ARIMA Time Series Implementation

This is an implementation of ARIMA model for forecasting future sales for a retailer. The data consists product sales information of united states. There are three product categories mentioned in the data set Office supplies, Technology and Furniture

```
# Importing required libraries
library(tseries)

library(forecast)

library(dplyr)

library(xlsx)

library(readxl)

library(ggplot2)



# Assigning working directory
setwd("F:/Mayur/ANALYTICS/Data scientist/Data scienc with R/Time series")

# importing data
df = read_excel("Sample - Superstore.xls",sheet = "Orders")

## observations with sales >=12000
View(df %>% filter(Sales>=10000)) # techonology related products
```

Checking uniqueness of customer id and customer name

```
length(unique(df$`Customer ID`))

## [1] 793

length(unique(df$`Customer Name`))

## [1] 793
```

Category and sales comparison

```
df %>% group_by(Category) %>% summarise(N = n(),
                                        Sales_Max = max(Sales),
                                        Sales_min = min(Sales),
                                        Sales_Total = sum(Sales),
                                        Sales_mean = mean(Sales))
```

```
## # A tibble: 3 x 6
##   Category            N Sales_Max Sales_min Sales_Total Sales_mean
##   <chr>           <int>     <dbl>     <dbl>       <dbl>      <dbl>
## 1 Furniture        2121     4416.     1.89      742000.       350.
## 2 Office Supplies  6026     9893.     0.444     719047.       119.
## 3 Technology       1847    22638.     0.99      836154.       453.
```

```
df %>% filter(Sales<10000) %>% group_by(Category) %>% summarise(N = n(),
                                        Sales_Max = max(Sales),
                                        Sales_min = min(Sales),
                                        Sales_Total = sum(Sales),
                                        Sales_mean = mean(Sales))
```

```
## # A tibble: 3 x 6
##   Category            N Sales_Max Sales_min Sales_Total Sales_mean
##   <chr>           <int>     <dbl>     <dbl>       <dbl>      <dbl>
## 1 Furniture        2121     4416.     1.89      742000.       350.
## 2 Office Supplies  6026     9893.     0.444     719047.       119.
## 3 Technology       1842     9100.     0.99      760316.       413.
```

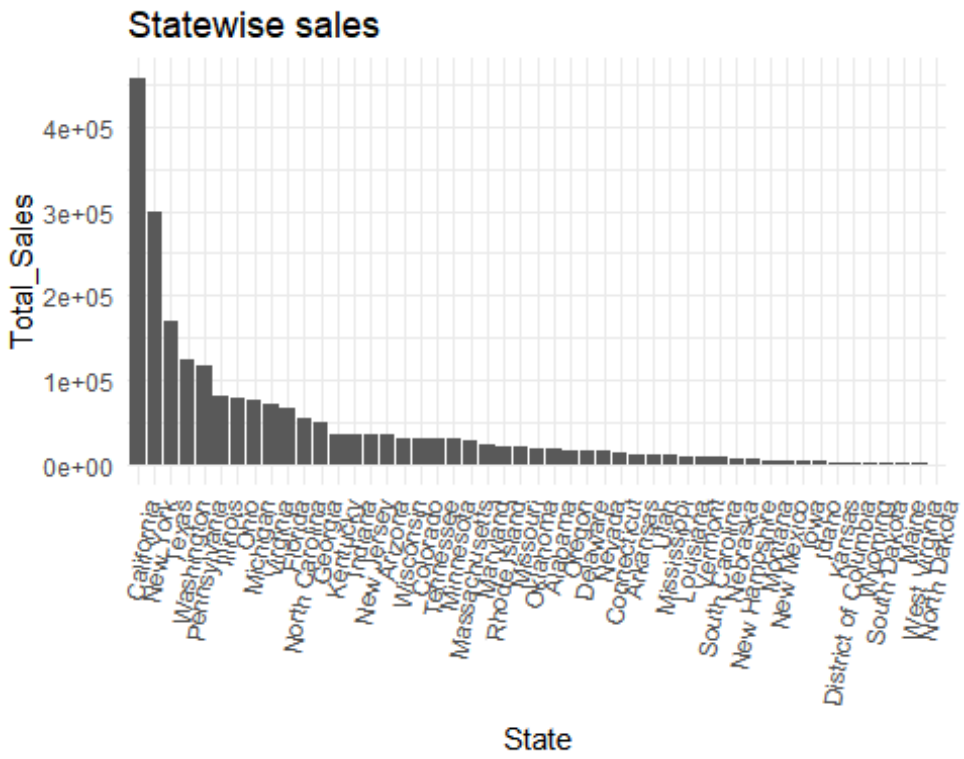
```
# drastic change in maximum sales amount, mean has around 8% change
# removing observations with >=10000 sales amount
# only 5 observations will get deleted

df = df %>% filter(Sales < 10000)

# deleting unwanted data sets
product_id = NULL
product_list = NULL
similar_id = NULL
```

```
# total countries in the data set
table(df$Country)
```

```
##
## United States
##          9989
```

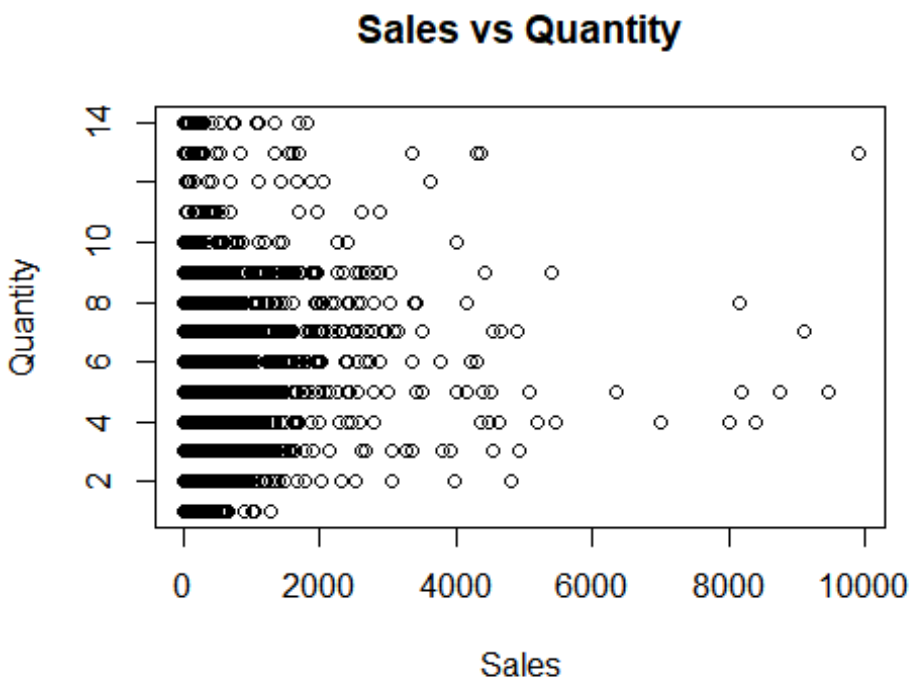
```
# only united states

# total states in the data set
```

```r
# visualising top sales contributors
df %>% group_by(State) %>% summarise(Total_Sales = sum(Sales)) %>%
        ggplot() + geom_bar(aes(reorder(State,-Total_Sales),Total_Sales),stat
= "identity")+
        theme_minimal() +
        theme(axis.text.x = element_text(angle = 80, hjust = 1)) +
        labs(title = 'Statewise sales',x = 'State')
```




```r
# California and New York are the top contributors

View(df %>% group_by(State) %>% summarise(N = n(),
                                  Sales_Max = max(Sales),
                                  Sales_min = min(Sales),
                                  Sales_Total = sum(Sales),
                                  Sales_mean = mean(Sales))
                                  %>% arrange(desc(Sales_Total)))

View(df %>% group_by(State,Category) %>% summarise(N = n(),
                                            Sales_Max = max(Sales),
                                            Sales_min = min(Sales),
                                            Sales_Total = sum(Sales),
                                            Sales_mean = mean(Sales))
                                          %>% arrange(State,desc(Sales_Total)))

# from the top 5 sales contributors only washington has high sales contributi
on by furniture category
```

```
# all other top 4 sales contributord are having technology as highest sales c
ontributor

# relation between quantity and sales
plot(df$Sales,df$Quantity,xlab = "Sales",ylab = "Quantity",main = "Sales vs Q
uantity")
```




```
## which category has maximum quantity sold
df %>% group_by(Category) %>% summarise(total_quantity = sum(Quantity),
                                        total_sales = sum(Sales)) %>% arrange
(desc(total_quantity))

## # A tibble: 3 x 3
##   Category        total_quantity total_sales
##   <chr>                    <dbl>       <dbl>
## 1 Office Supplies          22906     719047.
## 2 Furniture                 8028     742000.
## 3 Technology                6917     760316.

# Office supplies are having high quantity numbers where as technology sales
amount is higher
# may be technology items are more expensive
```

```
# pulling out year and month out of order date variable
library(lubridate)
```

```
## Warning: package 'lubridate' was built under R version 3.5.3

##
## Attaching package: 'lubridate'

## The following object is masked from 'package:base':
##
##     date

summary(df$`Order Date`)

##                   Min.                 1st Qu.                  Median
## "2014-01-03 00:00:00" "2015-05-23 00:00:00" "2016-06-26 00:00:00"
##                   Mean                 3rd Qu.                    Max.
## "2016-04-29 22:09:51" "2017-05-14 00:00:00" "2017-12-30 00:00:00"

df = df %>% mutate(month_year = format(as.Date(`Order Date`),"%Y-%m"))
```

```
# Exploring sub category vaiable
df %>% group_by(`Sub-Category`) %>% summarise(Freq = n(),
                                              Qty = sum(Quantity)) %>%
         arrange(desc(Qty))

## # A tibble: 17 x 3
##    `Sub-Category`  Freq   Qty
##    <chr>          <int> <dbl>
##  1 Binders         1523  5974
##  2 Paper           1370  5178
##  3 Furnishings      957  3563
##  4 Phones           889  3289
##  5 Storage          846  3158
##  6 Art              796  3000
##  7 Accessories      775  2976
##  8 Chairs           617  2356
##  9 Appliances       466  1729
## 10 Labels           364  1400
## 11 Tables           319  1241
## 12 Fasteners        217   914
## 13 Envelopes        254   906
## 14 Bookcases        228   868
## 15 Supplies         190   647
## 16 Machines         114   434
## 17 Copiers           64   218

# Papers and Binders are the top 2 items in terms of quantity sold
df %>% group_by(`Sub-Category`) %>% summarise(Freq = n(),
                                              Qty = sum(Quantity),
                                               Total_Sale = sum(Sales))
%>%
         arrange(desc(Total_Sale))
```

```
## # A tibble: 17 x 4
##    `Sub-Category`  Freq   Qty Total_Sale
##    <chr>          <int> <dbl>      <dbl>
##  1 Phones           889  3289    330007.
##  2 Chairs           617  2356    328449.
##  3 Storage          846  3158    223844.
##  4 Tables           319  1241    206966.
##  5 Binders         1523  5974    203413.
##  6 Accessories      775  2976    167380.
##  7 Machines         114   434    166600.
##  8 Bookcases        228   868    114880.
##  9 Appliances       466  1729    107532.
## 10 Copiers           64   218     96328.
## 11 Furnishings      957  3563     91705.
## 12 Paper           1370  5178     78479.
## 13 Supplies         190   647     46674.
## 14 Art              796  3000     27119.
## 15 Envelopes        254   906     16476.
## 16 Labels           364  1400     12486.
## 17 Fasteners        217   914      3024.
```

```
# sales is dominated by phones and chairs sub category
# checking the sales percentage share of each category

Sales_Share = df %>% group_by(Category) %>%
              summarise(Sales = sum(Sales))

Sales_Share %>% mutate(Share_Per = paste0(round(Sales/sum(Sales)*100,2),"%"))
%>%
         arrange(desc(Share_Per))
```

