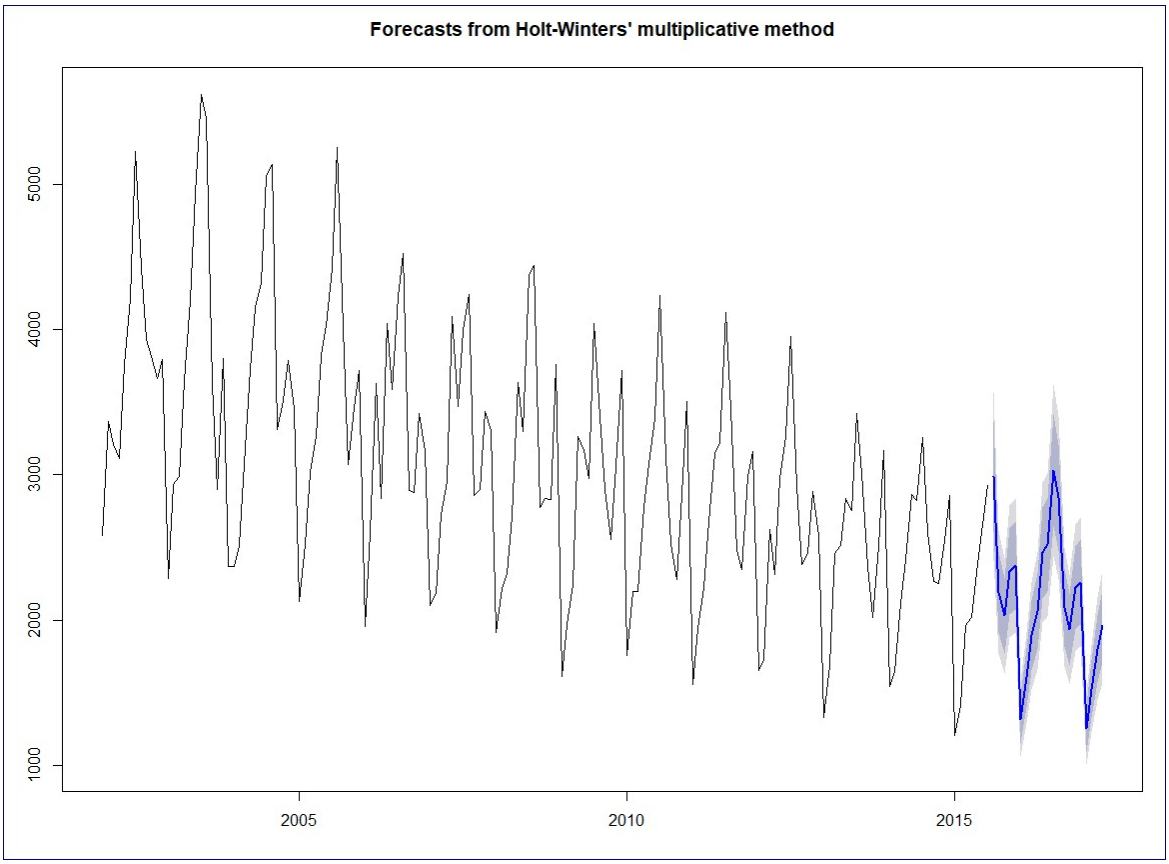

Figure 17: Forecasted sales using HW's method for Item B

4.4 Accuracy value using Holt-Winter method

```
> ## Accuracy measures: RMSE and MAPE using HW Item A
> ItemAVec <-cbind(ItemA_Test_H,ItemA_Train_HW$mean)
> ts.plot(ItemAVec , col=c("blue", "red"), main="Item A sales: Actual vs F orecast")
> RMSEA <- round(sqrt(sum(((ItemAVec[,1]-ItemAVec[,2])^2)/length(ItemAVec[,1]))),4)
> MAPEA <- round(mean(abs(ItemAVec[,1]-ItemAVec[,2])/ItemAVec[,1]),4)
> paste("Accuracy Measures: RMSE:", RMSEA, "and MAPE:", MAPEA)
[1] "Accuracy Measures: RMSE: 599.1511 and MAPE: 0.1209"
>
```

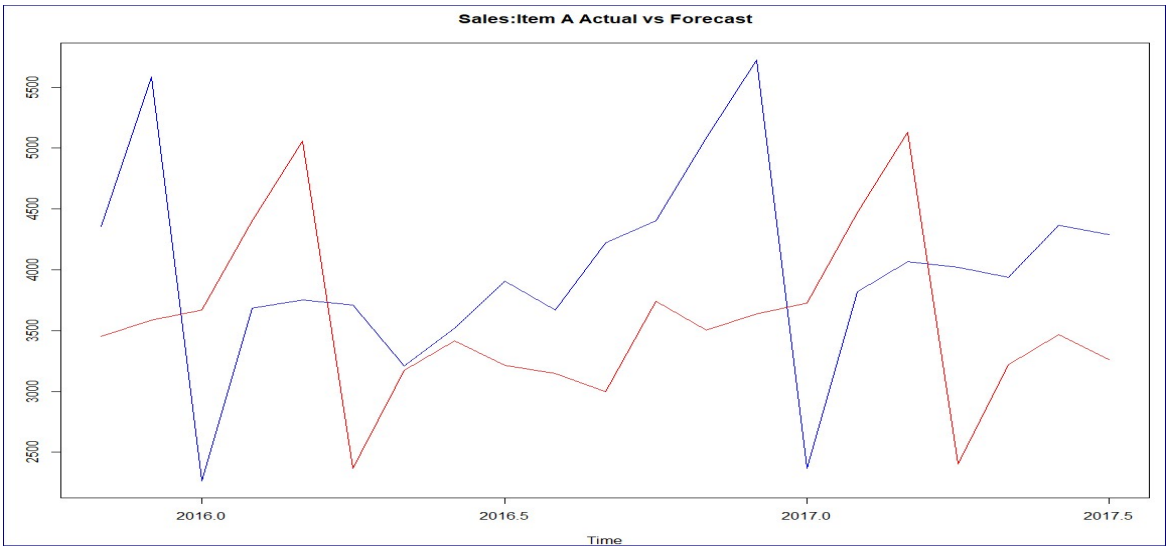

Figure 18: Plot Actual vs. Forecasted sales using HW's method for 2016-2017 years for Item A

```
> ## Accuracy measures: RMSE and MAPE using HW Item B
> ItemBVec<- cbind(ItemB_Test ,as.data.frame(forecast(ItemB_Train_HW,h=21))[,1])
> ts.plot(ItemBVec, col=c("blue", "red"), main="Sales:  Item B Actual vs Forecast")
```

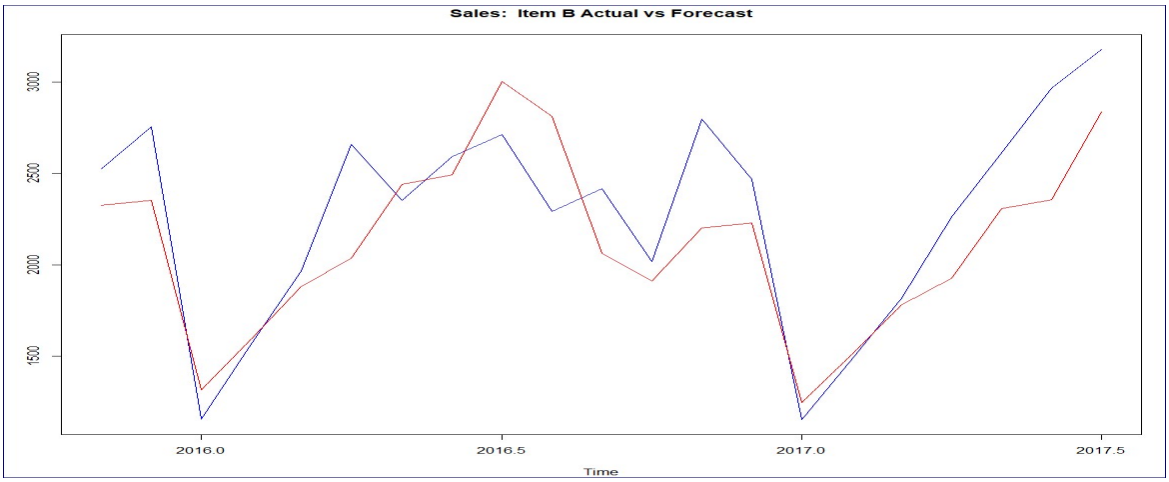

Figure 18: Plot Actual vs. Forecasted sales using HW's method for 2016-2017 years for Item B

As before blue line shows actual observations and red shows forecasted values. The latter is still below the actual values.

```
> RMSEA <- round(sqrt(sum(((ItemAVec[,1]-ItemAVec[,2])^2)/length(ItemAVec[,1]))),4)
> MAPEA <- round(mean(abs(ItemAVec[,1]-ItemAVec[,2])/ItemAVec[,1]),4)
> paste("Accuracy Measures: RMSE:", RMSEA, "and MAPE:", MAPEA)
[1] "Accuracy Measures: RMSE: 1170.6974 and MAPE: 0.2668"
> RMSEB <- round(sqrt(sum(((ItemBVec[,1]-ItemBVec[,2])^2)/length(ItemBVec[,1]))),4)
> MAPEB <- round(mean(abs(ItemBVec[,1]-ItemBVec[,2])/ItemBVec[,1]),4)
> paste("Accuracy Measures: RMSE:", RMSEB, "and MAPE:", MAPEB)
[1] "Accuracy Measures: RMSE: 326.803 and MAPE: 0.1081"
>
```

5. Stationarization of a non-stationary series

Since ARIMA model requires a stationary series, a formal stationarity test needs to be applied to the time series under consideration.

Augmented Dickey-Fuller Test: A formal test to check whether time series data follows stationary process.

Dickey-Fuller test
Statistical tests make strong assumptions about your data. They can only be used to inform the degree to which a null hypothesis can be accepted or rejected. The result must be interpreted for a given problem to be meaningful. Nevertheless, they can provide a quick check and confirmatory evidence that your time series is stationary or non-stationary.

Null Hypothesis (H0): If accepted, it suggests the time series has a unit root, meaning it is non-stationary. It has some time dependent structure.
Alternate Hypothesis (H1): The null hypothesis is rejected; it suggests the time series does not have a unit root, meaning it is stationary. It does not have time-dependent structure.

p-value > 0.05: Accept the null hypothesis (H0), the data has a unit root and is non-stationary.
p-value <= 0.05: Reject the null hypothesis (H0), the data does not have a unit root and is stationary.

$H0$: Time series is non-stationary $H1$: Time series is stationary

5.1 Check whether Sales series is stationary or not.

```
> ##Dickey-fuller test  adf
> adf.test(ItemA_Train, k=21)

        Augmented Dickey-Fuller Test

data:  ItemA_Train
Dickey-Fuller = -2.7407, Lag order = 21, p-value = 0.2672
alternative hypothesis: stationary

> adf.test(ItemB_Train, k=21)

        Augmented Dickey-Fuller Test

data:  ItemB_Train
Dickey-Fuller = -2.6554, Lag order = 21, p-value = 0.3028
alternative hypothesis: stationary

>
```

Item A or Item B sales data is not stationary. Log transformed data is put to Dickey-Fuller test to check for stationarity

```
> ##Dickey-fuller test
> adf.test(ItemA_Train_log, k=21)

        Augmented Dickey-Fuller Test

data:  ItemA_Train_log
Dickey-Fuller = -2.7724, Lag order = 21, p-value = 0.254
alternative hypothesis: stationary

> adf.test(ItemB_Train_log, k=21)

        Augmented Dickey-Fuller Test

data:  ItemB_Train_log
Dickey-Fuller = -2.5789, Lag order = 21, p-value = 0.3347
alternative hypothesis: stationary
```

From the above ADF test the Null Hypothesis is rejected.
The time series of differences (above) does appear to be stationary in mean and variance, as the level of the series stays roughly constant over time, and the variance of the series appears roughly constant over time.

