TIME SERIES FORCASTING ON SALES DATA

Loading the Package:

```
library(plyr)
library(dplyr)

## Warning: package 'dplyr' was built under R version 3.4.3

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:plyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

library(ggplot2)

## Warning: package 'ggplot2' was built under R version 3.4.4

#Loading the Data
dataNW<-read.csv("Demand Forecasting dataset.csv")
head(dataNW)

##   Row.ID       Order.ID Order.Date  Ship.Date      Ship.Mode Customer.ID
## 1      1 CA-2016-152156  11/8/2016 11/11/2016   Second Class    CG-12520
## 2      2 CA-2016-152156  11/8/2016 11/11/2016   Second Class    CG-12520
## 3      3 CA-2016-138688  6/12/2016  6/16/2016   Second Class    DV-13045
## 4      4 US-2015-108966 10/11/2015 10/18/2015 Standard Class    SO-20335
## 5      5 US-2015-108966 10/11/2015 10/18/2015 Standard Class    SO-20335
## 6      6 CA-2014-115812   6/9/2014  6/14/2014 Standard Class    BH-11710
##     Customer.Name   Segment       Country            City      State
## 1     Claire Gute  Consumer United States       Henderson   Kentucky
## 2     Claire Gute  Consumer United States       Henderson   Kentucky
## 3 Darrin Van Huff Corporate United States     Los Angeles California
## 4  Sean O'Donnell  Consumer United States Fort Lauderdale    Florida
## 5  Sean O'Donnell  Consumer United States Fort Lauderdale    Florida
## 6 Brosina Hoffman  Consumer United States     Los Angeles California
##   Postal.Code Region      Product.ID        Category Sub.Category    Sales
## 1       42420  South FUR-BO-10001798       Furniture    Bookcases 261.9600
## 2       42420  South FUR-CH-10000454       Furniture       Chairs 731.9400
## 3       90036   West OFF-LA-10000240 Office Supplies       Labels  14.6200
## 4       33311  South FUR-TA-10000577       Furniture       Tables 957.5775
## 5       33311  South OFF-ST-10000760 Office Supplies      Storage  22.3680
## 6       90032   West FUR-FU-10001487       Furniture  Furnishings  48.8600
##   Quantity Discount    Profit
## 1        2     0.00   41.9136
## 2        3     0.00  219.5820
## 3        2     0.00    6.8714
## 4        5     0.45 -383.0310
## 5        2     0.20    2.5164
## 6        7     0.00   14.1694
```

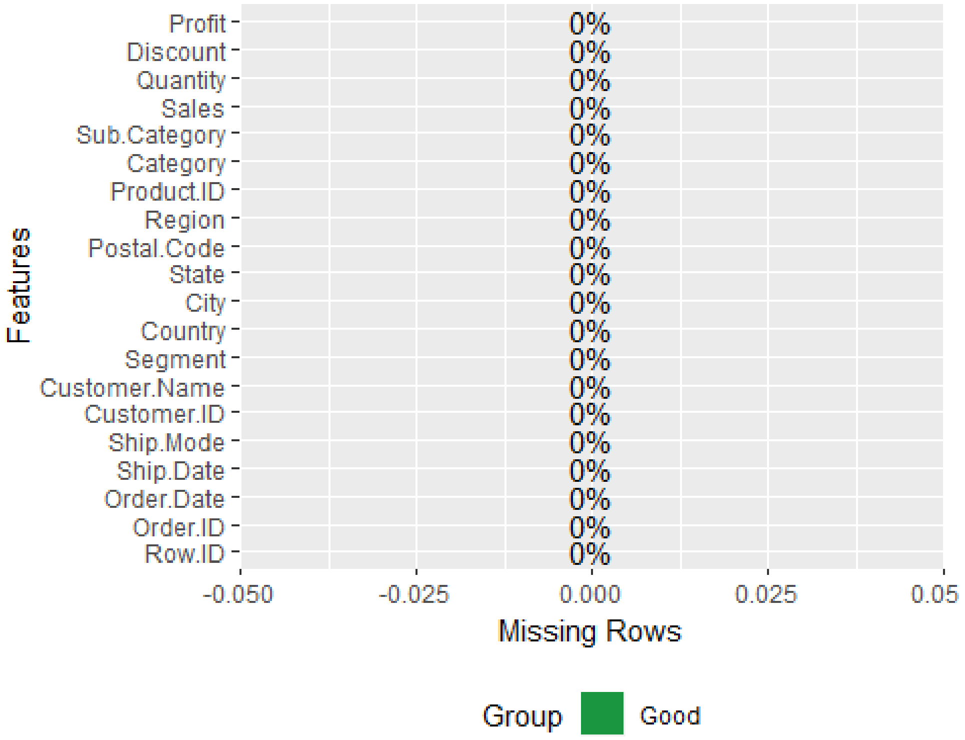
Data Explorationg:

1.Checking all the Variables

```
library(DataExplorer)

## Warning: package 'DataExplorer' was built under R version 3.4.4

plot_str(dataNW)
```

2. Checking Missing Values

```
plot_missing(dataNW)
```



```
str(dataNW)
```

```
## 'data.frame':    9994 obs. of  20 variables:
##  $ Row.ID       : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Order.ID     : Factor w/ 5009 levels "CA-2014-100006",..: 2501 2501
2297 4373 4373 202 202 202 202 202 ...
##  $ Order.Date   : Factor w/ 1237 levels "1/1/2017","1/10/2014",..: 305 305
836 94 94 922 922 922 922 922 ...
##  $ Ship.Date    : Factor w/ 1334 levels "1/1/2015","1/1/2016",..: 220 220
907 129 129 897 897 897 897 897 ...
##  $ Ship.Mode    : Factor w/ 4 levels "First Class",..: 3 3 3 4 4 4 4 4 4 4
...
##  $ Customer.ID  : Factor w/ 793 levels "AA-10315","AA-10375",..: 144 144
240 706 706 89 89 89 89 89 ...
##  $ Customer.Name: Factor w/ 793 levels "Aaron Bergman",..: 167 167 202 688
688 114 114 114 114 114 ...
##  $ Segment      : Factor w/ 3 levels "Consumer","Corporate",..: 1 1 2 1 1
1 1 1 1 1 ...
##  $ Country      : Factor w/ 1 level "United States": 1 1 1 1 1 1 1 1 1 1
...
##  $ City         : Factor w/ 531 levels "Aberdeen","Abilene",..: 195 195
267 154 154 267 267 267 267 267 ...
##  $ State        : Factor w/ 49 levels "Alabama","Arizona",..: 16 16 4 9 9
4 4 4 4 4 ...
##  $ Postal.Code  : int  42420 42420 90036 33311 33311 90032 90032 90032
90032 90032 ...
##  $ Region       : Factor w/ 4 levels "Central","East",..: 3 3 4 3 3 4 4 4
4 4 ...
##  $ Product.ID   : Factor w/ 1862 levels "FUR-BO-10000112",..: 13 56 947
320 1317 186 563 1762 795 438 ...
##  $ Category     : Factor w/ 3 levels "Furniture","Office Supplies",..: 1 1
2 1 2 1 2 3 2 2 ...
##  $ Sub.Category : Factor w/ 17 levels "Accessories",..: 5 6 11 17 15 10 3
14 4 2 ...
##  $ Sales        : num  262 731.9 14.6 957.6 22.4 ...
##  $ Quantity     : int  2 3 2 5 2 7 4 6 3 5 ...
##  $ Discount     : num  0 0 0 0.45 0.2 0 0 0.2 0.2 0 ...
##  $ Profit       : num  41.91 219.58 6.87 -383.03 2.52 ...
```