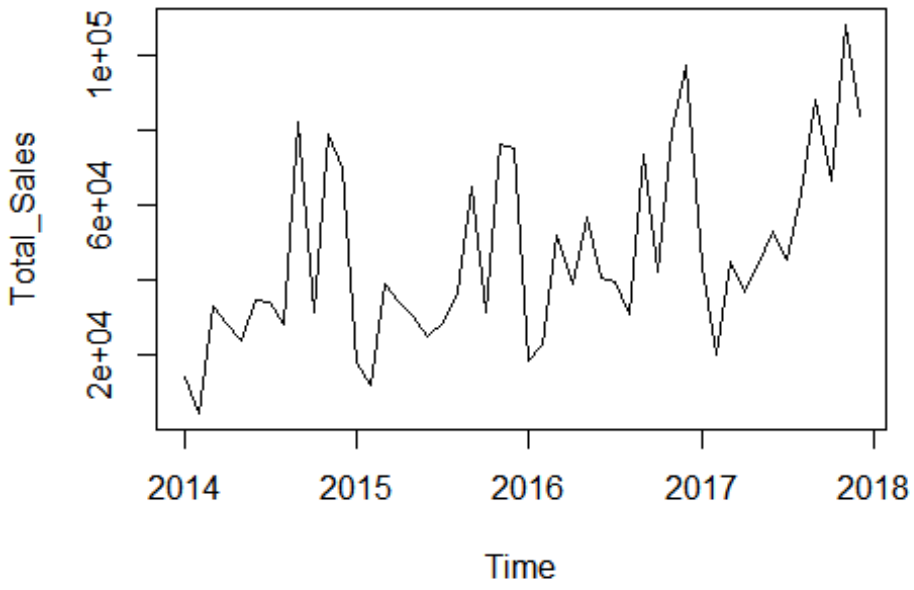
```
## # A tibble: 3 x 3
##   Category          Sales Share_Per
##   <chr>             <dbl> <chr>
## 1 Technology      760316. 34.23%
## 2 Furniture       742000. 33.4%
## 3 Office Supplies 719047. 32.37%
```

```
# all three categories seems to have quiet equal share
```

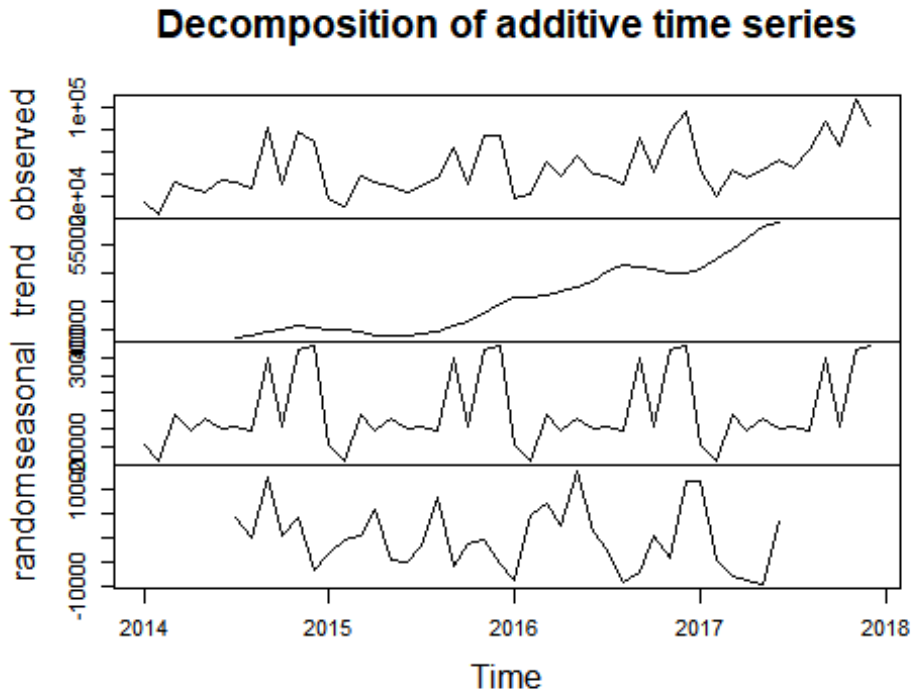
```
# Buidling time series model
# forecasting at united states level
ts_data = df %>%group_by(month_year) %>% summarise(Total_Sales = sum(Sales))
%>%
                arrange(month_year) %>% select(Total_Sales)

# converting data into ts class
ts_df = ts(ts_data,frequency = 12,start = c(2014,1))
end(ts_df)
```

```
## [1] 2017   12
```

```
plot.ts(ts_df)
```

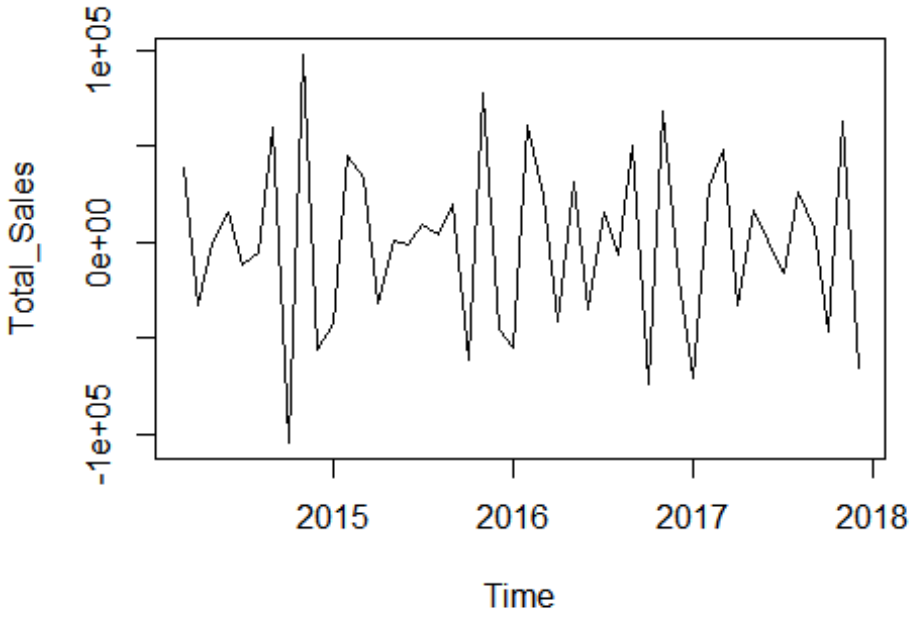



```
plot(decompose(ts_df))
```

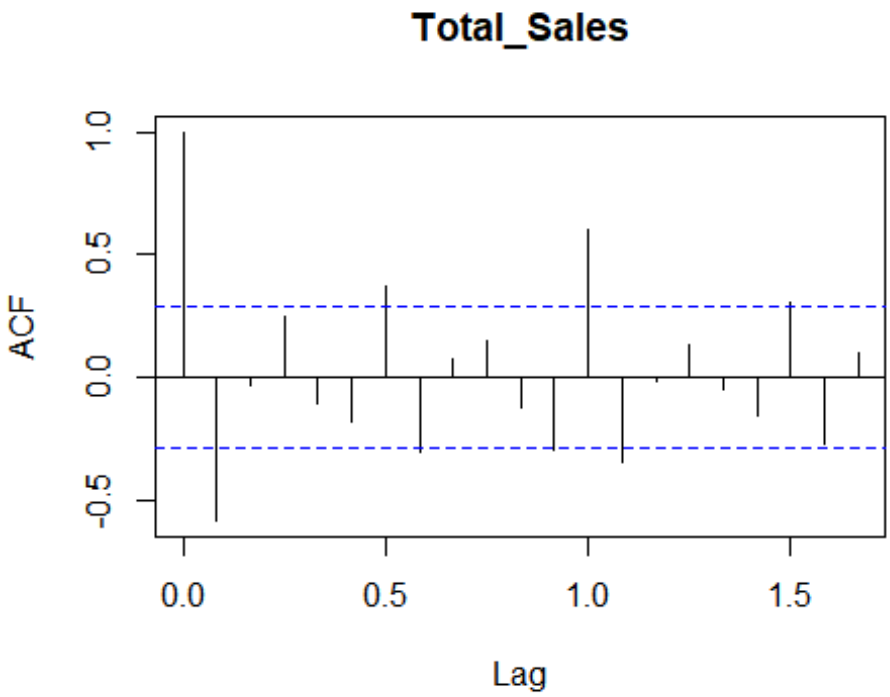


```
# statistical check for stationarity
adf.test(ts_df) #p-value is 0.07

##
##  Augmented Dickey-Fuller Test
##
## data:  ts_df
## Dickey-Fuller = -3.3304, Lag order = 3, p-value = 0.07796
## alternative hypothesis: stationary
```