Since Both Items sales data is non-stationary, it needs to convert into a stationary series.

Often differencing a time series lead to a stationary series.

5.2 ACF and PACF (performing to check the stationary data and autocorrelation)

The function Acf computes an estimate of the autocorrelation function of a (possibly multivariate) time series. Function Pacf computes an estimate of the partial autocorrelation function of a (possibly multivariate) time series.

ACF
Autocorrelation of different orders gives inside information regarding time series. It will help determine order p of the series
Significant autocorrelations imply observations of long past influences current observation.
lag.max: maximum lag at which to calculate the acf. Default is $10*\log10(N/m)$
where N is the number of observations and m the number of series.
Will be automatically limited to one less than the number of observations in the series

PACF
Partial autocorrelation adjusts for the intervening period.

- ACF and PACF used together to identify the order of the ARMA.
- Seasonal ACF and PACF examines correlations for seasonal data

```
> ##Run ACF and PACF plots
> par(mfrow = c(1,2))
> acf(ts(ItemA_Train_log),main="ACF Item A Sales")
> pacf(ts(ItemA_Train_log),main=" PACF Item A Sales")
> par(mfrow = c(1,2))
> acf(ts(ItemB_Train_log),main="ACF Item B Sales")
> pacf(ts(ItemB_Train_log),main=" PACF Item B Sales")
>
```

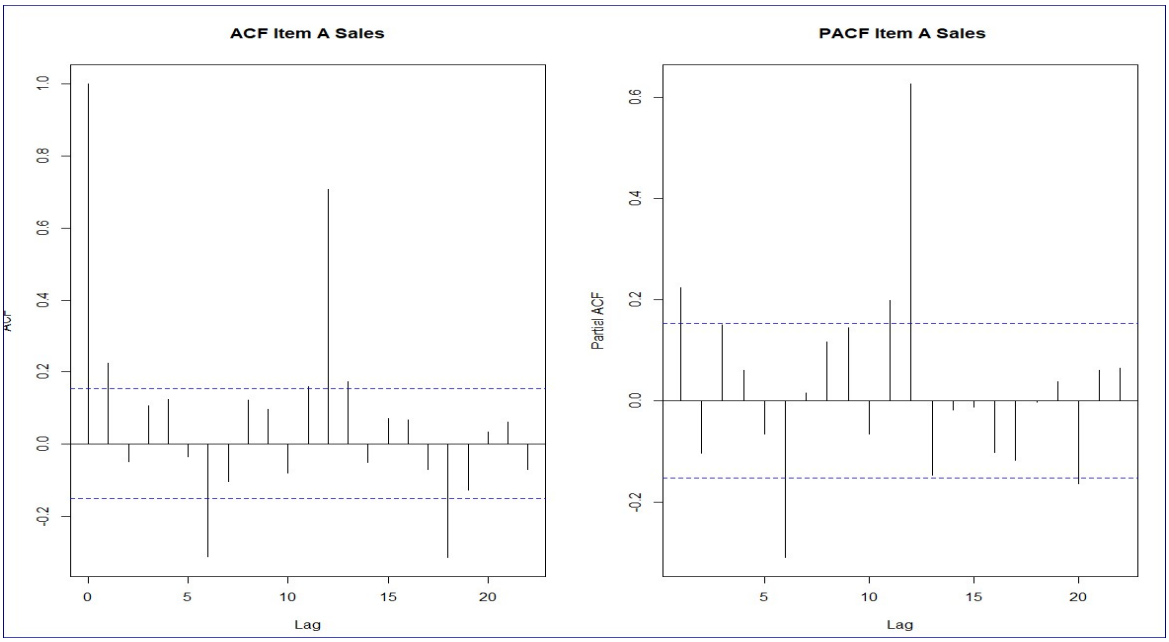

Figure 19: ACF and PACF of Item A Sales after log transformation

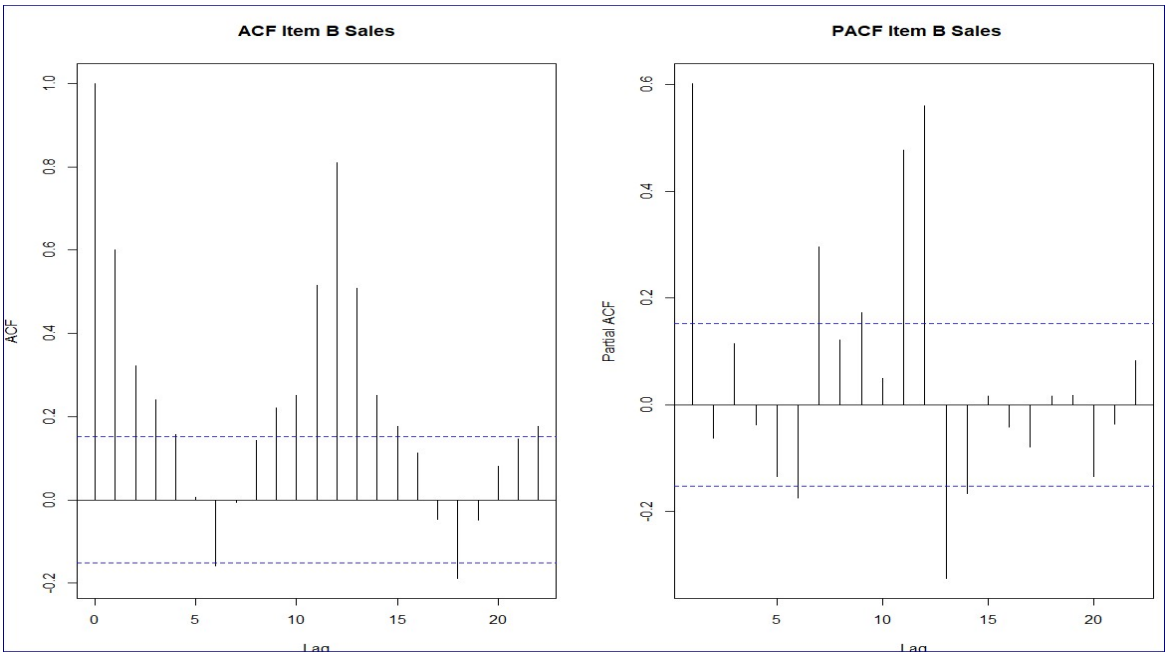

Figure 20: ACF and PACF of Item B Sales after log transformation

Using Figure (19: 20 indicates the presence of seasonality in using ACF and PACF plots of tractor sales data as the past values are significantly correlated. And, it is also clear that at every multiple of 12 the ACF swings higher than neighboring values.

6. ARIMA Model

Seasonal ARIMA models are more complex models with seasonal adjustments. These models are used when time series data has significant seasonality. The most general form of seasonal ARIMA is $ARIMA(p,d,q)*ARIMA(P,D,Q)$ [m], where P, D, Q are defined as seasonal AR

component, seasonal difference and seasonal MA component respectively. And, 'm' represents the frequency (time interval) at which the data is observed. For example, a monthly series will have m = 12.

ACF and PACF may be used to understand seasonality.

```
> #Step: Run auto arima
> ItemA_AutoARIMA <- auto.arima(ItemA_Train_log)
> ItemA_AutoARIMA
Series: ItemA_Train_log
ARIMA(1,0,0)(1,1,1)[12] with drift

Coefficients:
         ar1    sar1     sma1   drift
      0.1038  0.1008  -0.7538  0.0012
s.e.  0.0846  0.1546   0.1384  0.0003

sigma^2 estimated as 0.01112:  log likelihood=125.84
AIC=-241.68   AICc=-241.27   BIC=-226.5
```

According to the result, $ARIMA(0,1,1)(0,1,1)[12]$ is the indicated model for Sales of Item A AIC= 241.68) and BIC= (226.5).

Alternatively, one might investigate other suitable model(s) for a time series using ACF and PACF for the differenced series.

```
> ItemA_Train_log %>% diff() %>% ggtsdisplay()
> ItemA_Train_log %>% diff(lag=12) %>% diff()%>% ggtsdisplay()
```

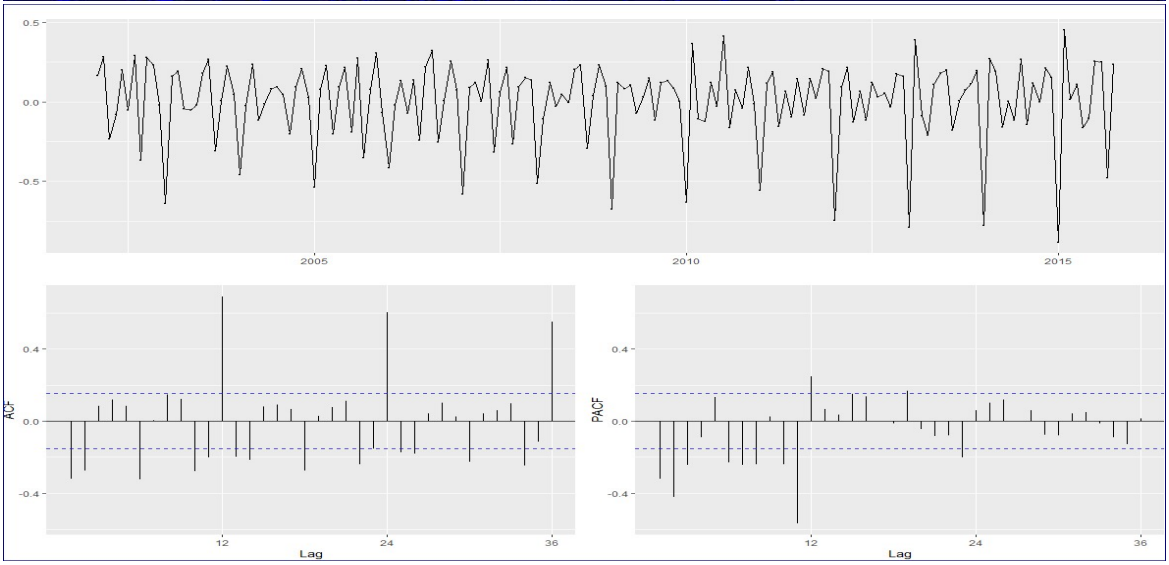

Figure 21: Plot of first difference series, ACF and PACF of Log(Item A)

The plots of ACF and PACF indicate possible values for p to be 1 and q to be 0.