3. Changing Order date and Shipping Date into “Date” format

```
library(lubridate)

## Warning: package 'lubridate' was built under R version 3.4.3

##
## Attaching package: 'lubridate'

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

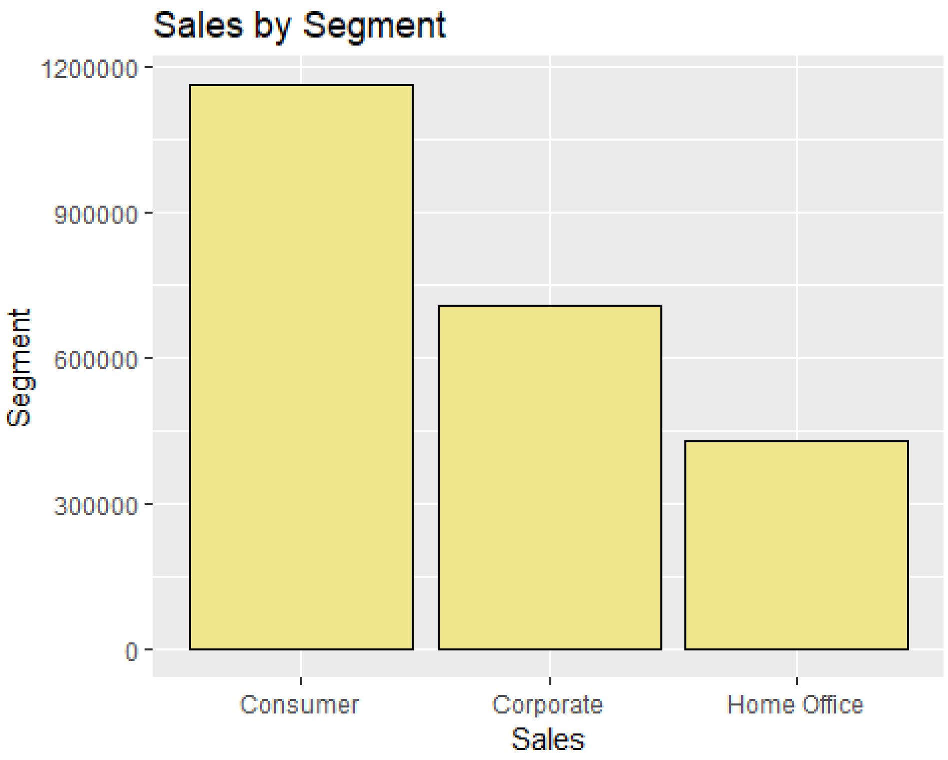
```
dataNW$Order.Date<-as.Date(dataNW$Order.Date,'%m/%d/%Y')
dataNW$Ship.Date<-as.Date(dataNW$Ship.Date,'%m/%d/%Y')
```

4. Data Visualization to check the Highest Sales with respect to Segment

```
segment_sale = aggregate(Sales ~ Segment, data = dataNW, sum)
segment_sale
```

```
##       Segment     Sales
## 1    Consumer 1161401.3
## 2   Corporate  706146.4
## 3 Home Office  429653.1
```

```
ggplot(segment_sale, aes(x = Segment, y = Sales)) + geom_bar(stat =
"identity", fill = 'khaki', colour = 'black') + ggtitle("Sales by Segment") +
labs(y = "Segment", x =
"Sales")
```



We can that Sales in Consumer Sector is Highest

Time Series Forecasting

1. Segment- Counsumer:

Extracting the Order Date and Sales filtered with Segment- Consumer.

```
library(dplyr)
Consumer<-dataNW %>%
  select(Order.Date, Sales) %>%
  filter(dataNW$Segment == "Consumer")

## Warning: package 'bindrcpp' was built under R version 3.4.3

## Warning: `as_dictionary()` is soft-deprecated as of rlang 0.3.0.
## Please use `as_data_pronoun()` instead
## This warning is displayed once per session.

## Warning: `new_overscope()` is soft-deprecated as of rlang 0.2.0.
## Please use `new_data_mask()` instead
## This warning is displayed once per session.

## Warning: The `parent` argument of `new_data_mask()` is deprecated.
## The parent of the data mask is determined from either:
##
##   * The `env` argument of `eval_tidy()`
##   * Quosure environments when applicable
## This warning is displayed once per session.

## Warning: `overscope_clean()` is soft-deprecated as of rlang 0.2.0.
## This warning is displayed once per session.
```

Displaying the data order by Order_Date:

```
counsumer<-Consumer[order(as.Date(Consumer$Order.Date, "%m/%d/%Y"),
decreasing = FALSE),]
tail(counsumer)

##      Order.Date   Sales
## 4034 2017-12-29 300.980
## 4035 2017-12-29 258.750
## 504  2017-12-30 323.136
## 505  2017-12-30  90.930
## 506  2017-12-30  52.776
## 2710 2017-12-30   3.024
```

Adding Year and Month

```
counsumer$year <- format(as.Date(counsumer$Order.Date), "%Y")
counsumer$month <- format(as.Date(counsumer$Order.Date), "%m")

head(counsumer)

##      Order.Date  Sales year month
## 4183 2014-01-03 16.448 2014    01
## 967  2014-01-05 19.536 2014    01
## 2841 2014-01-06 19.440 2014    01
## 4041 2014-01-07 76.728 2014    01
## 4042 2014-01-07 10.430 2014    01
## 337  2014-01-09  9.344 2014    01
```

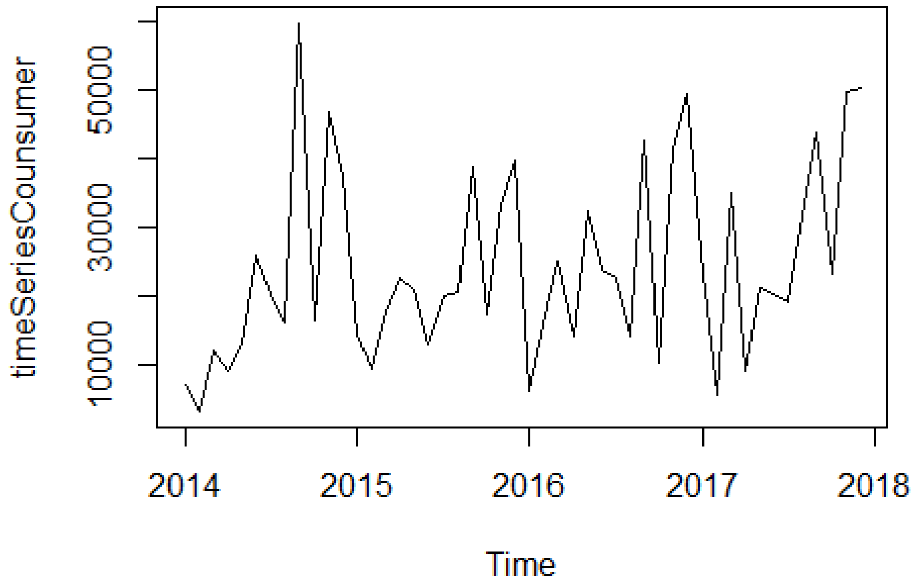
Using SQL Package to select year and month and the total sales order by year and month