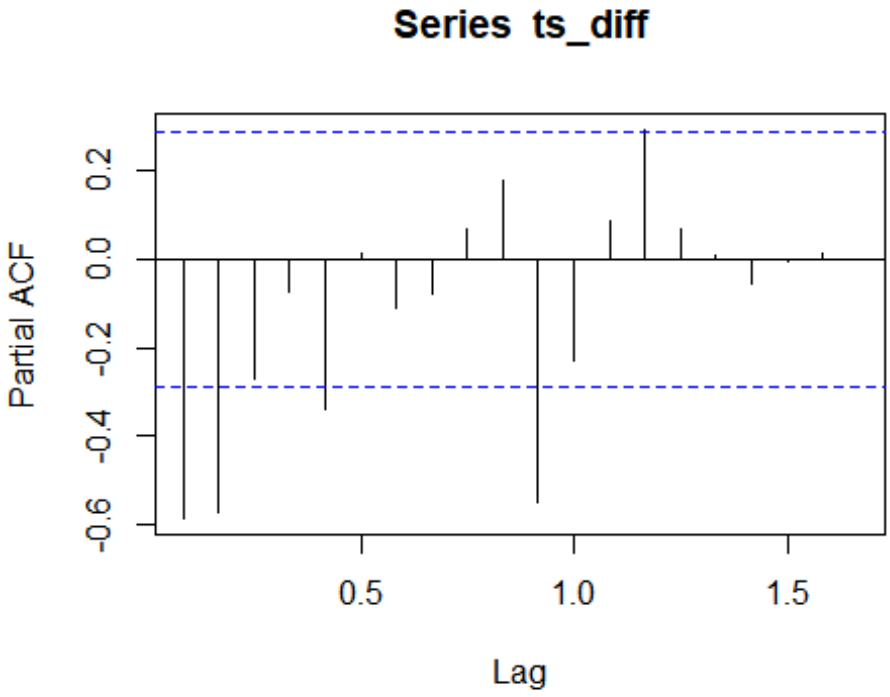
```
# differencing time series
ts_diff = diff(ts_df,differences = 2)
plot.ts(ts_diff)
```




```
adf.test(ts_diff)

## Warning in adf.test(ts_diff): p-value smaller than printed p-value

##
##  Augmented Dickey-Fuller Test
##
## data:  ts_diff
## Dickey-Fuller = -5.2651, Lag order = 3, p-value = 0.01
## alternative hypothesis: stationary

# time series looks stationary

# selecting p,d,q
acf(ts_diff, lag.max = 20)
```



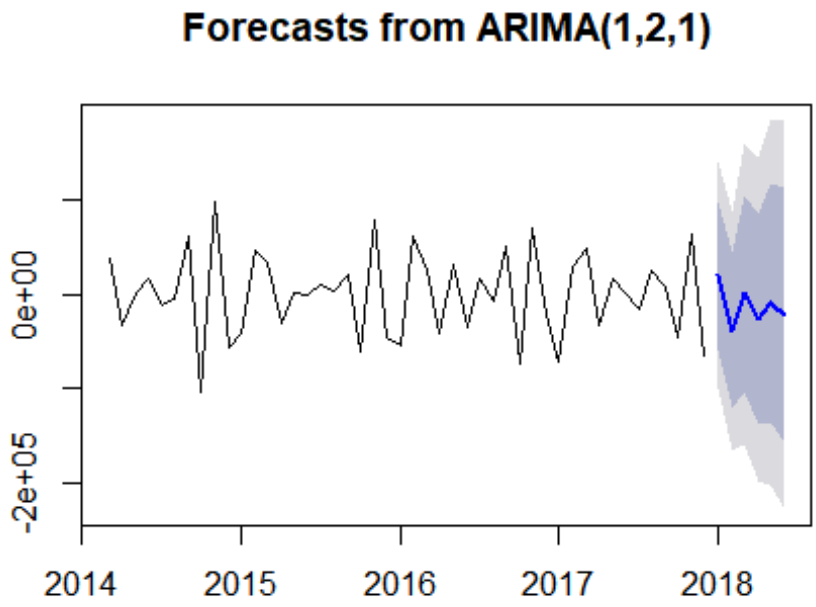

```r
pacf(ts_diff, lag.max = 20)
```



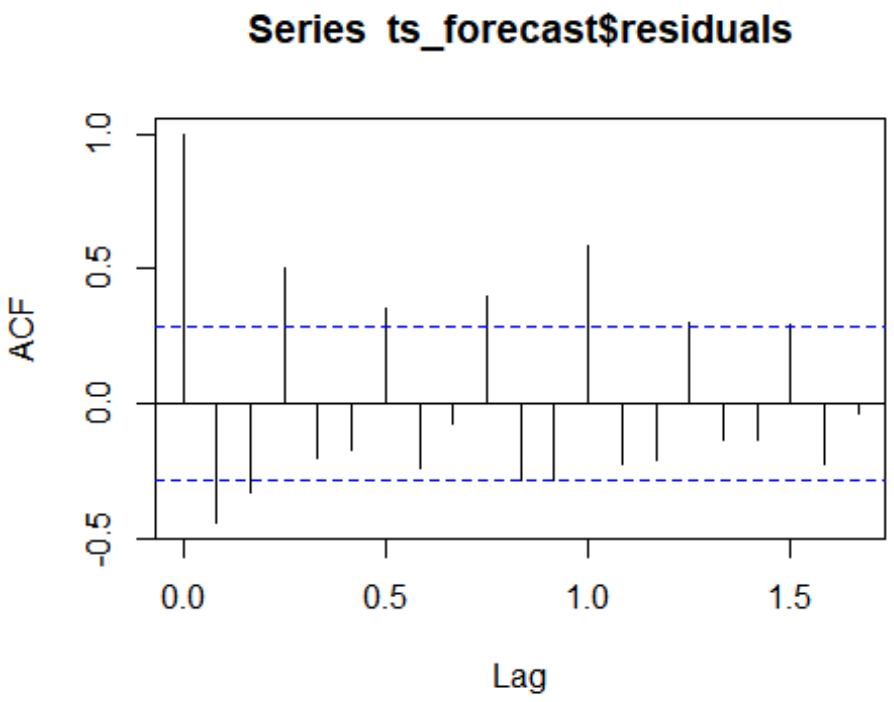
```r
# selecting p as 1 and q as 0
ts_model = arima(ts_diff,order = c(1,2,1))
ts_model
```

```
##
## Call:
## arima(x = ts_diff, order = c(1, 2, 1))
##
## Coefficients:
##           ar1      ma1
##       -0.6761  -1.0000
## s.e.   0.1125   0.0571
##
## sigma^2 estimated as 3.613e+09:  log likelihood = -549.32,  aic = 1104.64
```

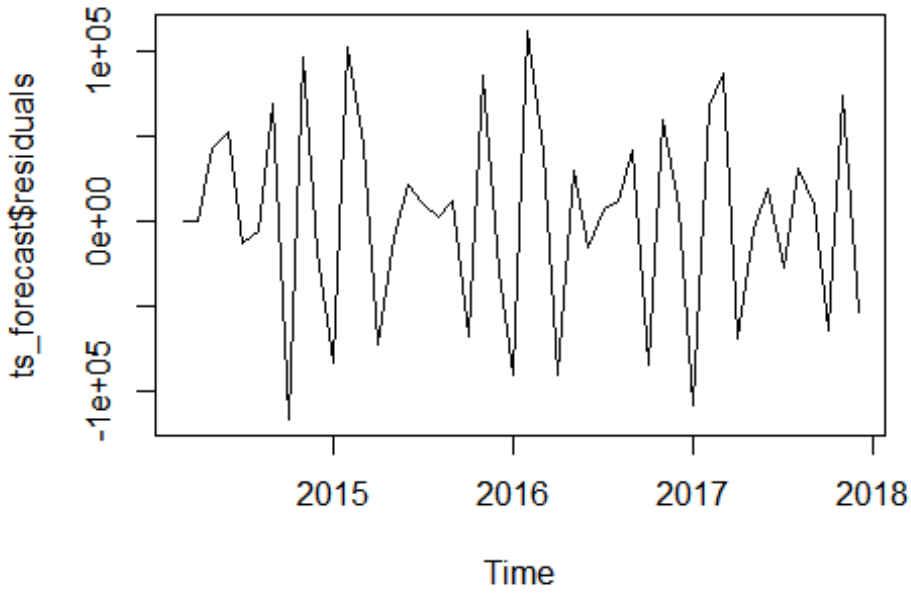
```r
# forecasting for next two years
ts_forecast = forecast:::forecast.Arima(ts_model,h=6)
plot(ts_forecast)
```