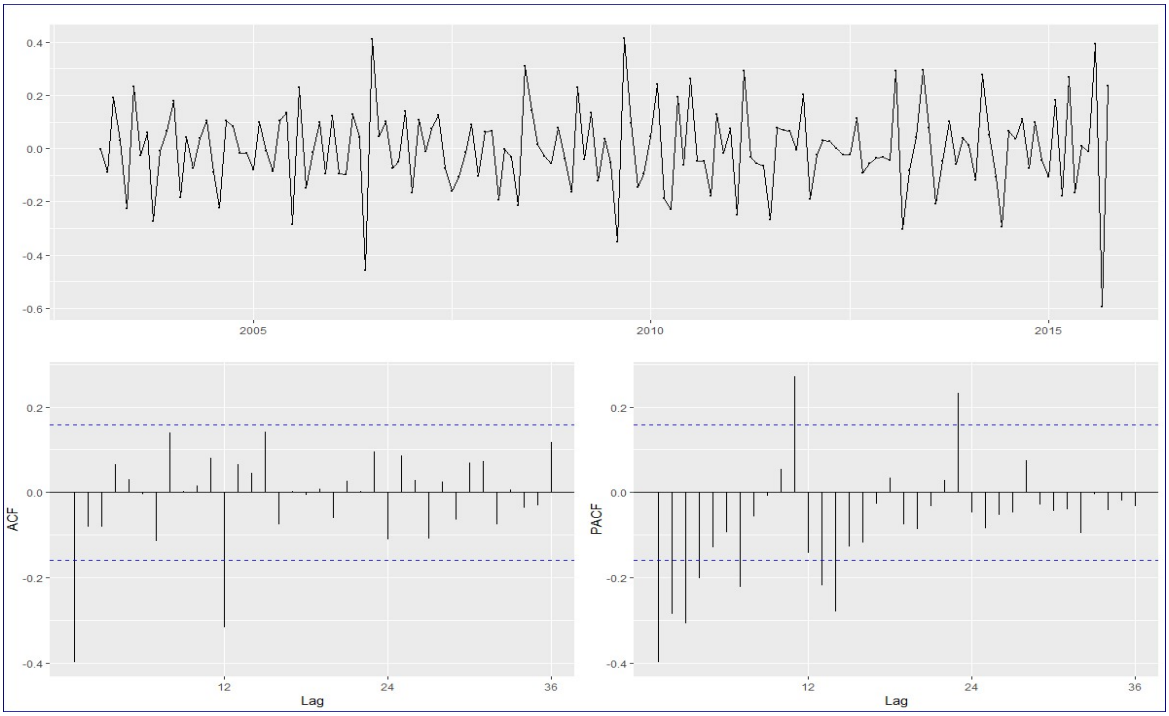

Figure 22: Plot of first difference and seasonal first difference series, ACF and PACF of Log(Item A Sales)

```
> ItemB_AutoARIMA <- auto.arima(ItemB_Train_log)
> ItemB_AutoARIMA
Series: ItemB_Train_log
ARIMA(0,0,0)(2,1,1)[12] with drift

Coefficients:
        sar1    sar2     sma1    drift
      0.0791  0.0263  -0.7170  -0.0035
s.e.  0.1718  0.1341   0.1549   0.0003

sigma^2 estimated as 0.01009:  log likelihood=133.97
AIC=-257.94   AICc=-257.53   BIC=-242.75
```

According to the result, *ARIMA*(0,1,1)(0,1,1)[12] is the indicated model for Sales of Item B AIC= 257.94) and BIC= (242.75).

```
> ItemB_Train_log %>% diff() %>% ggtsdisplay()
> ItemB_Train_log %>% diff(lag=12) %>% diff()%>% ggtsdisplay()
>
```

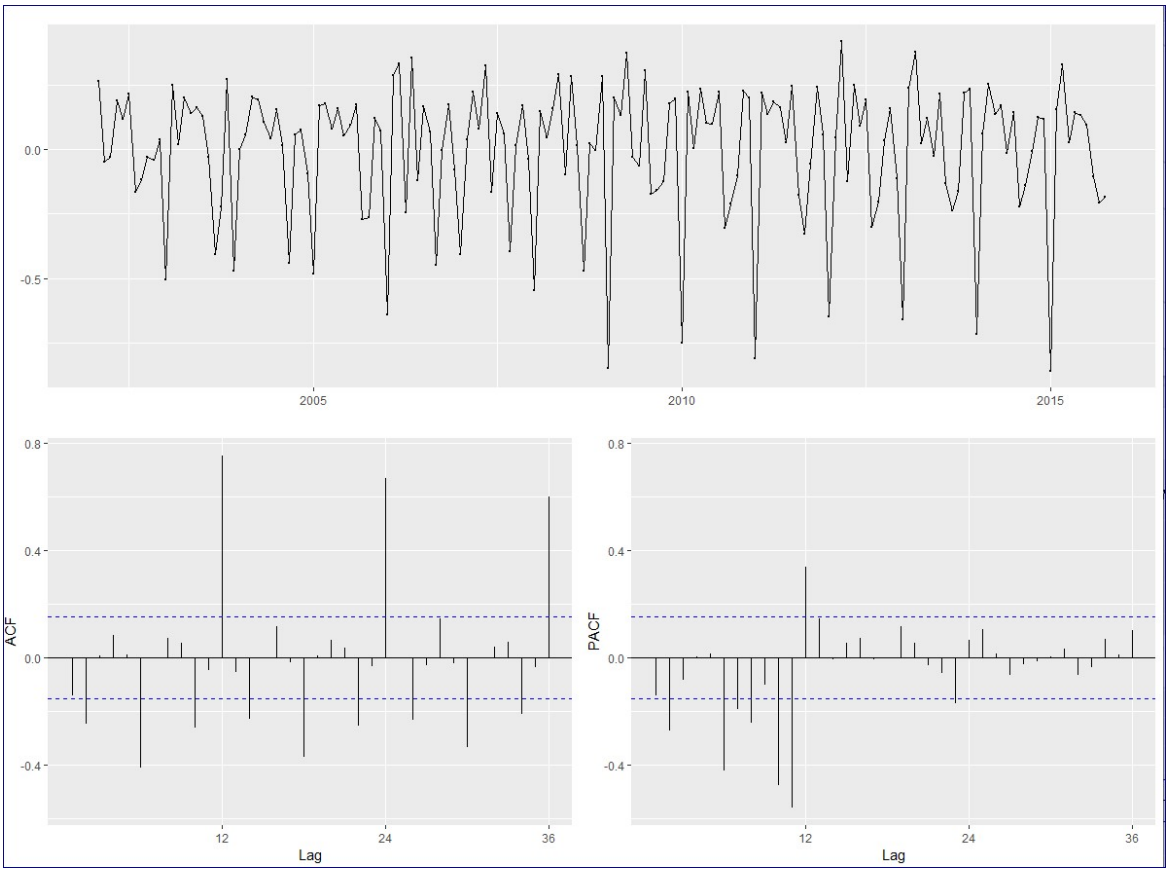

Figure 23: Plot of first difference series, ACF and PACF of Log(Item B)

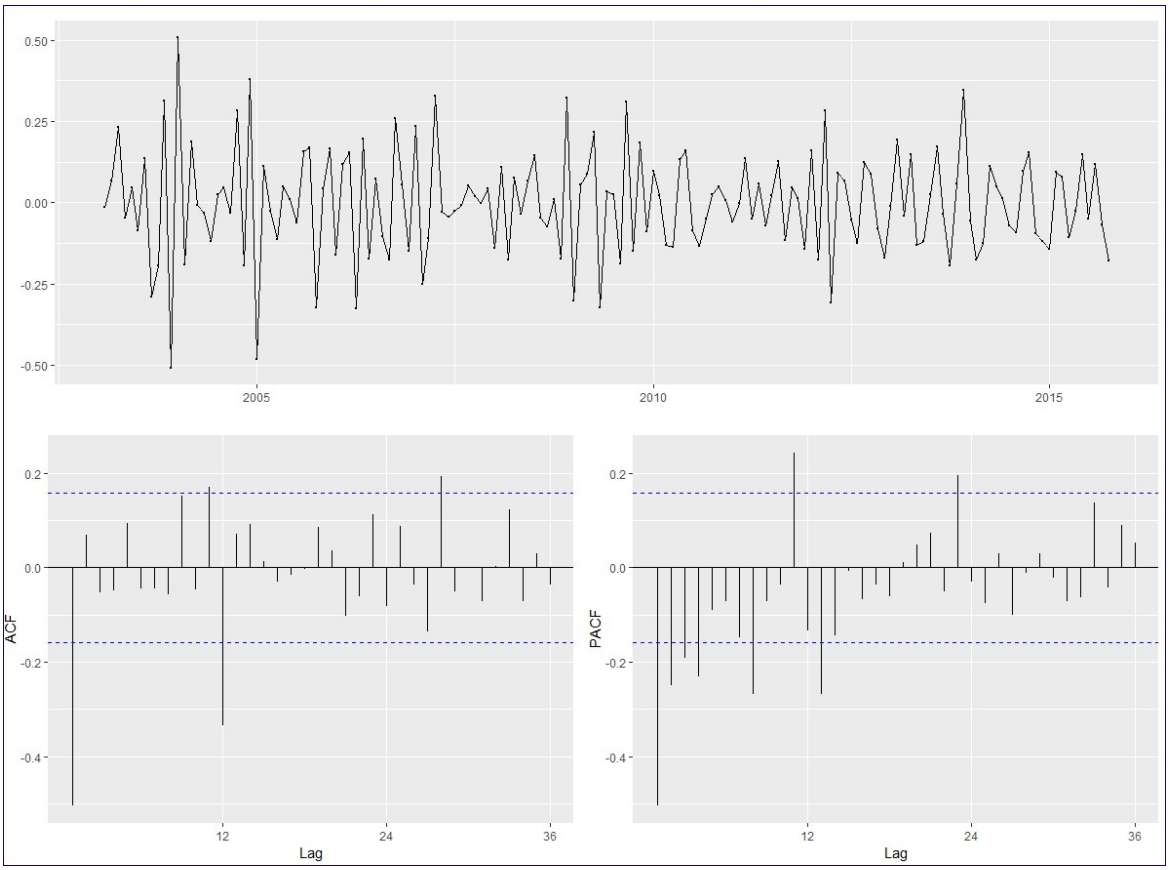

Figure 23: Plot of first difference and seasonal first difference series, ACF and PACF of Log(Item b Sales)

Both ACF and PACF plots indicate possible value of P to be 1 and Q to be 1.

One can choose different values of p,d,q ($or\ P,D\ and\ Q$) and compare AIC, BIC values and select the 'best' model accordingly.

We have selected here go with $ARIMA(0,1,1)(1,1,1)[12]$ as an alternative model and got lower values for AIC and BIC compared to results obtained from using auto.arima().

ARIMA of Item A

```
> ItemA_f = Arima(ItemA_Train_log,order=c(0,1,1), seasonal=c(1,1,1), method = "ML")
> ItemA_f
Series: ItemA_Train_log
ARIMA(0,1,1)(1,1,1)[12]

Coefficients:
          ma1     sar1     sma1
      -0.8753  -0.0177  -0.6834
s.e.   0.0489   0.1518   0.1380

sigma^2 estimated as 0.01132:  log likelihood=122.48
AIC=-236.96   AICc=-236.69   BIC=-224.84
```

ARIMA of Item B

```
> ItemB_f = Arima(ItemB_Train_log,order=c(0,1,1), seasonal=c(1,1,1), method = "ML")
> ItemB_f
Series: ItemB_Train_log
ARIMA(0,1,1)(1,1,1)[12]

Coefficients:
          ma1    sar1     sma1
      -0.9789  0.0458  -0.6741
s.e.   0.0373  0.1462   0.1212

sigma^2 estimated as 0.01027:  log likelihood=129.2
AIC=-250.4   AICc=-250.13   BIC=-238.28
```

Before ARIMA model is assumed to be reasonable for a series, it is important to check whether the residuals are following white noise or not. Towards that goal Box-Ljung test is applied.

Box- Ljung test: This check whether the residuals of time series data are stationary or not.