```
library(sqldf)

## Warning: package 'sqldf' was built under R version 3.4.4

## Loading required package: gsubfn

## Warning: package 'gsubfn' was built under R version 3.4.4

## Loading required package: proto

## Warning: package 'proto' was built under R version 3.4.4

## Loading required package: RSQLite

## Warning: package 'RSQLite' was built under R version 3.4.4

counsumer<-sqldf("select year, month, sum(Sales) as Month_Sales from
counsumer group by year, month order by year, month")
tail(counsumer)

##    year month Month_Sales
## 43 2017    07    18991.37
## 44 2017    08    31629.14
## 45 2017    09    43857.20
## 46 2017    10    23194.24
## 47 2017    11    49790.06
## 48 2017    12    50232.46
```

Analysis- Total sum of Sales in December 2017 is $50232.46

Changing the data into time series-

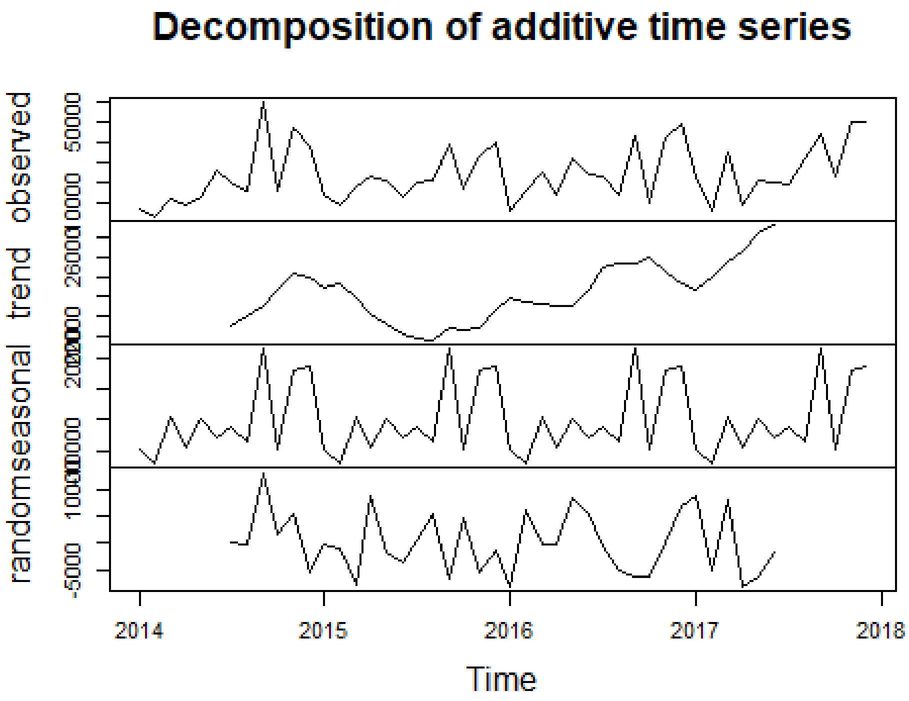
```
timeSeriesCounsumer<-ts(counsumer$Month_Sales,
frequency=12,start=c(2014,1),end=c(2017,12))
plot(timeSeriesCounsumer)
```



```
head(timeSeriesCounsumer)

## [1]  6927.817  3167.854 11869.304  9108.612 12819.163 25923.748
```

There is Seasonality on the time series

```
decompose_consumer<-decompose(timeSeriesCounsumer)
plot(decompose_consumer)
```



TIME SERIES FORCASTING

Simple Smoothing

For next 2 years:

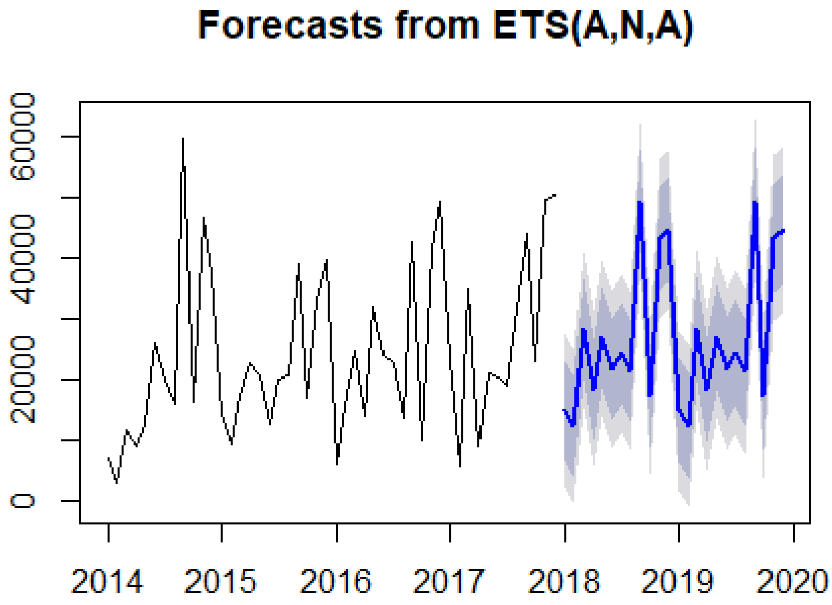
```
library(forecast)

## Warning: package 'forecast' was built under R version 3.4.3

##
## Attaching package: 'forecast'

## The following object is masked from 'package:ggplot2':
##
##     autolayer

simple_smoothing1 <- forecast(timeSeriesCounsumer)
summary(simple_smoothing1)