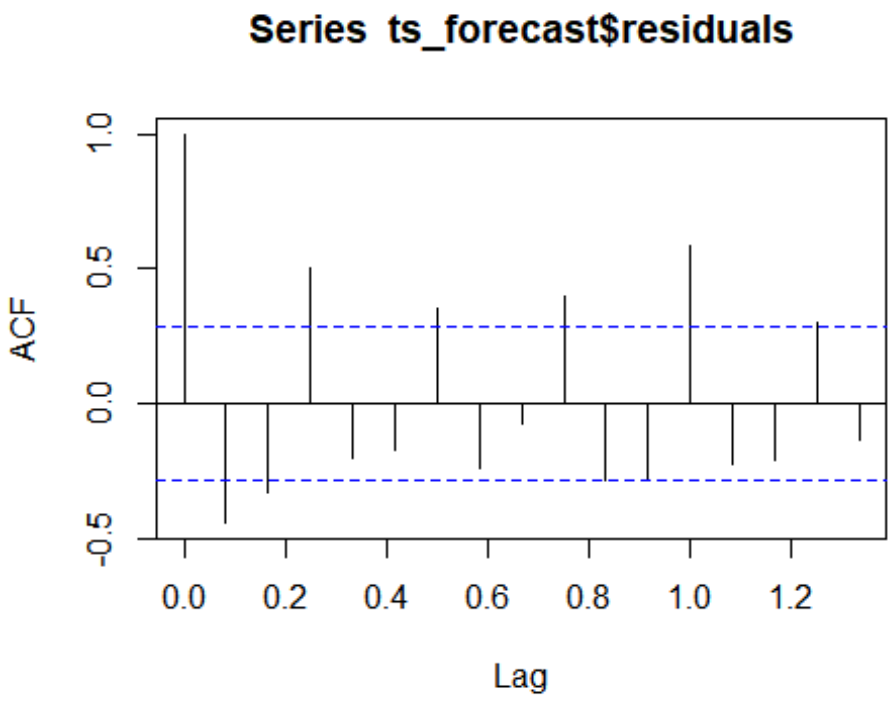
```r
# checking residuals
acf(ts_forecast$residuals, lag.max = 20)
```




```r
plot.ts(ts_forecast$residuals) # residuals looks random
```



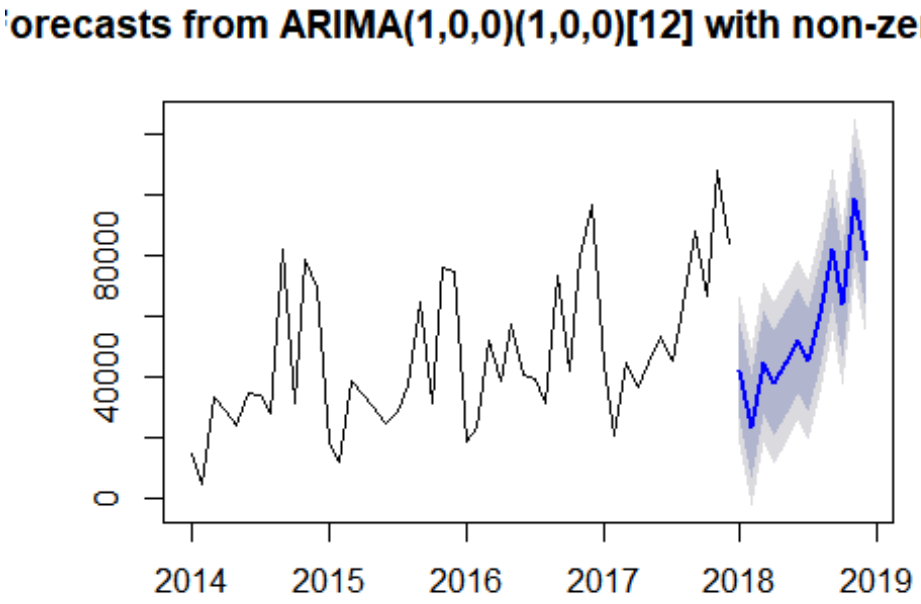

```r
acf(ts_forecast$residuals)
```




```
Box.test(ts_forecast$residuals,lag = 20,type = 'Ljung-Box')

##
##  Box-Ljung test
##
## data:  ts_forecast$residuals
## X-squared = 110.87, df = 20, p-value = 1.366e-14

# p value is < 0.05
# residuals are not independent
# forecasting needs improvement
# model is not performing well
```

```
# using auto.arima function
ts_auto_arima = auto.arima(ts_df,seasonal = TRUE,stationary = TRUE)
ts_auto_arima

## Series: ts_df
## ARIMA(1,0,0)(1,0,0)[12] with non-zero mean
##
## Coefficients:
##          ar1    sar1       mean
##       0.4011  0.8529  47439.420
## s.e.  0.1308  0.0582   9940.418
##
```

```
## sigma^2 estimated as 154065998:  log likelihood=-526.92
## AIC=1061.84   AICc=1062.77   BIC=1069.32

auto_arima_forecast = forecast:::forecast.Arima(ts_auto_arima,h=12)
plot(auto_arima_forecast)
```



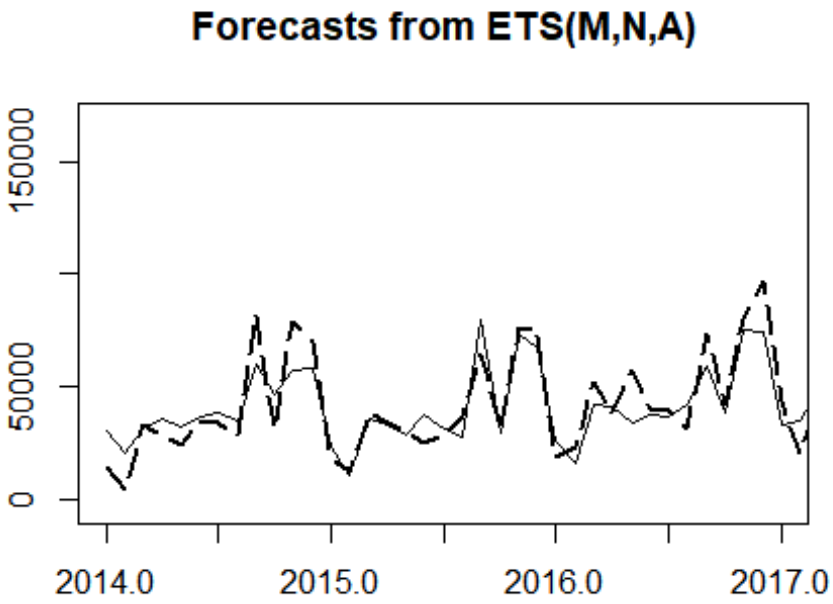

```
auto_arima_forecast
```

```
##          Point Forecast     Lo 80     Hi 80     Lo 95     Hi 95
## Jan 2018       42122.40 26215.358  58029.45 17794.677  66450.13
## Feb 2018       23346.14  6207.008  40485.28 -2865.902  49558.19
## Mar 2018       44870.34 27541.129  62199.55 18367.598  71373.08
## Apr 2018       37974.99 20615.388  55334.59 11425.770  64524.21
## May 2018       44667.47 27302.985  62031.96 18110.781  71224.16
## Jun 2018       52142.10 34776.829  69507.38 25584.208  78700.00
## Jul 2018       45574.47 28209.071  62939.87 19016.384  72132.56
## Aug 2018       60810.63 43445.209  78176.05 34252.511  87368.75
## Sep 2018       81919.33 64553.904  99284.75 55361.204 108477.45
## Oct 2018       63761.71 46396.287  81127.14 37203.587  90319.84
## Nov 2018       99048.46 81683.036 116413.88 72490.336 125606.58
## Dec 2018       78477.26 61111.831  95842.68 51919.131 105035.38
```

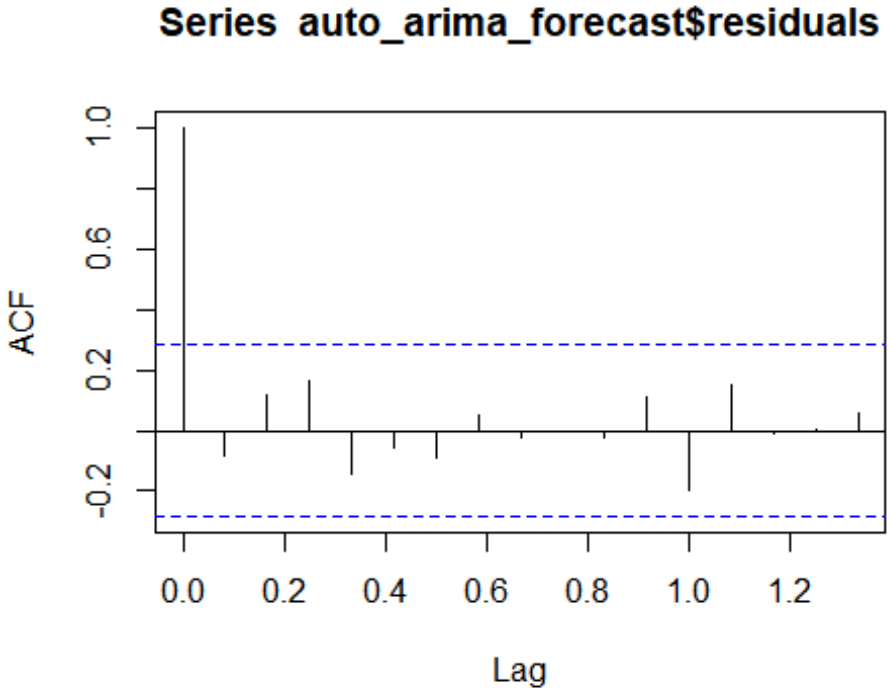
```
library(TSPred)
```

```
## Warning: package 'TSPred' was built under R version 3.5.3
```

```
# predicted vd actual graph
plotarimapred(ts_df,auto_arima_forecast$fitted,xlim = c(2014,2017))
```