H0: Residuals are stationary H1: Residuals are not stationary

To check the residual of the series Box-Ljung test has been applied at different lags (k=6, 12 and 24). None of the p-values are significant implying that the series is stationary.

```
> ##checking white noise Item A
> Box.test(ItemA_f$residuals, lag=6, type="Ljung-Box")

        Box-Ljung test

data:  ItemA_f$residuals
X-squared = 4.4657, df = 6, p-value = 0.6139

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

        Box-Ljung test

data:  ItemA_f$residuals
X-squared = 11.04, df = 12, p-value = 0.5255

> Box.test(ItemA_f$residuals, lag=24, type="Ljung-Box")

        Box-Ljung test

data:  ItemA_f$residuals
X-squared = 18.17, df = 24, p-value = 0.7947

>
```

Box-Ljung Test
To check is resuidual are independent
H0: Residuals are independent
Ha: Residuals are not independent

```
~/My Files/R/R project files/
> ##checking white noise for Item B
> Box.test(ItemB_f$residuals, lag=6, type="Ljung-Box")

        Box-Ljung test

data:  ItemB_f$residuals
X-squared = 3.3746, df = 6, p-value = 0.7606

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

        Box-Ljung test

data:  ItemB_f$residuals
X-squared = 7.2419, df = 12, p-value = 0.8412

> Box.test(ItemB_f$residuals, lag=24, type="Ljung-Box")

        Box-Ljung test

data:  ItemB_f$residuals
X-squared = 22.175, df = 24, p-value = 0.5688
```

Model Validation: Seasonal ARIMA model with seasonal frequency m

```
~/My Files/R/R project files/
> ##Run auto arima to get first preview of the fitted model
> ItemASalesForecasts <- forecast(ItemA_f, h=21)
> ItemBSalesForecasts <- forecast(ItemB_f, h=21)
> #par(mfrow = c(1,2))
> plot(ItemASalesForecasts, shadecols = "oldstyle",main ="ItemA Forecast ARIMA (0,1,1)*(1,1,1)[1
2]")
> plot(ItemBSalesForecasts, shadecols = "oldstyle",main ="ItemB Forecast ARIMA (0,1,1)*(1,1,1)[1
2]")
>
```

Forecasting made using SARIMA model (0,1,1)*(1,1,1) is as given below

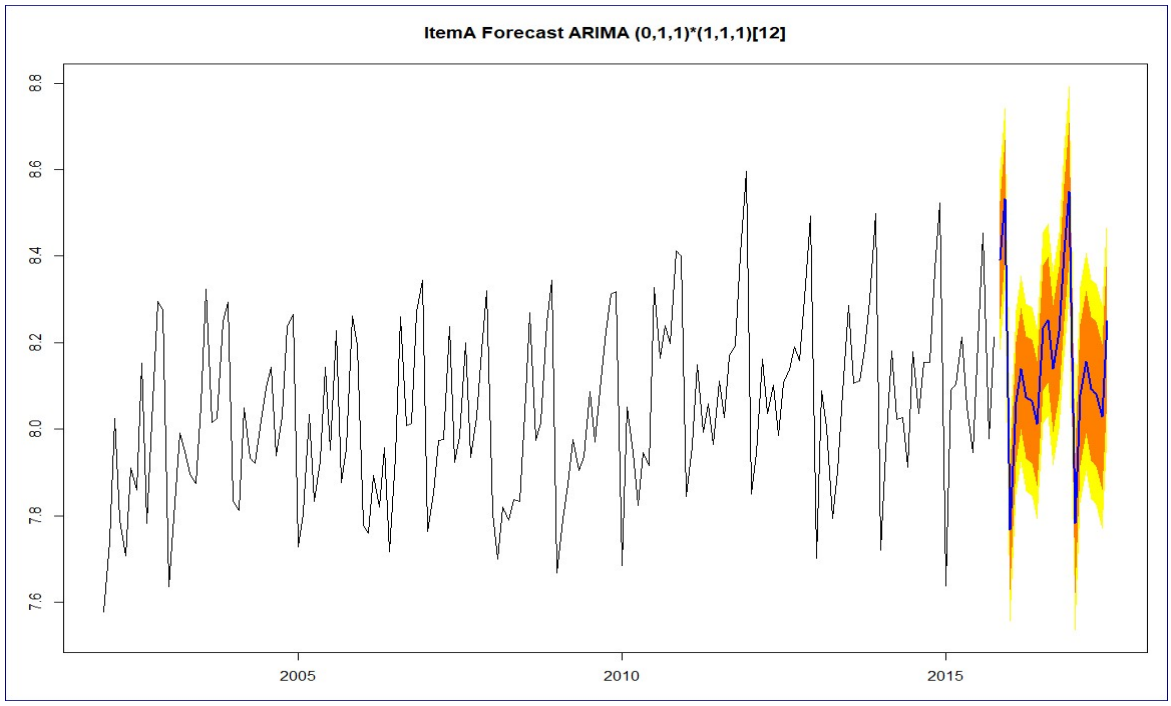

Figure 24: Tractor sales data forecast using ARIMA (0,1,1)(1,1,1)[12] for Item A

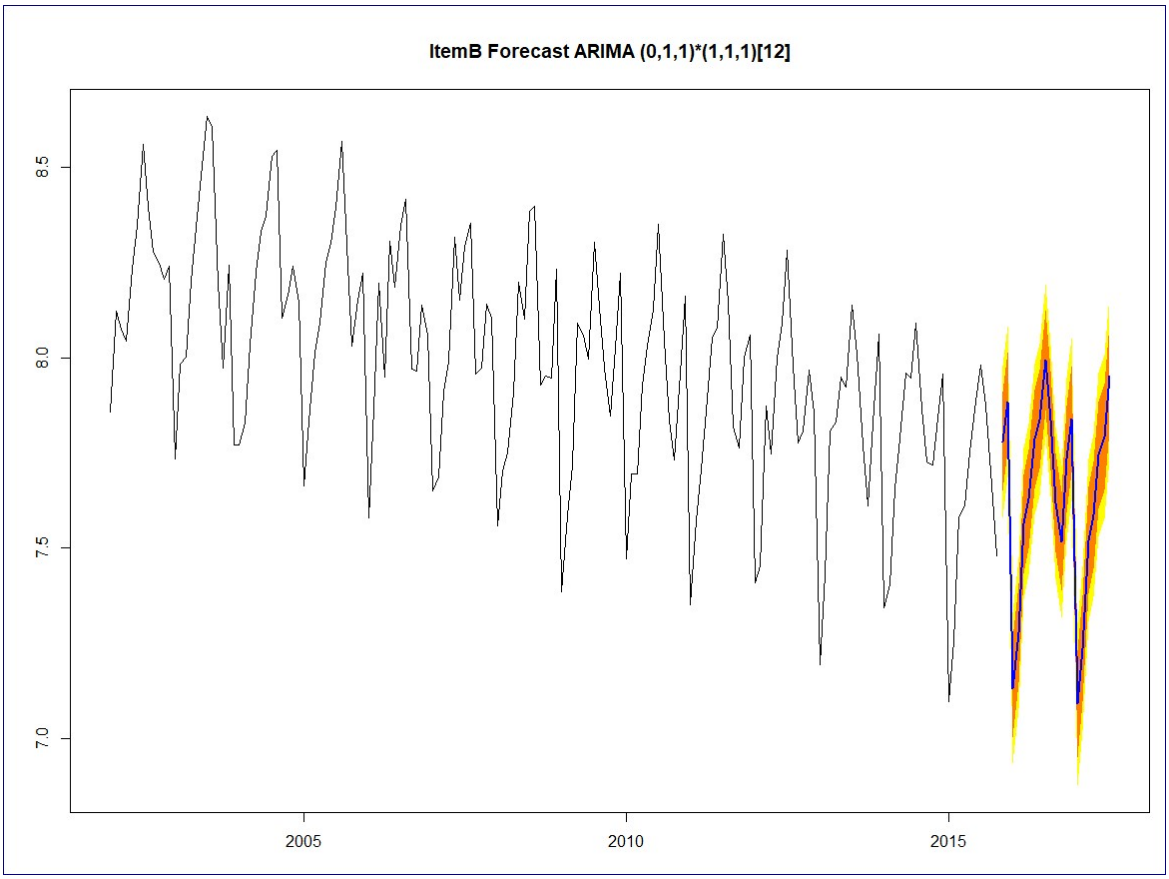

Figure 25: Tractor sales data forecast using ARIMA (0,1,1)(1,1,1)[12] for Item B

6.1 Accuracy measures: SARIMA (0,1,1)*(1,1,1)[12]

```
~/My Files/R/R project files/
> ###Accuracy measures: RMSE and MAPE using SARIMA
> ItemAVec1<- 10^(cbind(log(ItemA_Test) ,as.data.frame(forecast(ItemA_f,h=21))[,1]))
> ItemBVec1<- 10^(cbind(log(ItemB_Test) ,as.data.frame(forecast(ItemB_f,h=21))[,1]))
> par(mfrow = c(1,2))
> ts.plot(ItemAVec1, col=c("blue", "red"), main="Item A sales: Actual vs Forecast")
> ts.plot(ItemBVec1, col=c("blue", "red"), main="Item B sales: Actual vs Forecast")
>
```

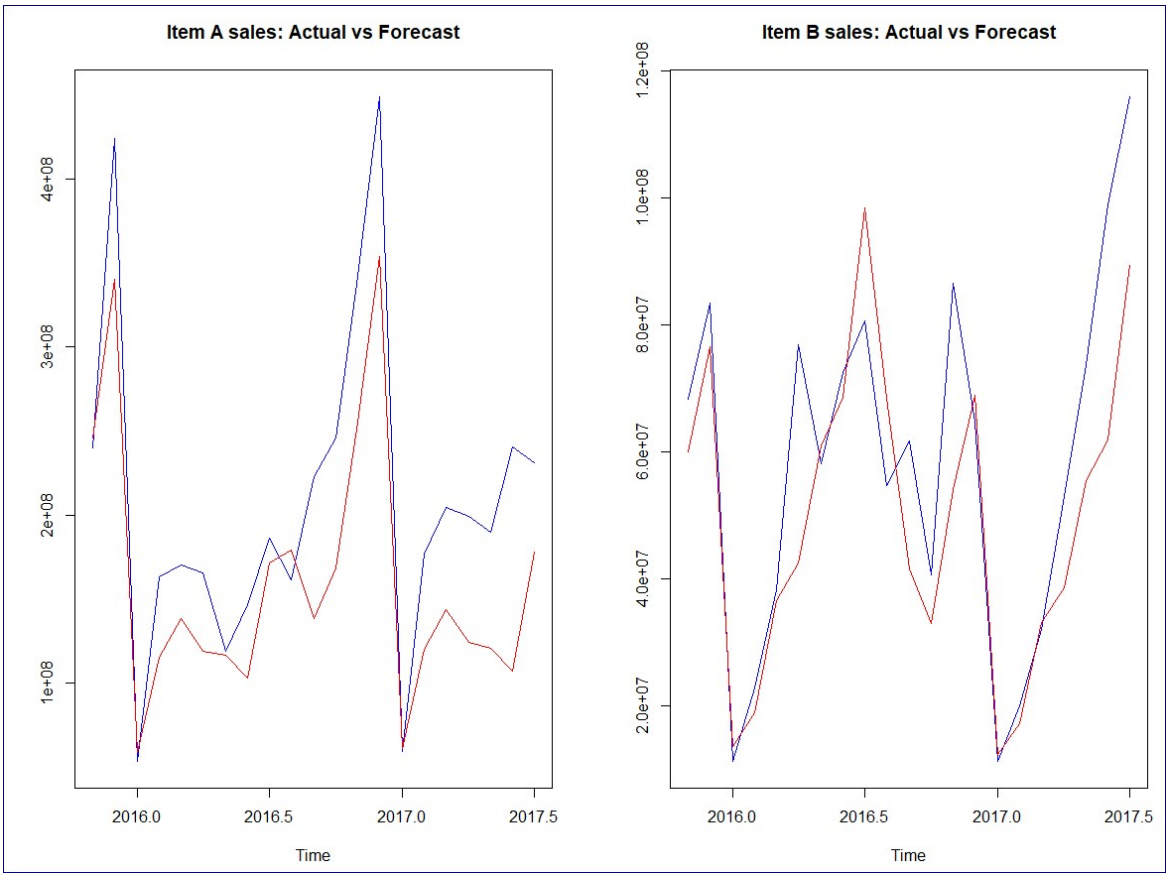

Figure 26: Plot Actual vs. Forecasted sales using ARIMA (0,1,1)*(1,1,1)[12] for 2016-2017 years for Item A and Item B

```
> RMSE1A <- round(sqrt(sum(((ItemAVec1[,1]-ItemAVec1[,2])^2)/length(ItemAVec1[,1]))),4)
> MAPE1A <- round(mean(abs(ItemAVec1[,1]-ItemAVec1[,2])/ItemAVec1[,1]),4)
> paste("Accuracy Measures: RMSE:", RMSE1A, "and MAPE:", MAPE1A)
[1] "Accuracy Measures: RMSE: 62879645.3553 and MAPE: 0.235"
> RMSE1B <- round(sqrt(sum(((ItemBVec1[,1]-ItemBVec1[,2])^2)/length(ItemBVec1[,1]))),4)
> MAPE1B <- round(mean(abs(ItemBVec1[,1]-ItemBVec1[,2])/ItemBVec1[,1]),4)
> paste("Accuracy Measures: RMSE:", RMSE1B, "and MAPE:", MAPE1B)
[1] "Accuracy Measures: RMSE: 16925834.4229 and MAPE: 0.1903"
>
```