##
## Forecast method: ETS(A,N,A)
##
## Model Information:
## ETS(A,N,A)
##
## Call:
##  ets(y = object, lambda = lambda, allow.multiplicative.trend =
allow.multiplicative.trend)
## 
##   Smoothing parameters:
##     alpha = 0.0894 
##     gamma = 1e-04 
## 
##   Initial states:
##     l = 22511.2417 
##     s=17756.47 16285.89 -9597.991 22523.04 -5569.22 -2499.559
##            -5228.508 35.49 -8440.917 1294.956 -14603 -11956.65
## 
##   sigma:  6417.444
## 
##      AIC     AICc      BIC 
## 1057.428 1072.428 1085.496 
## 
## Error measures:
##                     ME     RMSE      MAE       MPE     MAPE      MASE
## Training set 1031.312 6417.444 5317.537 -7.274554 29.98308 0.6618447
##                     ACF1
## Training set -0.04539999
## 
## Forecasts:
##          Point Forecast     Lo 80    Hi 80      Lo 95    Hi 95
## Jan 2018       14981.81  6757.521 23206.09  2403.8474 27559.77
## Feb 2018       12334.80  4077.697 20591.90  -293.3479 24962.94
## Mar 2018       28231.94 19942.149 36521.72 15553.8014 40910.07
## Apr 2018       18496.60 10174.255 26818.94  5768.6730 31224.53
## May 2018       26972.85 18618.079 35327.63 14195.3285 39750.38
## Jun 2018       21710.60 13323.520 30097.68  8883.6693 34537.53
## Jul 2018       24438.23 16018.969 32857.49 11562.0820 37314.37
## Aug 2018       21369.85 12918.536 29821.17  8444.6784 34295.03
## Sep 2018       49461.04 40977.786 57944.30 36487.0225 62435.06
## Oct 2018       17341.01  8825.935 25856.08  4318.3275 30363.69
## Nov 2018       43224.85 34678.082 51771.62 30153.6947 56296.01
## Dec 2018       44695.30 36116.874 53273.72 31575.7311 57814.87
## Jan 2019       14981.81  6371.917 23591.70  1814.1171 28149.50
## Feb 2019       12334.80  3693.558 20976.04  -880.8389 25550.44
## Mar 2019       28231.94 19559.457 36904.41 14968.5243 41495.35
## Apr 2019       18496.60  9792.995 27200.20  5185.5850 31807.61
## May 2019       26972.85 18238.233 35707.47 13614.4053 40331.30
## Jun 2019       21710.60 12945.075 30476.12  8304.8869 35116.31
## Jul 2019       24438.23 15641.908 33234.55 10985.4169 37891.04
## Aug 2019       21369.85 12542.844 30196.86  7870.1074 34869.60
## Sep 2019       49461.04 40603.449 58318.63 35914.5231 63007.56
## Oct 2019       17341.01  8452.937 26229.08  3747.8774 30934.14
## Nov 2019       43224.85 34306.411 52143.30 29585.2720 56864.44
## Dec 2019       44695.30 35746.517 53644.08 31009.3188 58381.28
```

December 2017 Total Sale in December is 50232.46

After 1 year, forecasted sale would be 44695.30

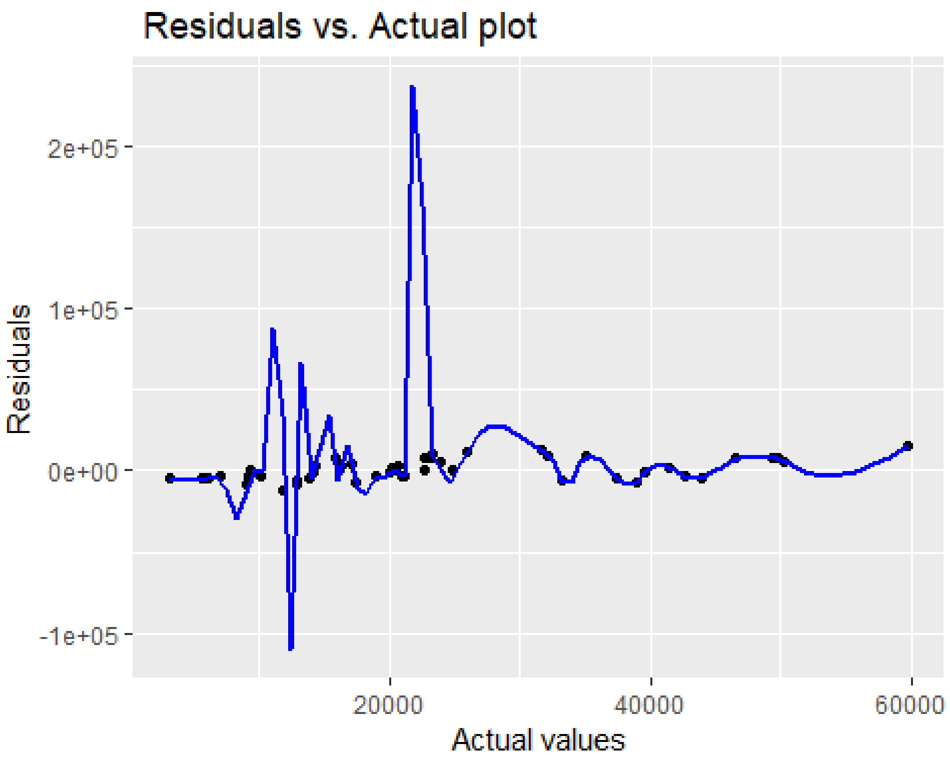
50232.46-44695.30/50232.46= 11%

```
accuracy(simple_smoothing1)

##                    ME     RMSE      MAE       MPE     MAPE      MASE
## Training set 1031.312 6417.444 5317.537 -7.274554 29.98308 0.6618447
##                     ACF1
## Training set -0.04539999

plot(simple_smoothing1)
```



Plot Between Actual and Residual:

```
library(ggplot2)
qplot(y = simple_smoothing1$residuals, x = timeSeriesCounsumer,
      ylab = "Residuals", xlab = "Actual values",
      main = " Residuals vs. Actual plot") +
  stat_smooth(method = "loess", span = 0.1, colour = I("blue"), se = FALSE)

## Don't know how to automatically pick scale for object of type ts.
Defaulting to continuous.
## Don't know how to automatically pick scale for object of type ts.
Defaulting to continuous.

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : Chernobyl! trL>n 48
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : Chernobyl! trL>n 48

## Warning in sqrt(sum.squares/one.delta): NaNs produced
```



Since the sales of Cousmer might fall down 1 year later- Lets check the forecast of Cosunmer in the three 3 Regions :

FOR SOUTH

```
library(dplyr)
Consumer_South<-dataNW %>%
  select(Order.Date, Sales) %>%
  filter(dataNW$Segment == "Consumer",dataNW$Region=="South")

tail(Consumer_South)

##     Order.Date   Sales
## 833 2017-07-16 242.352
## 834 2014-11-06  43.680
## 835 2014-08-12  14.040
## 836 2014-08-12 272.610
## 837 2015-09-06  85.980
## 838 2014-01-21  25.248
```

```
Consumer_South<-Consumer_South[order(as.Date(Consumer_South$Order.Date,
"%m/%d/%Y"), decreasing = FALSE),]
tail(Consumer_South)
```

```
##     Order.Date   Sales
## 771 2017-12-25  120.00
## 641 2017-12-29 1207.84
## 642 2017-12-29   12.53
## 643 2017-12-29   34.58
## 644 2017-12-29  300.98
## 645 2017-12-29  258.75
```