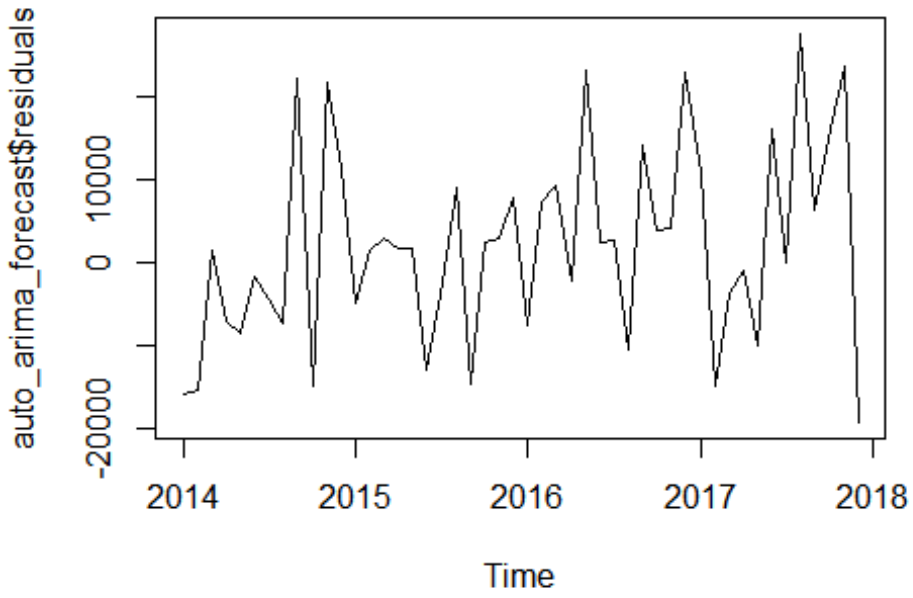



```r
# checking residuals auto corelation
acf(auto_arima_forecast$residuals)
```



```
# no auto corelation
```

```
# checking residuals
plot(auto_arima_forecast$residuals)
```




```
# residuals are random

Box.test(auto_arima_forecast$residuals,lag = 20,type = "Ljung-Box")

##
##  Box-Ljung test
##
## data:  auto_arima_forecast$residuals
## X-squared = 18.663, df = 20, p-value = 0.5438

# p value is 0.33 i.e. we can not reject null hypothesis
# we can say residuals are independent
# model is performing well
```

```
# mean squared error
mean(auto_arima_forecast$residuals)

## [1] 2105.651
```

```r
sqrt(mean(auto_arima_forecast$residuals))
```

```
## [1] 45.88737
```

```r
# checking mean absolute percentage error
mean((abs((ts_df-auto_arima_forecast$fitted)/ts_df))*100)
```

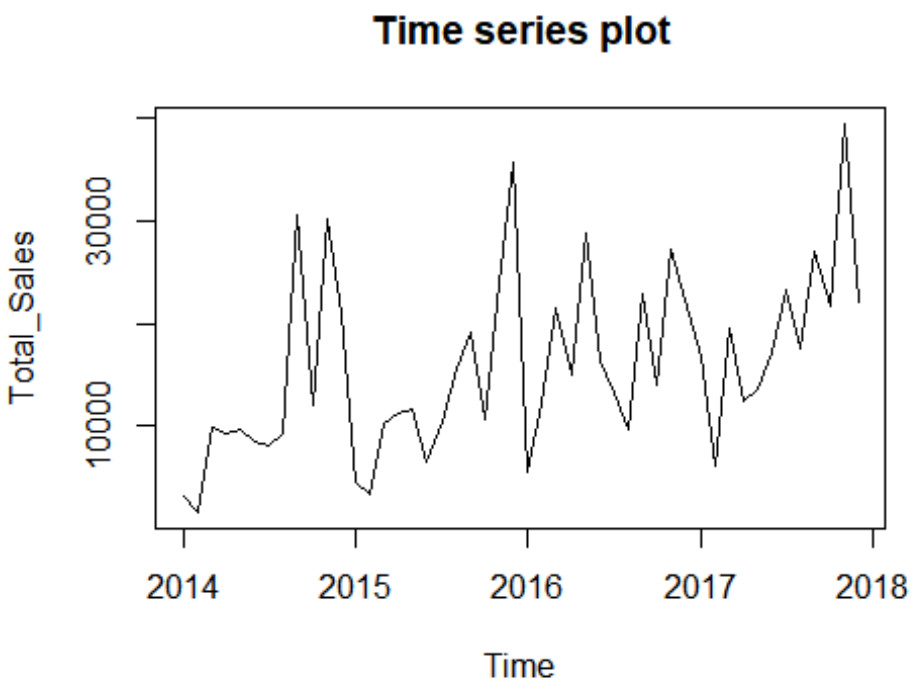
```
## [1] 28.95379
```

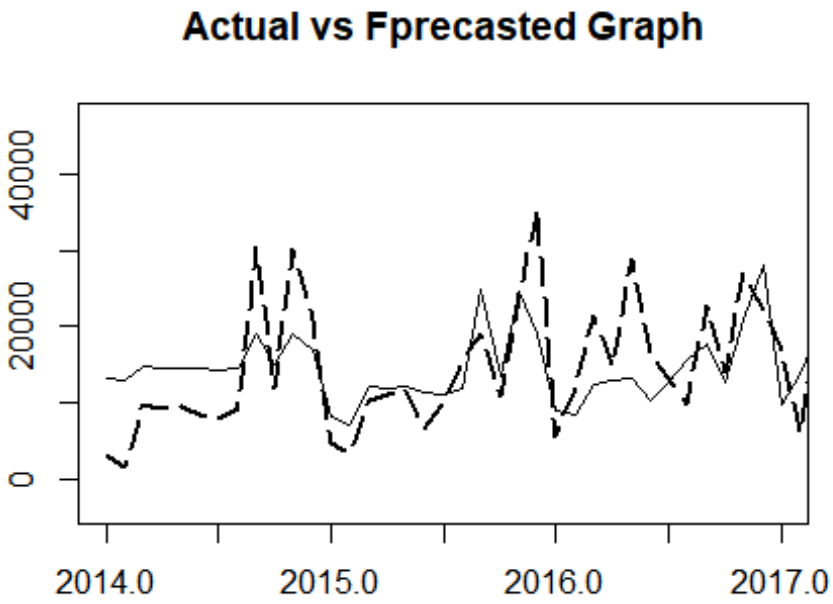
```r
# the model is off by 29%
# the model is performing well at (100-29) 71%
```

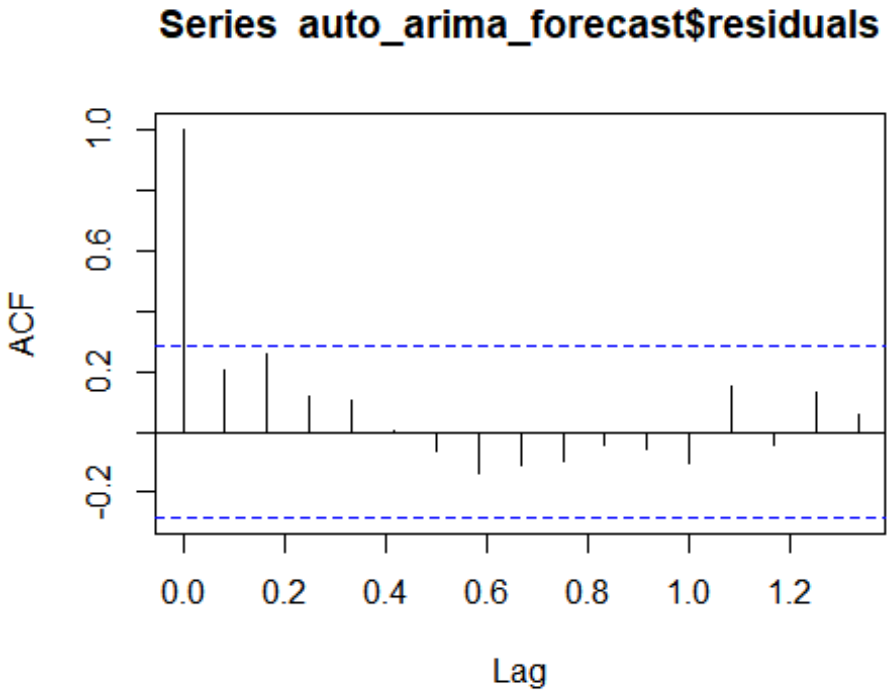
Creating forecast at a category level

```r
Category_Forecast = function(data,Catg)
{
  ts_data = data %>% filter(Category == Catg) %>%
          group_by(month_year) %>% summarise(Total_Sales = sum(Sales)) %>%
          arrange(month_year) %>% select(Total_Sales)

  # converting data into ts class
  ts_df = ts(ts_data,frequency = 12,start = c(2014,1))
  end(ts_df)

  print(plot.ts(ts_df,main="Time series plot"))
  #plot(decompose(ts_df))

  # statistical check for stationarity
  adf.test(ts_df)

  # checking with auto arima
  ts_auto_arima = auto.arima(ts_df,seasonal = TRUE,stationary = TRUE)

  auto_arima_forecast = forecast:::forecast.Arima(ts_auto_arima,h=12)
  #print(auto_arima_forecast)

  # predicted vd actual graph
  library(TSPred)
  plotarimapred(ts_df,auto_arima_forecast$fitted,xlim = c(2014,2017),
                main = "Actual vs Fprecasted Graph")

  # checking residuals auto corelation
  acf(auto_arima_forecast$residuals)

  print("checking stationarity of residuals")
  print(adf.test(auto_arima_forecast$residuals))

  # checking residuals
  plot(auto_arima_forecast$residuals,main="Residual plot",xlab="Residuals")
  # residuals are random
```