6.2 Comparison of Model

```
> ##Comparison MAPE
> RWWDA <-paste("Accuracy Measures: RMSE:", RMSEA2, "and MAPE:", MAPEA2)
> RWWDB <-paste("Accuracy Measures: RMSE:", RMSEB2, "and MAPE:", MAPEB2)
> HWMA <- paste("Accuracy Measures: RMSE:", RMSE, "and MAPE:", MAPE)
> HWMB <-paste("Accuracy Measures: RMSE:", RMSEB, "and MAPE:", MAPEB)
> ARIMAA <-paste("Accuracy Measures: RMSE:", RMSE1A, "and MAPE:", MAPE1A)
> ARIMAB <- paste("Accuracy Measures: RMSE:", RMSE1B, "and MAPE:", MAPE1B)
>
> RWWDA
[1] "Accuracy Measures: RMSE: 587.2886 and MAPE: 0.1318"
> RWWDB
[1] "Accuracy Measures: RMSE: 460.9989 and MAPE: 0.1987"
> HWMA
[1] "Accuracy Measures: RMSE: 467.8315 and MAPE: 0.1014"
> HWMB
[1] "Accuracy Measures: RMSE: 326.803 and MAPE: 0.1081"
> ARIMAA
[1] "Accuracy Measures: RMSE: 62879645.3553 and MAPE: 0.235"
> ARIMAB
[1] "Accuracy Measures: RMSE: 16925834.4229 and MAPE: 0.1903"
>
```

Comparison of Models	**Item A**			**Item B**	
Model	**RMSE**	**MAPE**		**RMSE**	**MAPE**
Random walk with drift	587.2886	0.1318		460.9989	0.1987
Holt Winter's Method	467.8315	0.1014		326.803	0.1081
ARIMA (0,1,1)(1,1,1)[12]	62879645.36	0.235		16925834	0.1903

6.3 Conclusion

For Time Series Forecasting problem, we observed the trend and seasonality in the data.
We have observed that the Item A has increasing trend, but for Item B the trend is declining.
Also, we observed for both item there are few months with high variation in seasonality; and for Item A there are few outliers.

As the seasonality was not following the trend pattern we have used the "Additive" seasonality.
We have performed the three models
1. Random Walk with Drift,
2. Holt Winters and
3. ARIMA model.

Below are MAPE and Box-Ljung test observations for Models.

Now, using above , we are comparing results obtained from Random walk with drift, HoltWinter's method and ARIMA models, it is concluded that HW method predicts well as compared to other models. As the value of MAPE measure is minimum though the value of RMSE value has little higher.

6.4 Final Forecasts

Once a model is chosen and validated, forecasts into the future to be determined. Item A and Item B sales to be forecasted for 17months: 2017 Aug – 2018 Dec.

Going strictly by MAPE, recommended model is Holt Winter's method.

```
> ###### Actual forecast for next 17 months(2017 Julyto till 2018 December)
> ItemA_HW <- hw(Consumable_ItemA, seasonal = "multiplicative")
> plot(ItemA_HW)
>
```

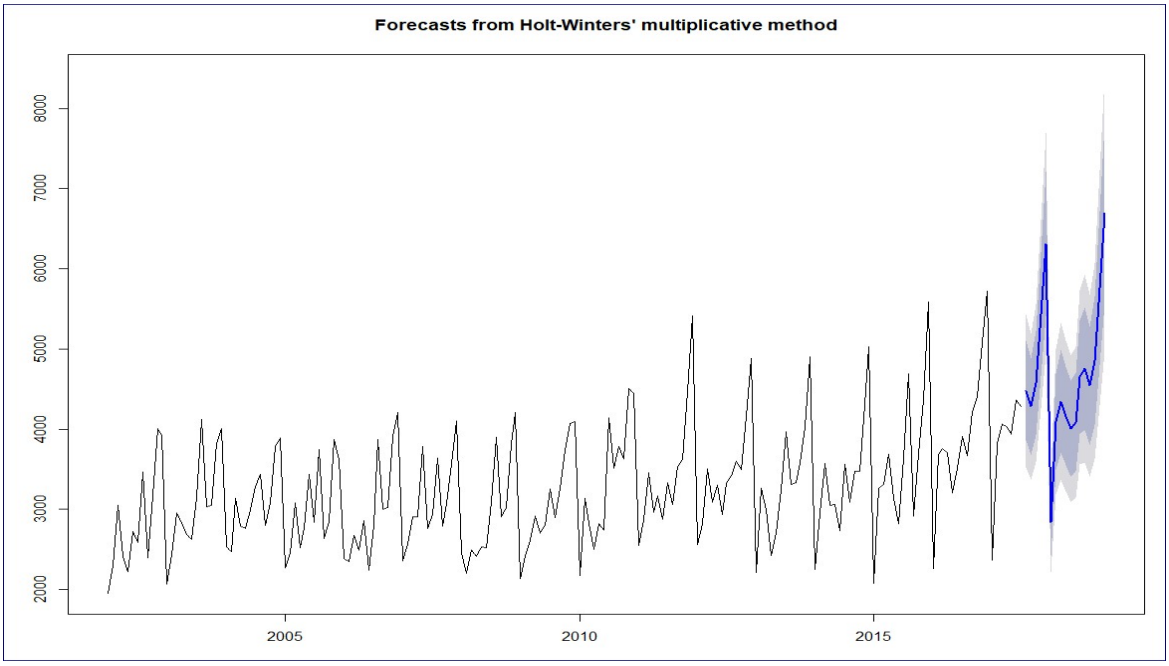

Figure 27: Plot Actual vs. Forecasted sales using HW method for 2017-2018 years for Item A

```
> ItemB_HW <- hw(Consumable_ItemB, seasonal = "multiplicative")
> plot(ItemB_HW)
>
```

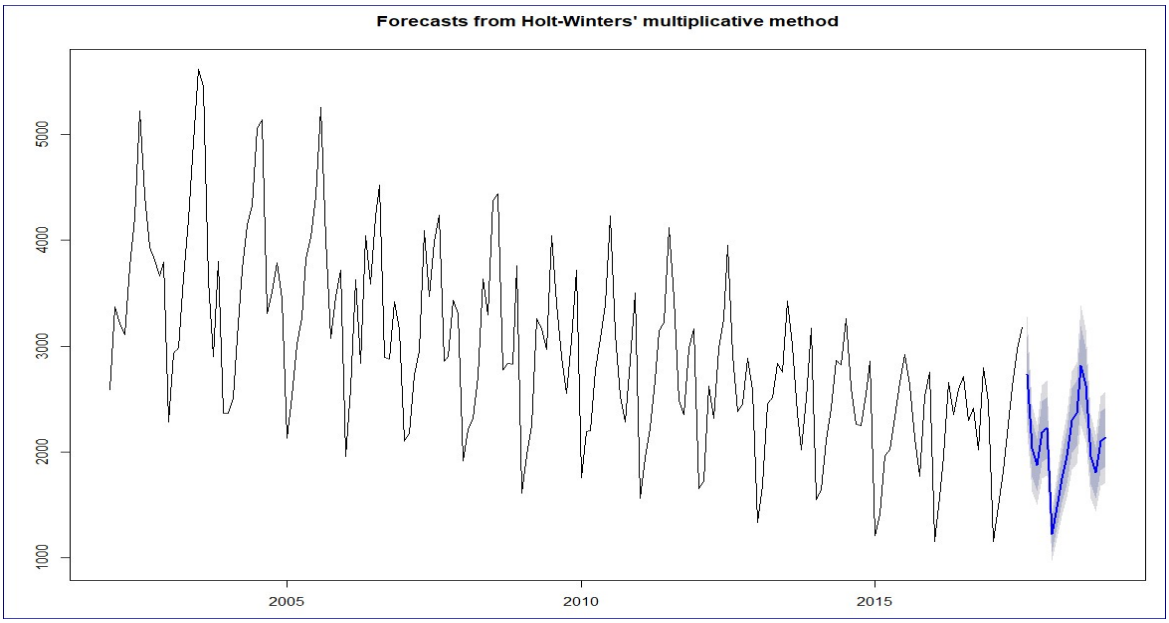

Figure 28: Plot Actual vs. Forecasted sales using HW method for 2017-2018 years for Item B

```
> ItemA_HW$model
Holt-Winters' multiplicative method

Call:
 hw(y = Consumable_ItemA, seasonal = "multiplicative")

  Smoothing parameters:
    alpha = 0.1116
    beta  = 0.0043
    gamma = 0.238

  Initial states:
    l = 2991.6494
    b = 15.4651
    s = 1.317 1.2694 1.0208 0.9394 1.1879 0.9874
          0.9298 0.9001 0.8842 0.9915 0.8206 0.752

  sigma:  0.1084

     AIC     AICc      BIC
3182.546 3186.168 3237.475
> ItemB_HW$model
Holt-Winters' multiplicative method
```

```
> ItemB_HW$model
Holt-Winters' multiplicative method

Call:
 hw(y = Consumable_ItemB, seasonal = "multiplicative")

  Smoothing parameters:
    alpha = 0.0136
    beta  = 0.0015
    gamma = 1e-04

  Initial states:
    l = 4075.2928
    b = -12.3577
    s = 1.0656 1.0431 0.8915 0.963 1.2909 1.3802
           1.1552 1.1173 0.9507 0.8443 0.7131 0.5851

  sigma:  0.1027

     AIC     AICc      BIC
3115.759 3119.380 3170.688
>
```

```
> HW_fore_finalA<-forecast(ItemA_HW, h=17)
> HW_fore_finalB<-forecast(ItemB_HW, h=17)
> round(HW_fore_finalA$mean,2)
        Jan     Feb     Mar     Apr     May     Jun     Jul     Aug     Sep     Oct
2017                                                            4478.75 4280.63 4574.39
2018 2834.95 4091.39 4347.87 4164.70 4008.57 4086.02 4647.03 4753.24 4541.66 4851.95
        Nov     Dec
2017 5430.49 6314.39
2018 5758.36 6693.75
> round(HW_fore_finalB$mean,2)
        Jan     Feb     Mar     Apr     May     Jun     Jul     Aug     Sep     Oct
2017                                                            2732.64 2032.12 1875.12
2018 1218.83 1480.41 1747.19 1960.99 2296.87 2366.97 2818.73 2627.38 1953.59 1802.43
        Nov     Dec
2017 2186.96 2226.92
2018 2101.90 2140.03
>
```

6.5 Actual Forecast using SARIMA(0,1,1)(1,1,1)[12] method

```
> ##Actual forecast by SARIMA (0,1,1)(1,1,1)[12]
> ItemA_final_arima=Arima(log10(Consumable_ItemA),order=c(0,1,1), seasonal=c(1, 1,1), method =
 "ML")
> ItemB_final_arima=Arima(log10(Consumable_ItemB),order=c(0,1,1), seasonal=c(1, 1,1), method =
 "ML")
> ItemA_final_arima
Series: log10(Consumable_ItemA)
ARIMA(0,1,1)(1,1,1)[12]

Coefficients:
          ma1     sar1     sma1
      -0.8441  -0.0869  -0.6039
s.e.   0.0410   0.1290   0.1150

sigma^2 estimated as 0.002083:  log likelihood=287.71
AIC=-567.42   AICc=-567.18   BIC=-554.78
> ItemB_final_arima
Series: log10(Consumable_ItemB)
ARIMA(0,1,1)(1,1,1)[12]

Coefficients:
          ma1    sar1     sma1
      -1.0000  0.0268  -0.6643
s.e.   0.0245  0.1336   0.1086

sigma^2 estimated as 0.001929:  log likelihood=291.72
AIC=-575.45   AICc=-575.21   BIC=-562.81
```