```
Consumer_South$year <- format(as.Date(Consumer_South$Order.Date), "%Y")
Consumer_South$month <- format(as.Date(Consumer_South$Order.Date), "%m")

head(Consumer_South)
```

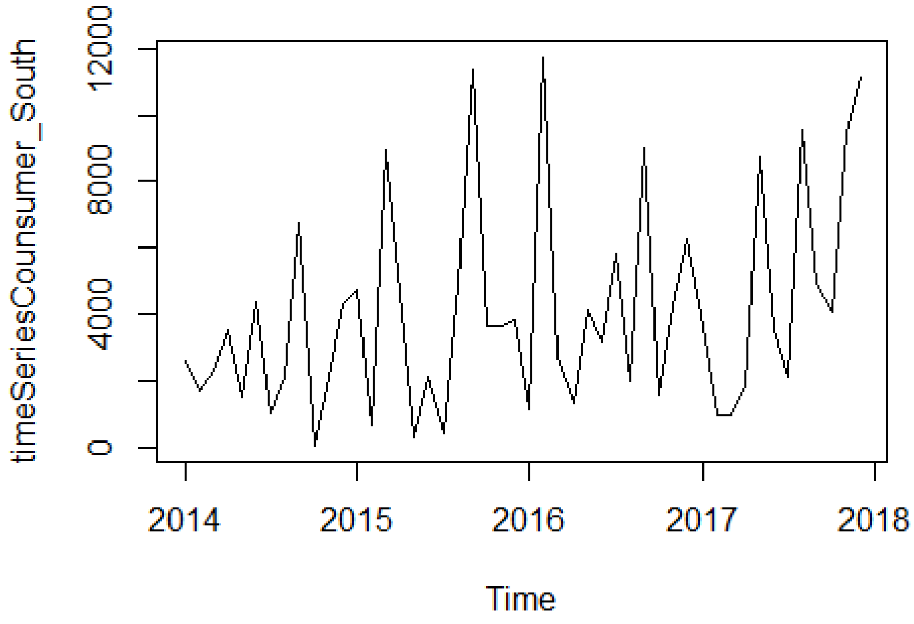
```
##     Order.Date  Sales year month
## 229 2014-01-13 545.94 2014    01
## 145 2014-01-15 149.95 2014    01
## 86  2014-01-20 699.93 2014    01
## 87  2014-01-20  22.96 2014    01
## 88  2014-01-20  38.60 2014    01
## 89  2014-01-20   6.63 2014    01
```

```
library(sqldf)
Consumer_South<-sqldf("select year, month, sum(Sales) as Month_Sales from
Consumer_South group by year, month order by year, month")
tail(Consumer_South)
```

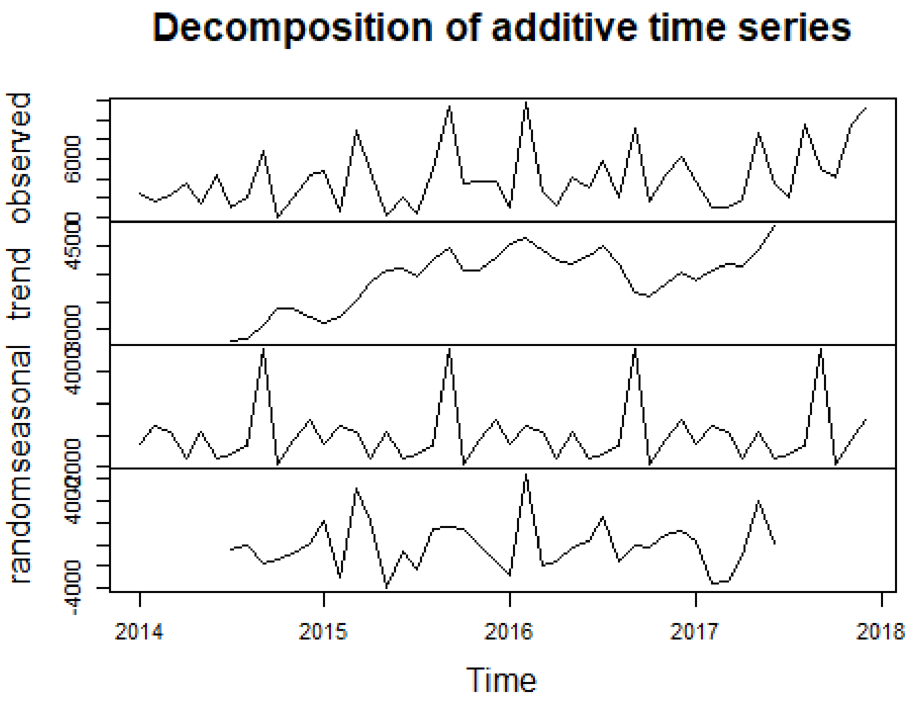
```
##    year month Month_Sales
## 43 2017    07    2143.372
## 44 2017    08    9568.338
## 45 2017    09    4910.376
## 46 2017    10    4075.180
## 47 2017    11    9475.910
## 48 2017    12   11159.322
```

Sum of Total Sales for Counsumer Segment in South in December 2017 is $11159.3222

```
timeSeriesCounsumer_South<-ts(Consumer_South$Month_Sales,
frequency=12,start=c(2014,1),end=c(2017,12))
plot(timeSeriesCounsumer_South)
```



```
decompose_consumer_south<-decompose(timeSeriesCounsumer_South)
plot(decompose_consumer_south)
```



TIME SERIES FORCASTING

Simple Smoothing

For next 2 years:

```
library(forecast)
simple_smoothing1_south <- forecast(timeSeriesCounsumer_South)
summary(simple_smoothing1_south)
```

```
##
## Forecast method: ETS(M,A,N)
##
## Model Information:
## ETS(M,A,N)
##
## Call:
##  ets(y = object, lambda = lambda, allow.multiplicative.trend =
allow.multiplicative.trend)
##
##   Smoothing parameters:
##     alpha = 0.0127
##     beta  = 1e-04
##
##   Initial states:
##     l = 2330.6528
##     b = 69.477
##
##   sigma:  0.7269
##
##      AIC     AICc      BIC
## 959.4379 960.8665 968.7939
##
## Error measures:
##                    ME     RMSE      MAE       MPE     MAPE     MASE
## Training set 38.47471 2954.568 2338.515 -214.7548 244.0591 0.692449
##                    ACF1
## Training set -0.1299707
##
## Forecasts:
##          Point Forecast    Lo 80    Hi 80     Lo 95    Hi 95
## Jan 2018       5759.238 394.3470 11124.13 -2445.654 13964.13
## Feb 2018       5828.899 398.4500 11259.35 -2476.256 14134.05
## Mar 2018       5898.561 402.5423 11394.58 -2506.874 14304.00
## Apr 2018       5968.223 406.6240 11529.82 -2537.508 14473.95
## May 2018       6037.884 410.6948 11665.07 -2568.159 14643.93
## Jun 2018       6107.546 414.7545 11800.34 -2598.827 14813.92
## Jul 2018       6177.208 418.8032 11935.61 -2629.512 14983.93
## Aug 2018       6246.869 422.8405 12070.90 -2660.214 15153.95
## Sep 2018       6316.531 426.8664 12206.20 -2690.934 15324.00
## Oct 2018       6386.193 430.8807 12341.50 -2721.671 15494.06
## Nov 2018       6455.854 434.8834 12476.83 -2752.426 15664.14
## Dec 2018       6525.516 438.8742 12612.16 -2783.199 15834.23
## Jan 2019       6595.178 442.8530 12747.50 -2813.991 16004.35
## Feb 2019       6664.839 446.8197 12882.86 -2844.801 16174.48
## Mar 2019       6734.501 450.7742 13018.23 -2875.630 16344.63
## Apr 2019       6804.163 454.7164 13153.61 -2906.478 16514.80
## May 2019       6873.825 458.6460 13289.00 -2937.344 16684.99
## Jun 2019       6943.486 462.5630 13424.41 -2968.230 16855.20
## Jul 2019       7013.148 466.4673 13559.83 -2999.136 17025.43
## Aug 2019       7082.810 470.3587 13695.26 -3030.061 17195.68
## Sep 2019       7152.471 474.2371 13830.71 -3061.006 17365.95
## Oct 2019       7222.133 478.1024 13966.16 -3091.972 17536.24
## Nov 2019       7291.795 481.9544 14101.63 -3122.957 17706.55
## Dec 2019       7361.456 485.7930 14237.12 -3153.963 17876.88
```

After 1 year forecast of total sales would be $6525.516

2. For Central