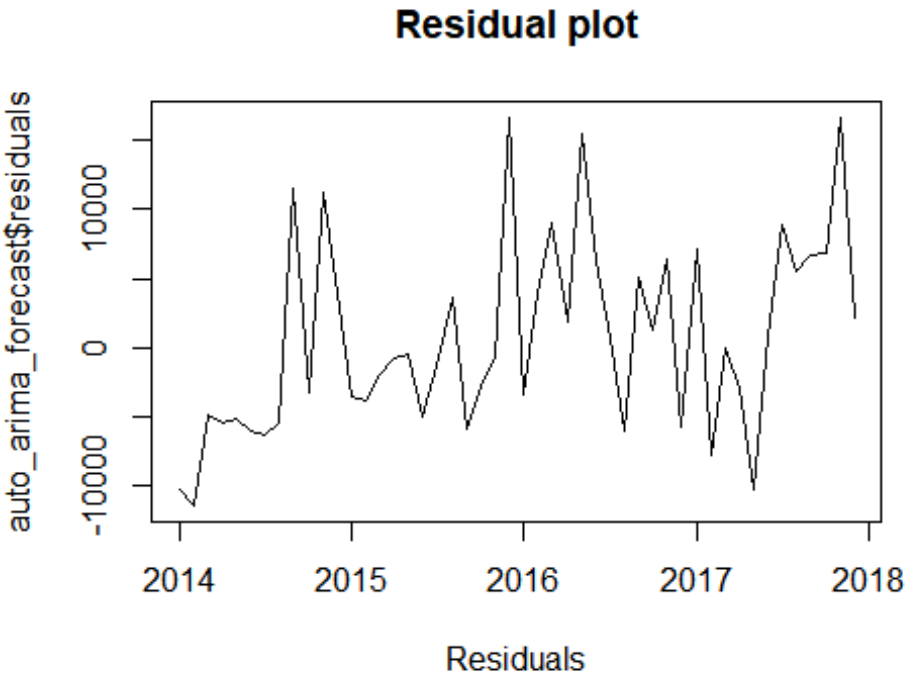
```
  print(plot(auto_arima_forecast))

  # mean squared error
  mean(auto_arima_forecast$residuals)
  sqrt(mean(auto_arima_forecast$residuals))

  # checking mean absolute percentage error
  print("Mean absolute percentage error")
  print(mean((abs((ts_df-auto_arima_forecast$fitted)/ts_df))*100))

  print(Box.test(auto_arima_forecast$residuals,lag = 20,type = "Ljung-Box"))

  forecasted_values = as.data.frame(auto_arima_forecast)

  return(forecasted_values)
}

Technology_forecast = Category_Forecast(data = df,Catg = "Technology")
```

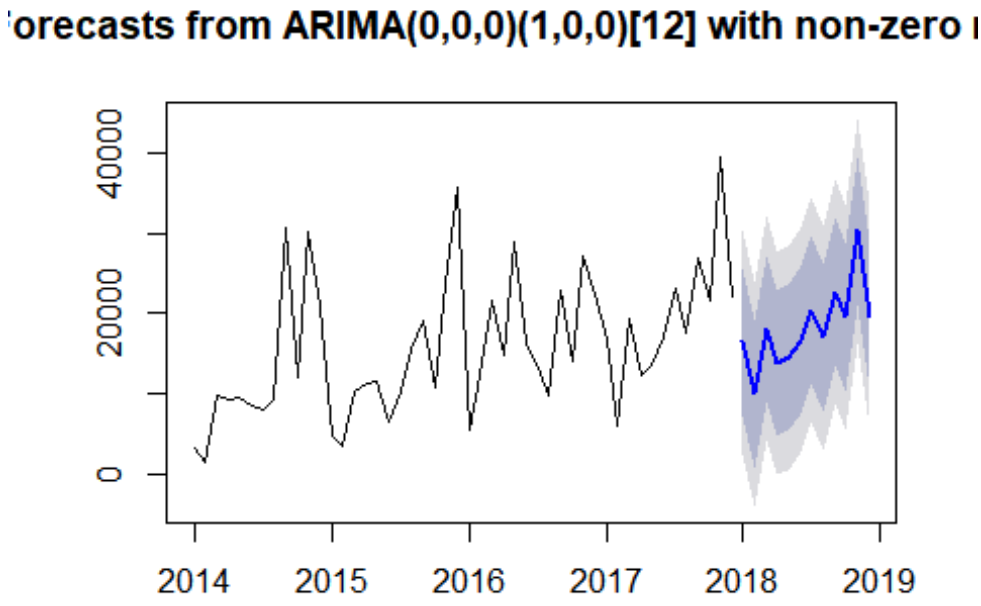



Actual vs Fprecasted Graph
40000
20000
0
2014.0
2015.0
2016.0
2017.0


Series auto_arima_forecast$residuals
ACF
1.0
0.6
0.2
-0.2
0.0
0.2
0.4
0.6
0.8
1.0
1.2
Lag

```
## [1] "checking stationarity of residuals"
##
##  Augmented Dickey-Fuller Test
##
## data:  auto_arima_forecast$residuals
## Dickey-Fuller = -2.6799, Lag order = 3, p-value = 0.3026
## alternative hypothesis: stationary
```


Residual plot
auto_arima_forecast$residuals
10000
0
-10000
2014
2015
2016
2017
2018
Residuals

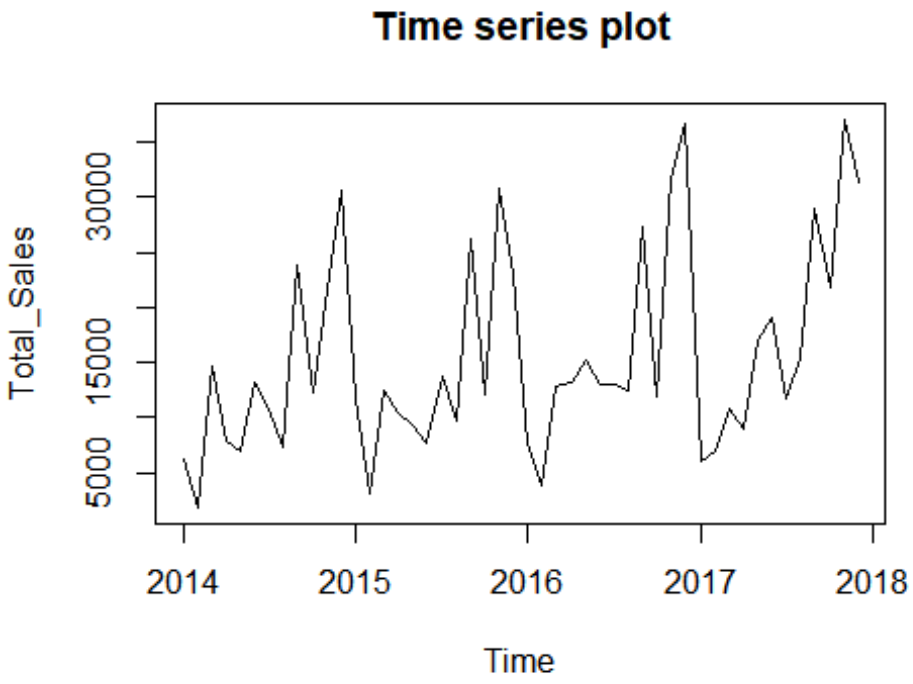

orecasts from ARIMA(0,0,0)(1,0,0)[12] with non-zero
40000
20000
0
2014
2015
2016
2017
2018
2019

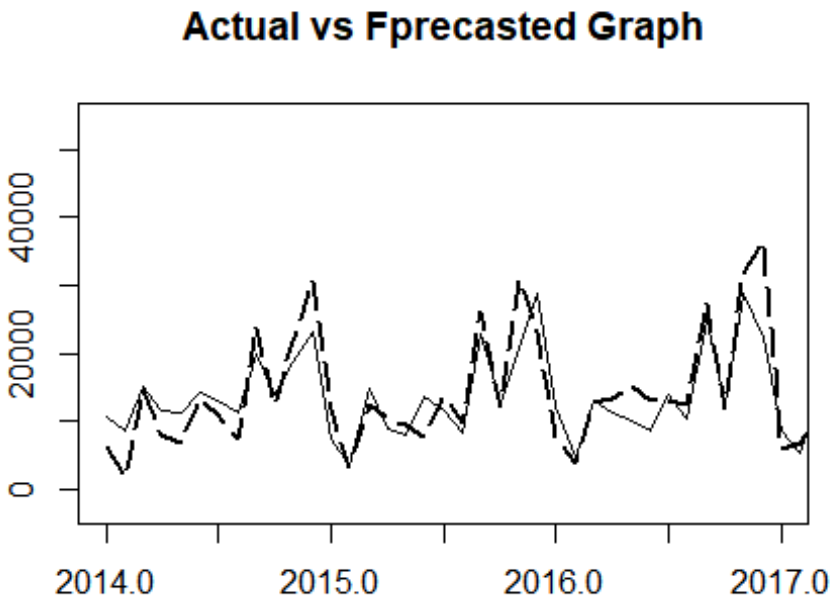
```
## [1] "Mean absolute percentage error"
## [1] 58.10326
##
##  Box-Ljung test
##
## data:  auto_arima_forecast$residuals
## X-squared = 21.791, df = 20, p-value = 0.3519
```

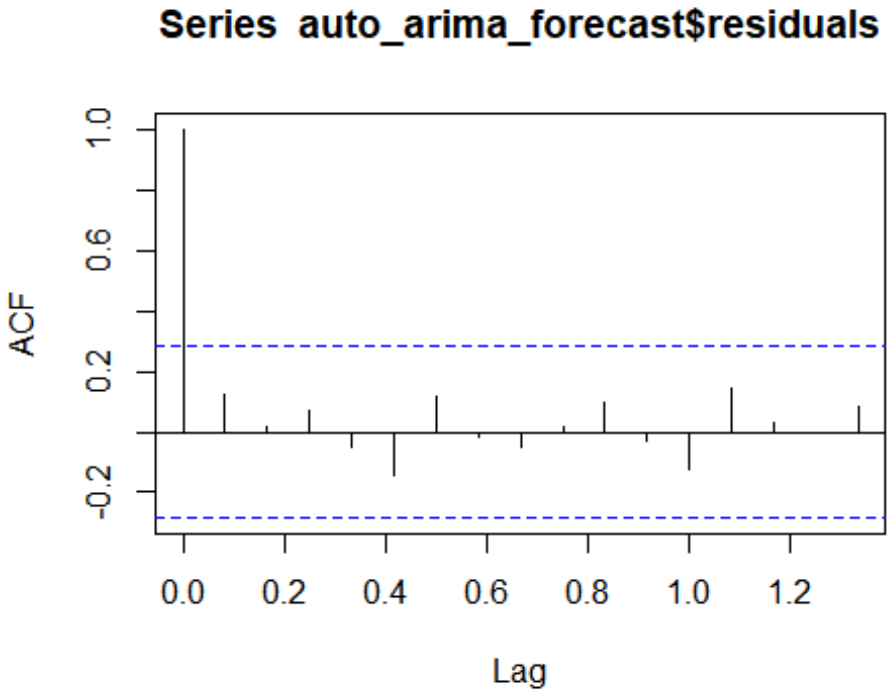
```
Technology_forecast
```

```
##          Point Forecast      Lo 80    Hi 80      Lo 95    Hi 95
## Jan 2018      16445.830  7338.2681 25553.39  2517.0160 30374.64
## Feb 2018       9910.238   802.6765 19017.80 -4018.5756 23839.05
## Mar 2018      18091.389  8983.8268 27198.95  4162.5747 32020.20
## Apr 2018      13790.537  4682.9754 22898.10  -138.2767 27719.35
## May 2018      14512.833  5405.2711 23620.39   584.0190 28441.65
## Jun 2018      16646.089  7538.5273 25753.65  2717.2752 30574.90
## Jul 2018      20399.695 11292.1334 29507.26  6470.8813 34328.51
## Aug 2018      16986.763  7879.2013 26094.33  3057.9492 30915.58
## Sep 2018      22678.342 13570.7796 31785.90  8749.5275 36607.16
## Oct 2018      19450.901 10343.3391 28558.46  5522.0870 33379.72
## Nov 2018      30294.550 21186.9879 39402.11 16365.7359 44223.36
## Dec 2018      19651.873 10544.3114 28759.44  5723.0593 33580.69
```