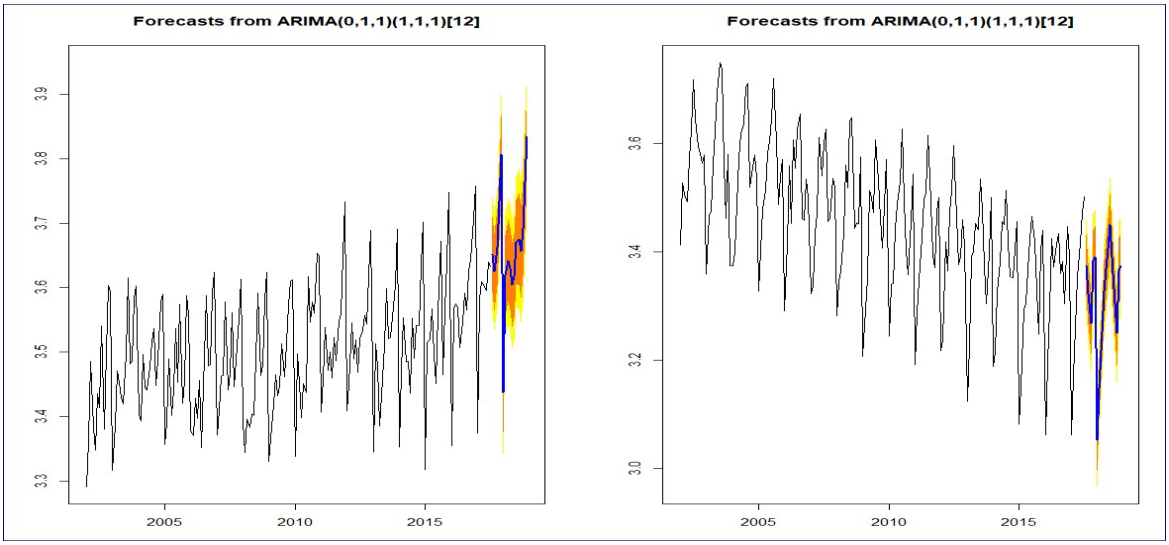

Figure 29: Plot Actual vs. Forecasted sales using SARIMA model for 2017-2018 years for Item A and Item B

```
> temASalesForecasts_actual <- forecast(ItemA_final_arima, h=17)
>
> ItemBSalesForecasts_actual <- forecast(ItemB_final_arima, h=17)
> par(mfrow = c(1,2))
> plot(ItemASalesForecasts_actual, shadecols = "oldstyle")
> plot(ItemBSalesForecasts_actual, shadecols = "oldstyle")
> Arima_fore_finalA <- round(10^ItemASalesForecasts_actual$mean,2)
> Arima_fore_finalA
        Jan     Feb     Mar     Apr     May     Jun     Jul     Aug     Sep     Oct
2017                                                    4478.52 4212.10 4589.46
2018 2734.61 4154.34 4383.63 4267.41 4020.14 4149.82 4678.13 4735.74 4532.57 4920.07
        Nov     Dec
2017 5416.69 6407.47
2018 5795.93 6826.58
> rima_fore_finalB <- round(10^ItemBSalesForecasts_actual$mean,2)
> Arima_fore_finalB
        Jan     Feb     Mar     Apr     May     Jun     Jul     Aug     Sep     Oct
2017                                                    2370.62 2106.06 1853.67
2018 1129.90 1404.52 1791.13 2127.85 2335.01 2546.23 2821.71 2284.32 2020.17 1780.62
        Nov     Dec
2017 2418.36 2461.74
2018 2319.18 2369.91
```

```
> ts.union(Arima_fore_finalA,Arima_fore_finalB)
         Arima_fore_finalA Arima_fore_finalB
Aug 2017           4478.52           2370.62
Sep 2017           4212.10           2106.06
Oct 2017           4589.46           1853.67
Nov 2017           5416.69           2418.36
Dec 2017           6407.47           2461.74
Jan 2018           2734.61           1129.90
Feb 2018           4154.34           1404.52
Mar 2018           4383.63           1791.13
Apr 2018           4267.41           2127.85
May 2018           4020.14           2335.01
Jun 2018           4149.82           2546.23
Jul 2018           4678.13           2821.71
Aug 2018           4735.74           2284.32
Sep 2018           4532.57           2020.17
Oct 2018           4920.07           1780.62
Nov 2018           5795.93           2319.18
Dec 2018           6826.58           2369.91
>
```

The store manager should keep after looking at the demand of Item A for over years.

“Thank you”