```
library(dplyr)
Consumer_Central<-dataNW %>%
  select(Order.Date, Sales) %>%
  filter(dataNW$Segment == "Consumer",dataNW$Region=="Central")

tail(Consumer_Central)

##      Order.Date   Sales
## 1207 2017-01-29  13.480
## 1208 2017-01-29  14.910
## 1209 2017-05-06 191.968
## 1210 2016-09-11  99.568
## 1211 2016-09-22  35.560
## 1212 2016-09-22  97.980

Consumer_Central<-Consumer_Central[order(as.Date(Consumer_Central$Order.Date, 
"%m/%d/%Y"), decreasing = FALSE),]
tail(Consumer_Central)

##     Order.Date   Sales
## 941 2017-12-25 31.7440
## 942 2017-12-25 40.9800
## 943 2017-12-25  3.1680
## 124 2017-12-26 44.7500
## 121 2017-12-28 27.1680
## 122 2017-12-28 78.8528

Consumer_Central$year <- format(as.Date(Consumer_Central$Order.Date), "%Y")
Consumer_Central$month <- format(as.Date(Consumer_Central$Order.Date), "%m")

head(Consumer_Central)
```

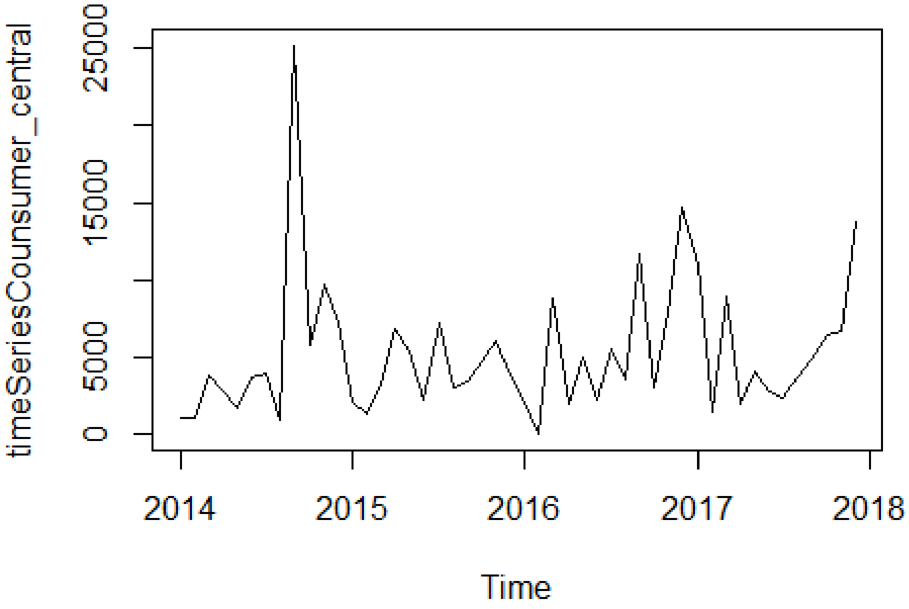
```
##     Order.Date  Sales year month
## 948 2014-01-03 16.448 2014    01
## 913 2014-01-07 76.728 2014    01
## 914 2014-01-07 10.430 2014    01
## 80  2014-01-09  9.344 2014    01
## 81  2014-01-09 31.200 2014    01
## 762 2014-01-20 13.980 2014    01
```

```
library(sqldf)
Consumer_Central<-sqldf("select year, month, sum(Sales) as Month_Sales from
Consumer_Central group by year, month order by year, month")
tail(Consumer_Central)
```

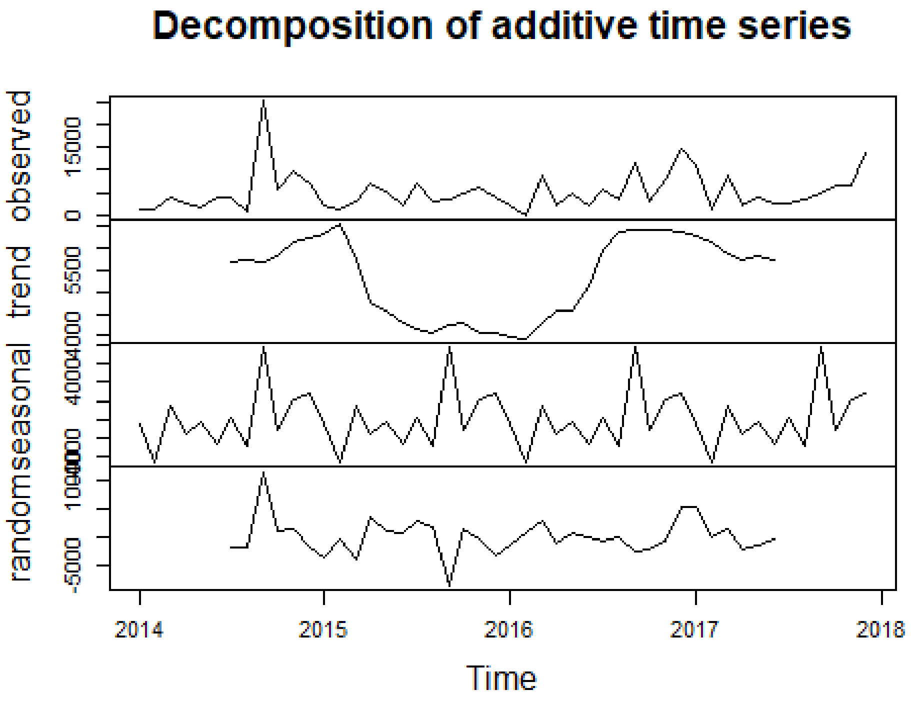
```
##    year month Month_Sales
## 43 2017    07    2444.300
## 44 2017    08    3685.180
## 45 2017    09    4918.147
## 46 2017    10    6455.635
## 47 2017    11    6666.942
## 48 2017    12   13793.727
```

Total Sum of Sales in December 2017 is $13793.727 for Counsmer Segment in Central region

```
timeSeriesCounsumer_central<-ts(Consumer_Central$Month_Sales,
frequency=12,start=c(2014,1),end=c(2017,12))
plot(timeSeriesCounsumer_central)
```