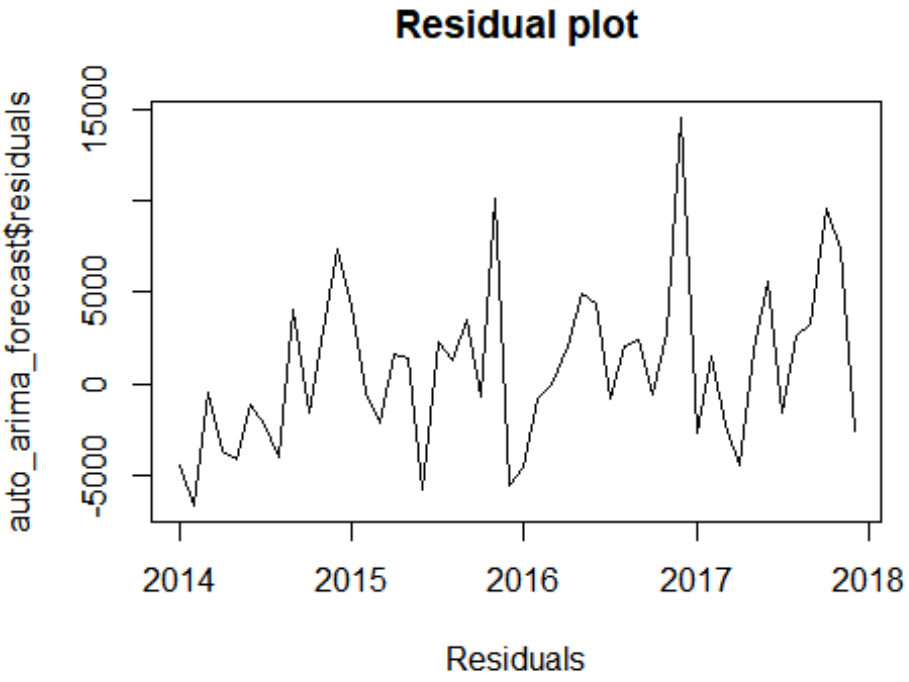
```
# forecasting for furniture product
Furniture_forecast = Category_Forecast(data = df,Catg = "Furniture")
```

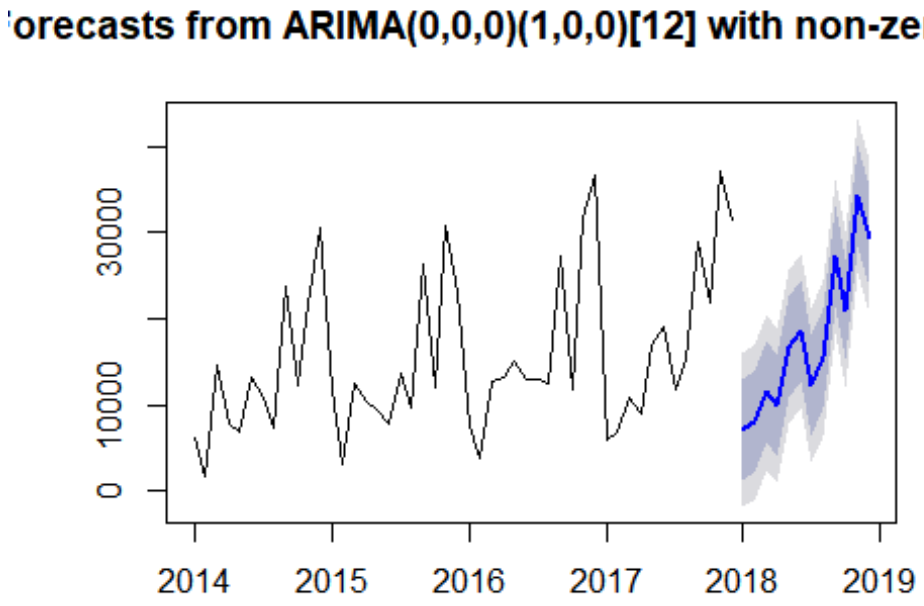

Time series plot
Total_Sales
5000
15000
30000
2014
2015
2016
2017
2018
Time


Actual vs Fprecasted Graph
0
20000
40000
2014.0
2015.0
2016.0
2017.0


Series auto_arima_forecast$residuals
ACF
1.0
0.6
0.2
-0.2
0.0
0.2
0.4
0.6
0.8
1.0
1.2
Lag

```
## [1] "checking stationarity of residuals"
##
##  Augmented Dickey-Fuller Test
##
## data:  auto_arima_forecast$residuals
## Dickey-Fuller = -3.7778, Lag order = 3, p-value = 0.02829
## alternative hypothesis: stationary
```

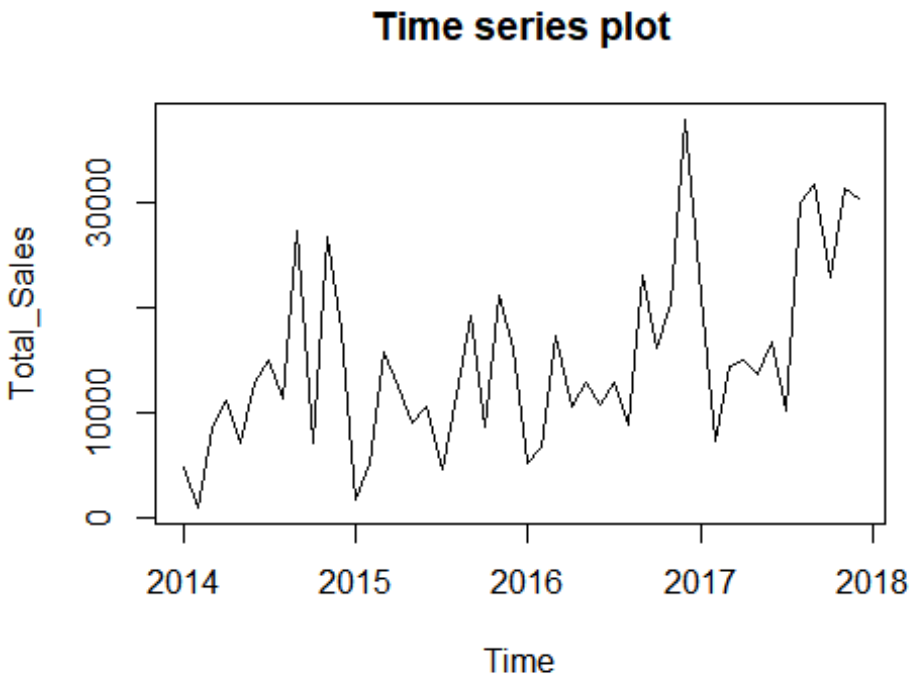

Residual plot
auto_arima_forecast$residuals
15000
5000
0
-5000
2014
2015
2016
2017
2018
Residuals

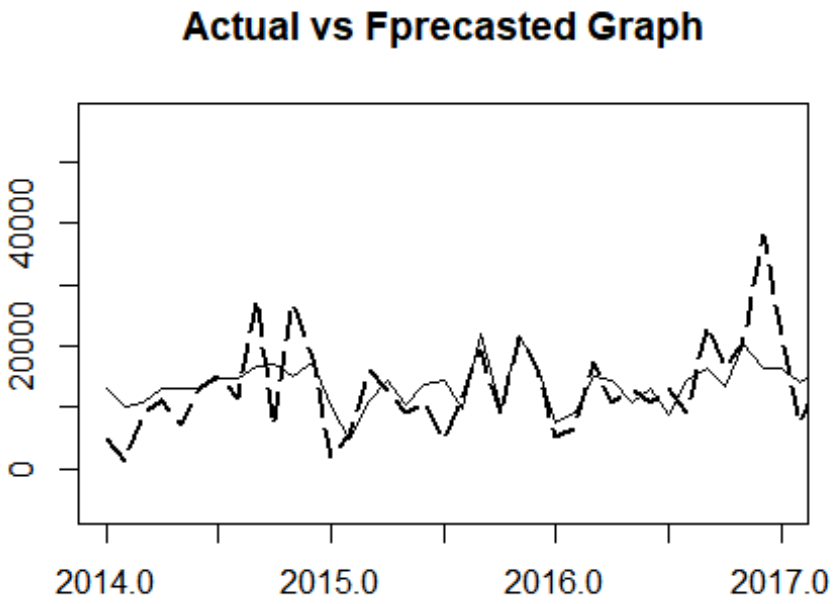

orecasts from ARIMA(0,0,0)(1,0,0)[12] with non-zero
30000
10000
0
2014
2015
2016
2017
2018
2019