7. Source Code

Project TS.R

```
~/My Files/R/R project files/ ~/
> library(readxl)# read excel files
> library(ggfortify)
> library(data.table)# For converting data into time series format
> library(ggplot2)# To plot various plots
> library(fpp2)# For examining seasonality graphically
> library(forecast)# For various functions related to Time serie
> library(stats)# For applying tests like acf, Ljung-Box Tests
> library(tseries)  # For applying Dickey Fuller test
>
> ##Importa the data file
> Consumable_data<- read_excel("C:/Users/SuprasannaPradhan/Documents/My Files/Great Lakes Proje
cts/Demand.xlsx")
> str(Consumable_data)
Classes 'tbl_df', 'tbl' and 'data.frame':       187 obs. of  4 variables:
 $ Year  : num  2002 2002 2002 2002 2002 ...
 $ Month : num  1 2 3 4 5 6 7 8 9 10 ...
 $ Item A: num  1954 2302 3054 2414 2226 ...
 $ Item B: num  2585 3368 3210 3111 3756 ...
> ts(Consumable_data)
Time Series:
Start = 1
End = 187
Frequency = 1
    Year Month Item A Item B
  1 2002     1   1954   2585
  2 2002     2   2302   3368
  3 2002     3   3054   3210
  4 2002     4   2414   3111
  5 2002     5   2226   3756
  6 2002     6   2725   4216
  7 2002     7   2589   5225
  8 2002     8   3470   4426
  9 2002     9   2400   3932
 10 2002    10   3180   3816
 11 2002    11   4009   3661
 12 2002    12   3924   3795
 13 2003     1   2072   2285
 14 2003     2   2434   2934
```

```
>
> ##convert data into time seires format
> Consumable_ItemA<-ts(Consumable_data[,3],start = c(2002,1),end=c(2017,7),frequency = 12 )
> Consumable_ItemB<-ts(Consumable_data[,4],start = c(2002,1),end=c(2017,7),frequency = 12 )
> autoplot(ts(cbind(Consumable_ItemA,Consumable_ItemB),start = c(2002,1),frequency = 12 ),
+          facets = FALSE)
>
> ## Plot the Time series data
> par(mfrow = c(1,2))
> plot(Consumable_ItemA, xlab="Years", ylab = "Sales of Item A",col="red", main = "Consumable Item A ")
> abline(reg=lm(Consumable_ItemA~time(Consumable_ItemA)))
> plot(Consumable_ItemB, xlab="Years", ylab = "Sales of Item B",col="blue",main = "Consumable Item B ")
> abline(reg=lm(Consumable_ItemB~time(Consumable_ItemB)))
> cycle(Consumable_ItemA)
     Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec
2002   1   2   3   4   5   6   7   8   9  10  11  12
2003   1   2   3   4   5   6   7   8   9  10  11  12
2004   1   2   3   4   5   6   7   8   9  10  11  12
2005   1   2   3   4   5   6   7   8   9  10  11  12
2006   1   2   3   4   5   6   7   8   9  10  11  12
2007   1   2   3   4   5   6   7   8   9  10  11  12
2008   1   2   3   4   5   6   7   8   9  10  11  12
2009   1   2   3   4   5   6   7   8   9  10  11  12
2010   1   2   3   4   5   6   7   8   9  10  11  12
2011   1   2   3   4   5   6   7   8   9  10  11  12
2012   1   2   3   4   5   6   7   8   9  10  11  12
2013   1   2   3   4   5   6   7   8   9  10  11  12
2014   1   2   3   4   5   6   7   8   9  10  11  12
2015   1   2   3   4   5   6   7   8   9  10  11  12
2016   1   2   3   4   5   6   7   8   9  10  11  12
2017   1   2   3   4   5   6   7
> cycle(Consumable_ItemB)
     Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec
2002   1   2   3   4   5   6   7   8   9  10  11  12
2003   1   2   3   4   5   6   7   8   9  10  11  12
2004   1   2   3   4   5   6   7   8   9  10  11  12
2005   1   2   3   4   5   6   7   8   9  10  11  12
```

```
> par(mfrow = c(2,2))
> plot(aggregate(Consumable_ItemA,FUN=mean))
> plot(aggregate(Consumable_ItemB,FUN=mean))
> boxplot(Consumable_ItemA~cycle(Consumable_ItemA))
> boxplot(Consumable_ItemB~cycle(Consumable_ItemB))
> ## Plot 1: Seasonal plot Year-wise
> ggseasonplot(Consumable_ItemA, year.labels=TRUE, year.labels.left=TRUE ) + ylab("degree") +
+             ggtitle("Seasonal plot: Sales Data of Item A")
> ggseasonplot(Consumable_ItemB, year.labels=TRUE, year.labels.left=TRUE ) + ylab("degree") +
+             ggtitle("Seasonal plot: Sales Data of Item B")
>
> ## Plot 2: Polar Seasonal plot Year-wise
> ggseasonplot(Consumable_ItemA, polar=TRUE) +   ylab("degree") +
+            ggtitle("Polar seasonal plot: Sales Data A")
>
> ggseasonplot(Consumable_ItemB, polar=TRUE) +   ylab("degree") +
+             ggtitle("Polar seasonal plot: Sales Data B")
>
> ## Seasonal plot Month-wise
> monthplot(Consumable_ItemA)
> monthplot(Consumable_ItemB)
> par(mfrow = c(1,2))
> monthplot(Consumable_ItemA)
> monthplot(Consumable_ItemB)
>
> ##Decomposition of TS
> ItemA<-decompose(Consumable_ItemA, type = "multiplicative")
> plot(ItemA)
> ItemB<-decompose(Consumable_ItemB, type = "multiplicative")
> plot(ItemB)
> ## Observing the seasonal indices of “TSDecmpose”
> Seasonal_A=round(t(ItemA$figure),3)
> colnames(Seasonal_A) <-c("Jan","Feb","Mar","Apr","May","Jun","Jul", "Aug","Sep","Oct","Nov","Dec")
> Seasonal_A
```

```
> Seasonal_A
      Jan   Feb   Mar   Apr   May   Jun   Jul   Aug   Sep   Oct   Nov  Dec
[1,] 0.71 0.856 0.954 0.885 0.914 0.884 1.024 1.114 0.976 1.041 1.262 1.38
> Seasonal_B=round(t(ItemB$figure),4)
> colnames(Seasonal_B) <-c("Jan","Feb","Mar","Apr","May","Jun","Jul", "Aug","Sep","Oct","Nov","Dec")
> Seasonal_B
        Jan   Feb    Mar    Apr    May    Jun    Jul    Aug    Sep   Oct    Nov    Dec
[1,] 0.5777 0.694 0.8586 0.9659 1.1276 1.1476 1.3715 1.2541 0.9603 0.888 1.0575 1.0974
>
> ## Decomposition of TS
> ItemA_se <-stl(log10(Consumable_ItemA[,1]), s.window='p')
> ItemB_se <-stl(log10(Consumable_ItemB[,1]), s.window='p')
> plot(ItemA_se,col='orange')
> plot(ItemB_se,col='blue')
>
>
> ##Spliting data into training and test data sets
> ## Spliting data set of Item A
> ItemA_Train <- window(Consumable_ItemA, start=c(2002,1), end=c(2015,10), freq=12)
> ItemA_Test <- window(Consumable_ItemA, start=c(2015,11),end=c(2017,7),freq=12)
> autoplot(ts(cbind(ItemA_Train,ItemA_Test),start = c(2002,1),frequency = 12 ),
+          facets = FALSE)
>
> ##Forecasting Methods
> ##Forcasting Item A  for 21 months
> ItemA_Train_log <- log(ItemA_Train)
> library(forecast)
> ItemADecmpose_Train_Log <-stl(ItemA_Train[,1],s.window="period")
> ItemA_Train_stl<-forecast(ItemADecmpose_Train_Log,method="rwdrift",h=21)
> plot(ItemA_Train_stl)
>
>
> ##Spliting data set for Item B
> ItemB_Train <- window(Consumable_ItemB, start=c(2002,1), end=c(2015,10), freq=12)
> ItemB_Test <- window(Consumable_ItemB, start=c(2015,11),freq=12)
> autoplot(ts(cbind(ItemB_Train,ItemB_Test),start = c(2002,1),frequency = 12 ),
```

```
>
> ##Forecasting Methods
> ##Forcasting Item B  for 21 months
> ItemB_Train_log <- log(ItemB_Train[,1])
> library(forecast)
> ItemBDecmpose_Train_Log <-stl(ItemB_Train[,1],s.window="period")
> ItemB_Train_stl<-forecast(ItemBDecmpose_Train_Log,method="rwdrift",h=21)
> plot(ItemB_Train_stl)
>
>
> ## Accuracy measures: RMSE and MAPE using RWD Item A
> ItemAVec2 <-cbind(ItemA_Test,ItemA_Train_stl$mean)
> ts.plot(ItemAVec2 , col=c("blue", "red"), main="Item A sales: Actual vs F orecast")
> RMSEA2 <- round(sqrt(sum(((ItemAVec2[,1]-ItemAVec2[,2])^2)/length(ItemAVec2[,1]))),4)
> MAPEA2 <- round(mean(abs(ItemAVec2[,1]-ItemAVec2[,2])/ItemAVec2[,1]),4)
> paste("Accuracy Measures: RMSE:", RMSEA2, "and MAPE:", MAPEA2)
[1] "Accuracy Measures: RMSE: 587.2886 and MAPE: 0.1318"
>
> ## Accuracy measures: RMSE and MAPE using RWD Item B
> ItemBVec2 <-cbind(ItemB_Test,ItemB_Train_stl$mean)
> ts.plot(ItemBVec2 , col=c("blue", "red"), main="Item B sales: Actual vs F orecast")
> RMSEB2 <- round(sqrt(sum(((ItemBVec2[,1]-ItemBVec2[,2])^2)/length(ItemBVec2[,1]))),4)
> MAPEB2 <- round(mean(abs(ItemBVec2[,1]-ItemBVec2[,2])/ItemBVec2[,1]),4)
> paste("Accuracy Measures: RMSE:", RMSEB2, "and MAPE:", MAPEB2)
[1] "Accuracy Measures: RMSE: 460.9989 and MAPE: 0.1987"
>
>
> ##Exponential Smoothing Method
> #Apply Holt winter method Item A
> #ItemA_Train_H <- window(Consumable_ItemA, start=c(2002,1), end=c(2015,7), freq=12)
> #ItemA_Test_H <- window(Consumable_ItemA, start=c(2015,8),end=c(2017,7),freq=12)
>
> #ItemB_Train<- window(Consumable_ItemB, start=c(2002,1), end=c(2015,7), freq=12)
> #ItemB_Test_H <- window(Consumable_ItemB, start=c(2015,8),freq=12)
>
> ItemA_Train_HW <- hw(ItemA_Train, seasonal = "multiplicative")
> plot(ItemA_Train_HW)
> ItemA_Train_HW$model
```

```
> ItemA_Train_HW <- hw(ItemA_Train, seasonal = "multiplicative")
> plot(ItemA_Train_HW)
> ItemA_Train_HW$model
Holt-Winters' multiplicative method

Call:
 hw(y = ItemA_Train, seasonal = "multiplicative")

  Smoothing parameters:
    alpha = 0.103
    beta  = 0.002
    gamma = 1e-04

  Initial states:
    l = 2979.4726
    b = 17.0502
    s = 1.357 1.2672 1.0339 0.9603 1.1173 1.0318
           0.8863 0.9128 0.8803 0.9674 0.8514 0.7343

  sigma:  0.1046

     AIC     AICc      BIC
2791.428 2795.563 2844.332
>
> ##Forecast data for next 21 months using HW method for Item A
> plot(forecast(ItemA_Train_HW, h=21))
>
>
> #Apply Holt winter method Item B
> ItemB_Train_HW <- hw(ItemB_Train, seasonal = "multiplicative")
> plot(ItemB_Train_HW)
> ItemB_Train_HW$model
Holt-Winters' multiplicative method

Call:
 hw(y = ItemB_Train, seasonal = "multiplicative")

  Smoothing parameters:
```

```
> ItemB_Train_HW$model
Holt-Winters' multiplicative method

Call:
 hw(y = ItemB_Train, seasonal = "multiplicative")

  Smoothing parameters:
    alpha = 0.0218
    beta  = 0.0014
    gamma = 1e-04

  Initial states:
    l = 4061.6997
    b = -12.8047
    s = 1.0543 1.0362 0.8955 0.962 1.3052 1.3877
          1.1468 1.118 0.9285 0.854 0.72 0.5918

  sigma:  0.0994

     AIC     AICc      BIC
2749.006 2753.142 2801.910
> ItemB_Train_HW
         Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
Nov 2015       2324.509 2028.464 2620.553 1871.748 2777.269
Dec 2015       2354.725 2054.751 2654.700 1895.954 2813.497
Jan 2016       1315.876 1148.191 1483.561 1059.424 1572.329
Feb 2016       1593.820 1390.645 1796.996 1283.090 1904.550
Mar 2016       1881.995 1641.990 2122.001 1514.939 2249.052
Apr 2016       2036.880 1777.008 2296.751 1639.441 2434.319
May 2016       2441.608 2129.948 2753.268 1964.966 2918.250
Jun 2016       2493.099 2174.695 2811.502 2006.142 2980.055
Jul 2016       3003.123 2619.355 3386.892 2416.200 3590.046
Aug 2016       2811.524 2452.007 3171.042 2261.690 3361.359
Sep 2016       2062.784 1798.823 2326.746 1659.090 2466.478
Oct 2016       1911.276 1666.513 2156.039 1536.943 2285.609
Nov 2016       2201.491 1919.323 2483.658 1769.953 2633.028
Dec 2016       2229.556 1943.532 2515.580 1792.119 2666.993
```

```
>
> ##Forecast data for next 21 months using HW method for Item B
> plot(forecast(ItemB_Train_HW, h=21))
>
> ## Accuracy measures: RMSE and MAPE using HW Item A
> ItemAVec<- cbind(ItemA_Test ,as.data.frame(forecast(ItemA_Train_HW,h=21))[,1])
> ts.plot(ItemAVec, col=c("blue", "red"), main="Sales:Item A Actual vs Forecast")
> RMSEA <- round(sqrt(sum(((ItemAVec[,1]-ItemAVec[,2])^2)/length(ItemAVec[,1]))),4)
> MAPEA <- round(mean(abs(ItemAVec[,1]-ItemAVec[,2])/ItemAVec[,1]),4)
> paste("Accuracy Measures: RMSE:", RMSEA, "and MAPE:", MAPEA)
[1] "Accuracy Measures: RMSE: 618.4349 and MAPE: 0.1356"
>
> ## Accuracy measures: RMSE and MAPE using HW Item B
> ItemBVec<- cbind(ItemB_Test ,as.data.frame(forecast(ItemB_Train_HW,h=21))[,1])
> ts.plot(ItemBVec, col=c("blue", "red"), main="Sales:  Item B Actual vs Forecast")
> RMSEB <- round(sqrt(sum(((ItemBVec[,1]-ItemBVec[,2])^2)/length(ItemBVec[,1]))),4)