```
decompose_consumer_central<-decompose(timeSeriesCounsumer_central)
plot(decompose_consumer_central)
```



There is no trend but there is sesonality.

TIME SERIES FORCASTING

Simple Smoothing

For next 2 years:

```
library(forecast)
simple_smoothing1_central <- forecast(timeSeriesCounsumer_central)
summary(simple_smoothing1_central)

##
## Forecast method: ETS(M,N,M)
##
## Model Information:
## ETS(M,N,M)
##
## Call:
##  ets(y = object, lambda = lambda, allow.multiplicative.trend =
allow.multiplicative.trend)
##
##   Smoothing parameters:
##     alpha = 1e-04
##     gamma = 1e-04
```

```
##
##   Initial states:
##     l = 5336.0278
##     s=1.7751 1.249 0.8951 2.4845 0.5117 0.8457
##            0.4432 0.7023 0.6477 1.1667 0.1873 1.0917
##
##   sigma:  0.4815
##
##      AIC     AICc      BIC
## 951.3362 966.3362 979.4042
##
## Error measures:
##                      ME     RMSE      MAE      MPE     MAPE      MASE
## Training set -84.98732 3422.136 2348.882 -58.4567 86.25298 0.7611872
##                    ACF1
## Training set 0.02228257
##
## Forecasts:
##          Point Forecast     Lo 80     Hi 80     Lo 95     Hi 95
## Jan 2018      5824.9671 2230.2999  9419.634 327.39805 11322.536
## Feb 2018       999.4334  382.6693  1616.198  56.17414  1942.693
## Mar 2018      6225.8723 2383.8010 10067.944 349.93125 12101.813
## Apr 2018      3456.4686 1323.4343  5589.503 194.27418  6718.663
## May 2018      3747.9618 1435.0431  6060.881 210.65782  7285.266
## Jun 2018      2365.1504  905.5836  3824.717 132.93555  4597.365
## Jul 2018      4513.2703 1728.0691  7298.472 253.67267  8772.868
## Aug 2018      2730.4477 1045.4509  4415.445 153.46741  5307.428
## Sep 2018     13257.3473 5076.0558 21438.639 745.14173 25769.553
## Oct 2018      4776.6902 1828.9289  7724.452 268.47836  9284.902
## Nov 2018      6665.4715 2552.1172 10778.826 374.63904 12956.304
## Dec 2018      9472.7526 3626.9865 15318.519 532.42484 18413.080
## Jan 2019      5824.9671 2230.2996  9419.635 327.39754 11322.537
## Feb 2019       999.4334  382.6693  1616.198  56.17405  1942.693
## Mar 2019      6225.8723 2383.8006 10067.944 349.93070 12101.814
## Apr 2019      3456.4686 1323.4341  5589.503 194.27388  6718.663
## May 2019      3747.9618 1435.0428  6060.881 210.65749  7285.266
## Jun 2019      2365.1504  905.5834  3824.717 132.93534  4597.365
## Jul 2019      4513.2703 1728.0689  7298.472 253.67228  8772.868
## Aug 2019      2730.4477 1045.4507  4415.445 153.46717  5307.428
## Sep 2019     13257.3474 5076.0551 21438.640 745.14055 25769.554
## Oct 2019      4776.6902 1828.9287  7724.452 268.47793  9284.903
## Nov 2019      6665.4715 2552.1169 10778.826 374.63845 12956.305
## Dec 2019      9472.7526 3626.9860 15318.519 532.42400 18413.081
```

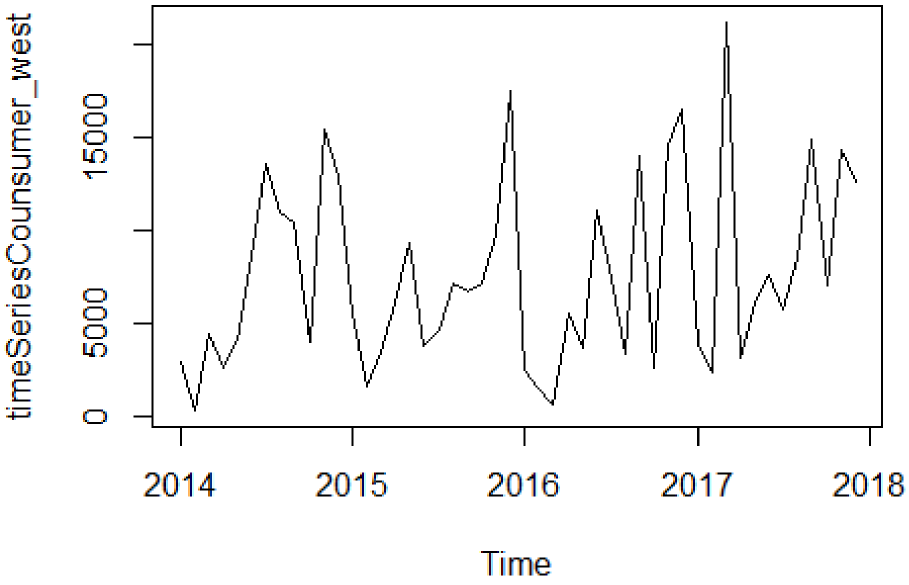
Forecast of Total sales after 1 year would be $9472.7526

3. FOR WEST

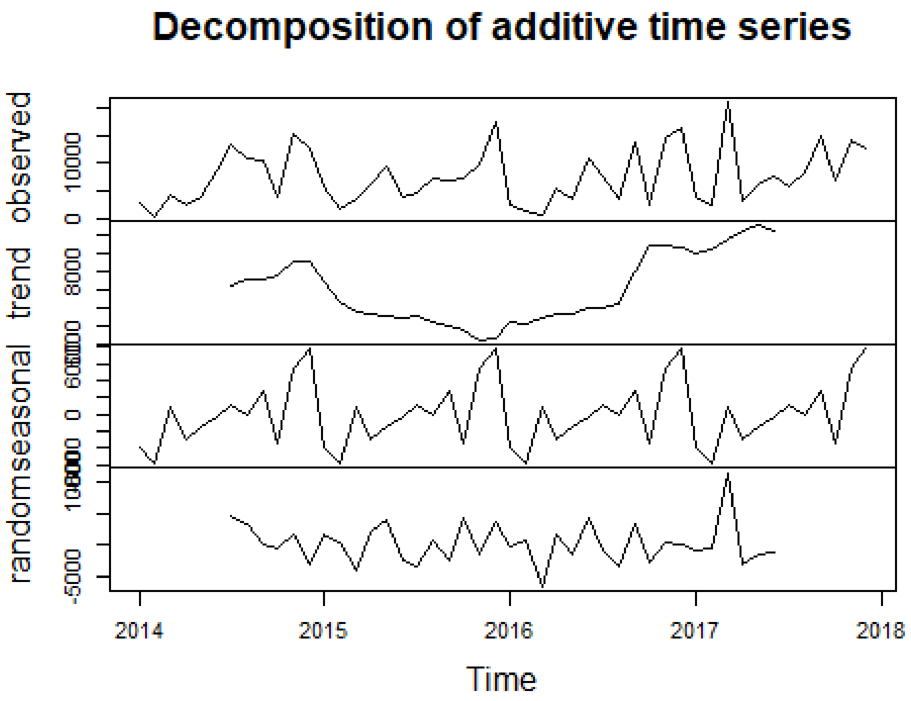
```
library(dplyr)
Consumer_West<-dataNW %>%
  select(Order.Date, Sales) %>%
  filter(dataNW$Segment == "Consumer",dataNW$Region=="West")

tail(Consumer_West)