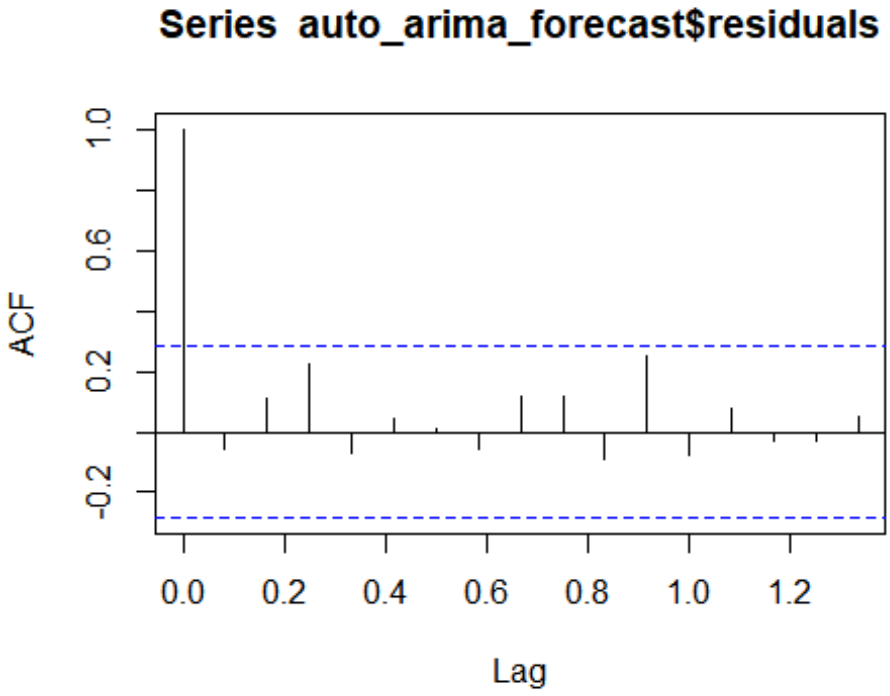
```
## [1] "Mean absolute percentage error"
## [1] 31.54591
##
##  Box-Ljung test
##
## data:  auto_arima_forecast$residuals
## X-squared = 8.683, df = 20, p-value = 0.9863

Furniture_forecast

##          Point Forecast     Lo 80    Hi 80      Lo 95    Hi 95
## Jan 2018       7180.888  1334.485 13027.29 -1760.4139 16122.19
## Feb 2018       7968.165  2121.762 13814.57  -973.1370 16909.47
## Mar 2018      11481.885  5635.482 17328.29  2540.5832 20423.19
## Apr 2018       9887.372  4040.969 15733.77   946.0698 18828.67
## May 2018      16772.928 10926.525 22619.33  7831.6260 25714.23
## Jun 2018      18562.485 12716.082 24408.89  9621.1835 27503.79
## Jul 2018      12284.232  6437.829 18130.63  3342.9301 21225.53
## Aug 2018      15450.467  9604.064 21296.87  6509.1656 24391.77
## Sep 2018      27304.775 21458.372 33151.18 18363.4729 36246.08
## Oct 2018      21071.392 15224.989 26917.80 12130.0904 30012.69
## Nov 2018      34309.786 28463.383 40156.19 25368.4846 43251.09
## Dec 2018      29380.721 23534.318 35227.12 20439.4187 38322.02
```

Office supplies forecast

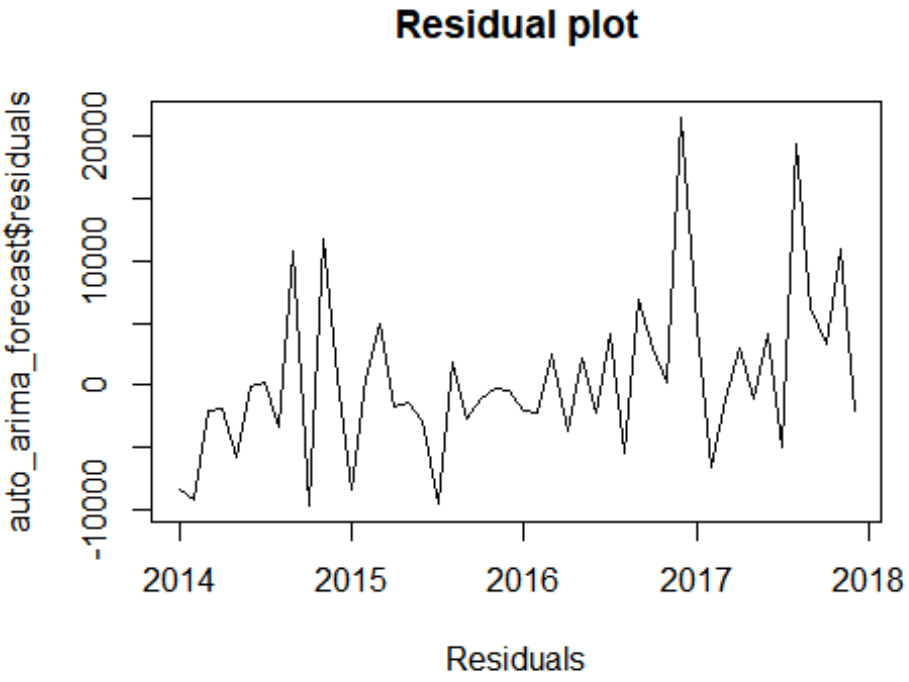
```
OS_forecast = Category_Forecast(df,"Office Supplies")
```

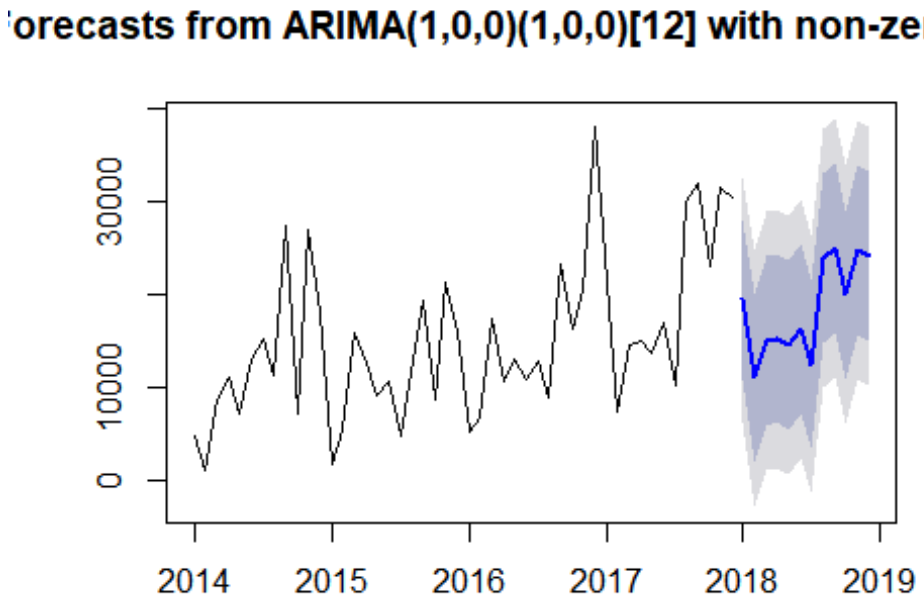

Time series plot
Total_Sales
0
10000
30000
2014
2015
2016
2017
2018
Time


Actual vs Fprecasted Graph
0
20000
40000
2014.0
2015.0
2016.0
2017.0


Series auto_arima_forecast$residuals
ACF
-0.2
0.2
0.6
1.0
0.0
0.2
0.4
0.6
0.8
1.0
1.2
Lag

```
## [1] "checking stationarity of residuals"
##
##  Augmented Dickey-Fuller Test
##
## data:  auto_arima_forecast$residuals
## Dickey-Fuller = -3.4232, Lag order = 3, p-value = 0.06366
## alternative hypothesis: stationary
```


Residual plot
auto_arima_forecast$residuals
-10000
0
10000
20000
2014
2015
2016
2017
2018
Residuals


orecasts from ARIMA(1,0,0)(1,0,0)[12] with non-zero
0
10000
30000
2014
2015
2016
2017
2018
2019

```
## [1] "Mean absolute percentage error"
## [1] 61.04858
##
##  Box-Ljung test
##
## data:  auto_arima_forecast$residuals
## X-squared = 12.274, df = 20, p-value = 0.9063
```

```
OS_forecast
```

```
##          Point Forecast     Lo 80    Hi 80     Lo 95    Hi 95
## Jan 2018       19468.12 10827.348 28108.90  6253.199 32683.05
## Feb 2018       11051.35  2009.235 20093.46 -2777.371 24880.07
## Mar 2018       15056.17  5976.833 24135.51  1170.522 28941.82
## Apr 2018       15319.69  6236.819 24402.56  1428.640 29210.73
## May 2018       14536.12  5452.914 23619.32   644.557 28427.68
## Jun 2018       16367.79  7284.550 25451.02  2476.176 30259.40
## Jul 2018       12510.86  3427.625 21594.10 -1380.751 26402.48
## Aug 2018       23966.18 14882.942 33049.42 10074.566 37857.79
## Sep 2018       25027.34 15944.097 34110.57 11135.721 38918.95
## Oct 2018       19906.72 10823.480 28989.96  6015.104 33798.33
## Nov 2018       24782.49 15699.254 33865.73 10890.878 38674.11
## Dec 2018       24184.00 15100.759 33267.24 10292.383 38075.61
```

At a category level auto.arima is not performing well. Apart from Furniture category, both technology and office supplies forecast has very high MAPE value. For both these categories the model is off by more than 50%.

As per our EDA we have seen that

```
## # A tibble: 3 x 3
##   Category          Sales Share_Per
##   <chr>             <dbl> <chr>
## 1 Technology      760316. 34.23%
## 2 Furniture       742000. 33.4%
## 3 Office Supplies 719047. 32.37%
```

In the overall sales amount share of Technology product is 34.23%, Furniture have 33.4% and Office supplies are having 32.37% share. From the overall sale s forecasted value if we want to find technology sales value then we can use the sales share percentage.