> MAPEB <- round(mean(abs(ItemBVec[,1]-ItemBVec[,2])/ItemBVec[,1]),4)
> paste("Accuracy Measures: RMSE:", RMSEB, "and MAPE:", MAPEB)
[1] "Accuracy Measures: RMSE: 326.803 and MAPE: 0.1081"
>
>
>
> ##Dickey-fuller test  adf
> adf.test(ItemA_Train, k=21)

        Augmented Dickey-Fuller Test

data:  ItemA_Train
Dickey-Fuller = -2.7407, Lag order = 21, p-value = 0.2672
alternative hypothesis: stationary

> adf.test(ItemB_Train, k=21)

        Augmented Dickey-Fuller Test
```

```
data:  ItemA_Train
Dickey-Fuller = -2.7407, Lag order = 21, p-value = 0.2672
alternative hypothesis: stationary

> adf.test(ItemB_Train, k=21)

        Augmented Dickey-Fuller Test

data:  ItemB_Train
Dickey-Fuller = -2.6554, Lag order = 21, p-value = 0.3028
alternative hypothesis: stationary

> ##Dickey-fuller test
> adf.test(ItemA_Train_log, k=21)

        Augmented Dickey-Fuller Test

data:  ItemA_Train_log
Dickey-Fuller = -2.7724, Lag order = 21, p-value = 0.254
alternative hypothesis: stationary

> adf.test(ItemB_Train_log, k=21)

        Augmented Dickey-Fuller Test

data:  ItemB_Train_log
Dickey-Fuller = -2.5789, Lag order = 21, p-value = 0.3347
alternative hypothesis: stationary

> ##Run ACF and PACF plots
> par(mfrow = c(1,2))
> acf(ts(ItemA_Train_log),main="ACF Item A Sales")
> pacf(ts(ItemA_Train_log),main=" PACF Item A Sales")
> par(mfrow = c(1,2))
> acf(ts(ItemB_Train_log),main="ACF Item B Sales")
> pacf(ts(ItemB_Train_log),main=" PACF Item B Sales")
>
> ##Step: Run auto arima
> ItemA_AutoARIMA <- auto.arima(ItemA_Train_log)
> ItemA_AutoARIMA
Series: ItemA_Train_log
ARIMA(1,0,0)(1,1,1)[12] with drift

Coefficients:
         ar1    sar1     sma1   drift
      0.1038  0.1008  -0.7538  0.0012
s.e.  0.0846  0.1546   0.1384  0.0003

sigma^2 estimated as 0.01112:  log likelihood=125.84
AIC=-241.68   AICc=-241.27   BIC=-226.5
> ItemA_Train_log %>% diff() %>% ggtsdisplay()
> ItemA_Train_log %>% diff(lag=12) %>% diff()%>% ggtsdisplay()
>
> ItemB_AutoARIMA <- auto.arima(ItemB_Train_log)
> ItemB_AutoARIMA
Series: ItemB_Train_log
ARIMA(0,0,0)(2,1,1)[12] with drift

Coefficients:
        sar1    sar2     sma1    drift
      0.0791  0.0263  -0.7170  -0.0035
s.e.  0.1718  0.1341   0.1549   0.0003

sigma^2 estimated as 0.01009:  log likelihood=133.97
AIC=-257.94   AICc=-257.53   BIC=-242.75
> ItemB_Train_log %>% diff() %>% ggtsdisplay()
> ItemB_Train_log %>% diff(lag=12) %>% diff()%>% ggtsdisplay()
>
>
> ItemA_f = Arima(ItemA_Train_log,order=c(0,1,1), seasonal=c(1,1,1), method = "ML")
> ItemA_f
Series: ItemA_Train_log
```

```
Coefficients:
          ma1     sar1     sma1
      -0.8753  -0.0177  -0.6834
s.e.   0.0489   0.1518   0.1380

sigma^2 estimated as 0.01132:  log likelihood=122.48
AIC=-236.96   AICc=-236.69   BIC=-224.84
> ##checking white noise Item A
> Box.test(ItemA_f$residuals, lag=6, type="Ljung-Box")

        Box-Ljung test

data:  ItemA_f$residuals
X-squared = 4.4657, df = 6, p-value = 0.6139

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

        Box-Ljung test

data:  ItemA_f$residuals
X-squared = 11.04, df = 12, p-value = 0.5255

> Box.test(ItemA_f$residuals, lag=24, type="Ljung-Box")

        Box-Ljung test

data:  ItemA_f$residuals
X-squared = 18.17, df = 24, p-value = 0.7947

>
> ItemB_f = Arima(ItemB_Train_log,order=c(0,1,1), seasonal=c(1,1,1), method = "ML")
> ItemB_f
Series: ItemB_Train_log
ARIMA(0,1,1)(1,1,1)[12]

Coefficients:
          ma1    sar1     sma1
      -0.9789  0.0458  -0.6741
s.e.   0.0373  0.1462   0.1212

sigma^2 estimated as 0.01027:  log likelihood=129.2
AIC=-250.4   AICc=-250.13   BIC=-238.28
> ##checking white noise for Item B
> Box.test(ItemB_f$residuals, lag=6, type="Ljung-Box")

        Box-Ljung test

data:  ItemB_f$residuals
X-squared = 3.3746, df = 6, p-value = 0.7606

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

        Box-Ljung test

data:  ItemB_f$residuals
X-squared = 7.2419, df = 12, p-value = 0.8412

> Box.test(ItemB_f$residuals, lag=24, type="Ljung-Box")

        Box-Ljung test

data:  ItemB_f$residuals
X-squared = 22.175, df = 24, p-value = 0.5688

>
>
> ##Run auto arima to get first preview of the fitted model
> ItemASalesForecasts <- forecast(ItemA_f, h=21)
> ItemBSalesForecasts <- forecast(ItemB_f, h=21)
> #par(mfrow = c(1,2))
> plot(ItemASalesForecasts, shadecols = "oldstyle",main ="ItemA Forecast ARIMA (0,1,1)*(1,1,1)[12]")
> plot(ItemBSalesForecasts, shadecols = "oldstyle",main ="ItemB Forecast ARIMA (0,1,1)*(1,1,1)[12]")
```

```
> ##Accuracy measures: RMSE and MAPE using SARIMA
> ItemAVec1<- 10^(cbind(log(ItemA_Test) ,as.data.frame(forecast(ItemA_f,h=21))[,1]))
> ItemBVec1<- 10^(cbind(log(ItemB_Test) ,as.data.frame(forecast(ItemB_f,h=21))[,1]))
>
> par(mfrow = c(1,2))
> ts.plot(ItemAVec1, col=c("blue", "red"), main="Item A sales: Actual vs Forecast")
> ts.plot(ItemBVec1, col=c("blue", "red"), main="Item B sales: Actual vs Forecast")
>
> RMSE1A <- round(sqrt(sum(((ItemAVec1[,1]-ItemAVec1[,2])^2)/length(ItemAVec1[,1]))),4)
> MAPE1A <- round(mean(abs(ItemAVec1[,1]-ItemAVec1[,2])/ItemAVec1[,1]),4)
> paste("Accuracy Measures: RMSE:", RMSE1A, "and MAPE:", MAPE1A)
[1] "Accuracy Measures: RMSE: 62879645.3553 and MAPE: 0.235"
>
> RMSE1B <- round(sqrt(sum(((ItemBVec1[,1]-ItemBVec1[,2])^2)/length(ItemBVec1[,1]))),4)
> MAPE1B <- round(mean(abs(ItemBVec1[,1]-ItemBVec1[,2])/ItemBVec1[,1]),4)
> paste("Accuracy Measures: RMSE:", RMSE1B, "and MAPE:", MAPE1B)
[1] "Accuracy Measures: RMSE: 16925834.4229 and MAPE: 0.1903"
>
> ##Comparison MAPE
> RWWDA <-paste("Accuracy Measures: RMSE:", RMSEA2, "and MAPE:", MAPEA2)
> RWWDB <-paste("Accuracy Measures: RMSE:", RMSEB2, "and MAPE:", MAPEB2)
> HWMA <- paste("Accuracy Measures: RMSE:", RMSE, "and MAPE:", MAPE)
> HWMB <-paste("Accuracy Measures: RMSE:", RMSEB, "and MAPE:", MAPEB)
> ARIMAA <-paste("Accuracy Measures: RMSE:", RMSE1A, "and MAPE:", MAPE1A)
> ARIMAB <- paste("Accuracy Measures: RMSE:", RMSE1B, "and MAPE:", MAPE1B)
>
> RWWDA
[1] "Accuracy Measures: RMSE: 587.2886 and MAPE: 0.1318"
> RWWDB
[1] "Accuracy Measures: RMSE: 460.9989 and MAPE: 0.1987"
> HWMA
[1] "Accuracy Measures: RMSE: 467.8315 and MAPE: 0.1014"
> HWMB
[1] "Accuracy Measures: RMSE: 326.803 and MAPE: 0.1081"
```

```
> ##### Actual forecast for next 17 months(2017 Julyto till 2018 December)
> ItemA_HW <- hw(Consumable_ItemA, seasonal = "multiplicative")
> plot(ItemA_HW)
> plot(forecast(ItemA_HW,h=17))
> ItemB_HW <- hw(Consumable_ItemB, seasonal = "multiplicative")
> plot(ItemB_HW)
> plot(forecast(ItemB_HW,h=17))
> ItemA_HW$model
Holt-Winters' multiplicative method
```

```
> plot(forecast(ItemA_HW,h=17))
> ItemB_HW <- hw(Consumable_ItemB, seasonal = "multiplicative")
> plot(ItemB_HW)
> plot(forecast(ItemB_HW,h=17))
> ItemA_HW$model
Holt-Winters' multiplicative method

Call:
 hw(y = Consumable_ItemA, seasonal = "multiplicative")

  Smoothing parameters:
    alpha = 0.1116
    beta  = 0.0043
    gamma = 0.238

  Initial states:
    l = 2991.6494
    b = 15.4651
    s = 1.317 1.2694 1.0208 0.9394 1.1879 0.9874
           0.9298 0.9001 0.8842 0.9915 0.8206 0.752

  sigma:  0.1084

     AIC     AICc      BIC
3182.546 3186.168 3237.475
> ItemB_HW$model
Holt-Winters' multiplicative method

Call:
 hw(y = Consumable_ItemB, seasonal = "multiplicative")

  Smoothing parameters:
    alpha = 0.0136
    beta  = 0.0015
    gamma = 1e-04

  Initial states:
    l = 4075.2928
```

```
  Initial states:
    l = 4075.2928
    b = -12.3577
    s = 1.0656 1.0431 0.8915 0.963 1.2909 1.3802
           1.1552 1.1173 0.9507 0.8443 0.7131 0.5851

  sigma:  0.1027

     AIC     AICc      BIC
3115.759 3119.380 3170.688
>
> HW_fore_finalA<-forecast(ItemA_HW, h=17)
> HW_fore_finalB<-forecast(ItemB_HW, h=17)
> round(HW_fore_finalA$mean,2)
         Jan     Feb     Mar     Apr     May     Jun     Jul     Aug     Sep     Oct
2017                                                         4478.75 4280.63 4574.39
2018 2834.95 4091.39 4347.87 4164.70 4008.57 4086.02 4647.03 4753.24 4541.66 4851.95
         Nov     Dec
2017 5430.49 6314.39
2018 5758.36 6693.75
> round(HW_fore_finalB$mean,2)
         Jan     Feb     Mar     Apr     May     Jun     Jul     Aug     Sep     Oct
2017                                                         2732.64 2032.12 1875.12
2018 1218.83 1480.41 1747.19 1960.99 2296.87 2366.97 2818.73 2627.38 1953.59 1802.43
         Nov     Dec
2017 2186.96 2226.92
2018 2101.90 2140.03
>
> ##Actual forecast by SARIMA (0,1,1)(1,1,1)[12]
> ItemA_final_arima=Arima(log10(Consumable_ItemA),order=c(0,1,1), seasonal=c(1, 1,1), method = "ML")
> ItemB_final_arima=Arima(log10(Consumable_ItemB),order=c(0,1,1), seasonal=c(1, 1,1), method = "ML")
> ItemA_final_arima
Series: log10(Consumable_ItemA)
ARIMA(0,1,1)(1,1,1)[12]

Coefficients:
          ma1     sar1     sma1
```

```
Coefficients:
          ma1     sar1     sma1
      -0.8441  -0.0869  -0.6039
s.e.   0.0410   0.1290   0.1150

sigma^2 estimated as 0.002083:  log likelihood=287.71
AIC=-567.42   AICc=-567.18   BIC=-554.78
> ItemB_final_arima
Series: log10(Consumable_ItemB)
ARIMA(0,1,1)(1,1,1)[12]

Coefficients:
          ma1    sar1     sma1
      -1.0000  0.0268  -0.6643
s.e.   0.0245  0.1336   0.1086

sigma^2 estimated as 0.001929:  log likelihood=291.72
AIC=-575.45   AICc=-575.21   BIC=-562.81
>
> ItemASalesForecasts_actual <- forecast(ItemA_final_arima, h=17)
>
> ItemBSalesForecasts_actual <- forecast(ItemB_final_arima, h=17)
> par(mfrow = c(1,2))
> plot(ItemASalesForecasts_actual, shadecols = "oldstyle")
> plot(ItemBSalesForecasts_actual, shadecols = "oldstyle")
>
> Arima_fore_finalA <- round(10^ItemASalesForecasts_actual$mean,2)
> Arima_fore_finalA
         Jan     Feb     Mar     Apr     May     Jun     Jul     Aug     Sep     Oct
2017                                                         4478.52 4212.10 4589.46
2018 2734.61 4154.34 4383.63 4267.41 4020.14 4149.82 4678.13 4735.74 4532.57 4920.07
         Nov     Dec
2017 5416.69 6407.47
2018 5795.93 6826.58
> Arima_fore_finalB <- round(10^ItemBSalesForecasts_actual$mean,2)
> Arima_fore_finalB
         Jan     Feb     Mar     Apr     May     Jun     Jul     Aug     Sep     Oct
```

```
> Arima_fore_finalB
         Jan     Feb     Mar     Apr     May     Jun     Jul     Aug     Sep     Oct
2017                                                             2370.62 2106.06 1853.67
2018 1129.90 1404.52 1791.13 2127.85 2335.01 2546.23 2821.71 2284.32 2020.17 1780.62
         Nov     Dec
2017 2418.36 2461.74
2018 2319.18 2369.91
> ts.union(Arima_fore_finalA,Arima_fore_finalB)
         Arima_fore_finalA Arima_fore_finalB
Aug 2017           4478.52           2370.62
Sep 2017           4212.10           2106.06
Oct 2017           4589.46           1853.67
Nov 2017           5416.69           2418.36
Dec 2017           6407.47           2461.74
Jan 2018           2734.61           1129.90
Feb 2018           4154.34           1404.52
Mar 2018           4383.63           1791.13
Apr 2018           4267.41           2127.85
May 2018           4020.14           2335.01
Jun 2018           4149.82           2546.23
Jul 2018           4678.13           2821.71
Aug 2018           4735.74           2284.32
Sep 2018           4532.57           2020.17
Oct 2018           4920.07           1780.62
Nov 2018           5795.93           2319.18
Dec 2018           6826.58           2369.91
>
```