##      Order.Date   Sales
## 1667 2017-10-06   9.344
## 1668 2016-09-29  36.240
## 1669 2017-02-26  91.960
## 1670 2017-02-26 258.576
## 1671 2017-02-26  29.600
## 1672 2017-05-04 243.160

Consumer_West<-Consumer_West[order(as.Date(Consumer_West$Order.Date,
"%m/%d/%Y"), decreasing = FALSE),]
tail(Consumer_West)

##      Order.Date   Sales
## 1400 2017-12-28   4.280
## 336  2017-12-29 393.568
## 337  2017-12-29 302.376
## 960  2017-12-29  19.600
## 961  2017-12-29  68.460
## 898  2017-12-30   3.024

Consumer_West$year <- format(as.Date(Consumer_West$Order.Date), "%Y")
Consumer_West$month <- format(as.Date(Consumer_West$Order.Date), "%m")

head(Consumer_West)

##     Order.Date    Sales year month
## 934 2014-01-06   19.440 2014    01
## 862 2014-01-13 1325.850 2014    01
## 863 2014-01-13  333.999 2014    01
## 864 2014-01-13   19.900 2014    01
## 965 2014-01-19   32.340 2014    01
## 966 2014-01-19   56.064 2014    01

library(sqldf)
Consumer_West<-sqldf("select year, month, sum(Sales) as Month_Sales from
Consumer_West group by year, month order by year, month")
tail(Consumer_West)

##    year month Month_Sales
## 43 2017    07    5730.812
## 44 2017    08    8444.222
## 45 2017    09   14898.420
## 46 2017    10    7013.451
## 47 2017    11   14295.271
## 48 2017    12   12589.989
```

Total Sales in December 2017 is $12589.989

```
timeSeriesCounsumer_west<-ts(Consumer_West$Month_Sales,
frequency=12,start=c(2014,1),end=c(2017,12))
plot(timeSeriesCounsumer_west)
```



```
decompose_consumer_west<-decompose(timeSeriesCounsumer_west)
plot(decompose_consumer_west)
```



TIME SERIES FORCASTING

Simple Smoothing

For next 2 years:

```
library(forecast)
simple_smoothing1_west <- forecast(timeSeriesCounsumer_west)
summary(simple_smoothing1_west)

##
## Forecast method: ETS(M,N,M)
##
## Model Information:
## ETS(M,N,M)
##
## Call:
##  ets(y = object, lambda = lambda, allow.multiplicative.trend =
allow.multiplicative.trend)
##
##   Smoothing parameters:
##     alpha = 5e-04
##     gamma = 1e-04
##
##   Initial states:
##     l = 7060.744
##     s=1.6411 1.7225 0.6817 1.36 1.0907 1.0313
##            0.983 0.7391 0.6657 1.3756 0.2322 0.477
## 
##   sigma:  0.4476
## 
##        AIC      AICc       BIC 
##  977.2084  992.2084 1005.2764 
## 
## Error measures:
##                     ME     RMSE      MAE       MPE    MAPE      MASE
## Training set 499.7869 3508.182 2613.277 -48.36256 75.6766 0.6726323
##                     ACF1
## Training set -0.02950972
## 
## Forecasts:
##          Point Forecast     Lo 80     Hi 80     Lo 95     Hi 95
## Jan 2018       3372.676 1438.0003  5307.352  413.8448  6331.508
## Feb 2018       1641.411  699.8447  2582.978  201.4094  3081.413
## Mar 2018       9724.167 4146.0694 15302.266 1193.2027 18255.132
## Apr 2018       4706.303 2006.6144  7405.992  577.4856  8835.120
## May 2018       5225.699 2228.0673  8223.330  641.2171  9810.181
## Jun 2018       6949.518 2963.0468 10935.989  852.7365 13046.299
## Jul 2018       7290.967 3108.6289 11473.306  894.6327 13687.302
## Aug 2018       7710.770 3287.6183 12133.922  946.1432 14475.397
## Sep 2018       9615.613 4099.7796 15131.447 1179.8738 18051.353
## Oct 2018       4819.576 2054.9075  7584.245  591.3803  9047.772
## Nov 2018      12177.834 5192.2235 19163.444 1494.2647 22861.403
## Dec 2018      11603.561 4947.3715 18259.751 1423.7976 21783.325
## Jan 2019       3372.676 1437.9962  5307.356  413.8385  6331.514
## Feb 2019       1641.411  699.8426  2582.980  201.4063  3081.416
## Mar 2019       9724.168 4146.0575 15302.278 1193.1844 18255.151
## Apr 2019       4706.303 2006.6086  7405.997  577.4767  8835.129
## May 2019       5225.699 2228.0609  8223.337  641.2072  9810.190
## Jun 2019       6949.518 2963.0382 10935.997  852.7234 13046.312
## Jul 2019       7290.967 3108.6200 11473.315  894.6190 13687.316
## Aug 2019       7710.770 3287.6088 12133.931  946.1286 14475.412
## Sep 2019       9615.613 4099.7679 15131.459 1179.8557 18051.371
## Oct 2019       4819.576 2054.9016  7584.251  591.3713  9047.781
## Nov 2019      12177.834 5192.2086 19163.460 1494.2418 22861.426
## Dec 2019      11603.561 4947.3573 18259.765 1423.7757 21783.347
```

Forecast of Total Sales after 1 year would be $11603.561

4. FOR EAST

```r
library(dplyr)
Consumer_East<-dataNW %>%
  select(Order.Date, Sales) %>%
  filter(dataNW$Segment == "Consumer",dataNW$Region=="East")

tail(Consumer_East)
```

```
##      Order.Date  Sales
## 1464 2017-12-11  40.20
## 1465 2017-12-11 735.98
## 1466 2017-12-11  22.75
## 1467 2017-08-03  16.52
## 1468 2015-05-17  31.50
## 1469 2015-05-17  55.60
```

```
Consumer_East<-Consumer_East[order(as.Date(Consumer_East$Order.Date,
"%m/%d/%Y"), decreasing = FALSE),]
tail(Consumer_East)
```

```
##     Order.Date   Sales
## 90  2017-12-28  33.264
## 91  2017-12-28  14.850
## 469 2017-12-29   6.030
## 145 2017-12-30 323.136
## 146 2017-12-30  90.930
## 147 2017-12-30  52.776
```

```
Consumer_East$year <- format(as.Date(Consumer_East$Order.Date), "%Y")
Consumer_East$month <- format(as.Date(Consumer_East$Order.Date), "%m")

head(Consumer_East)
```

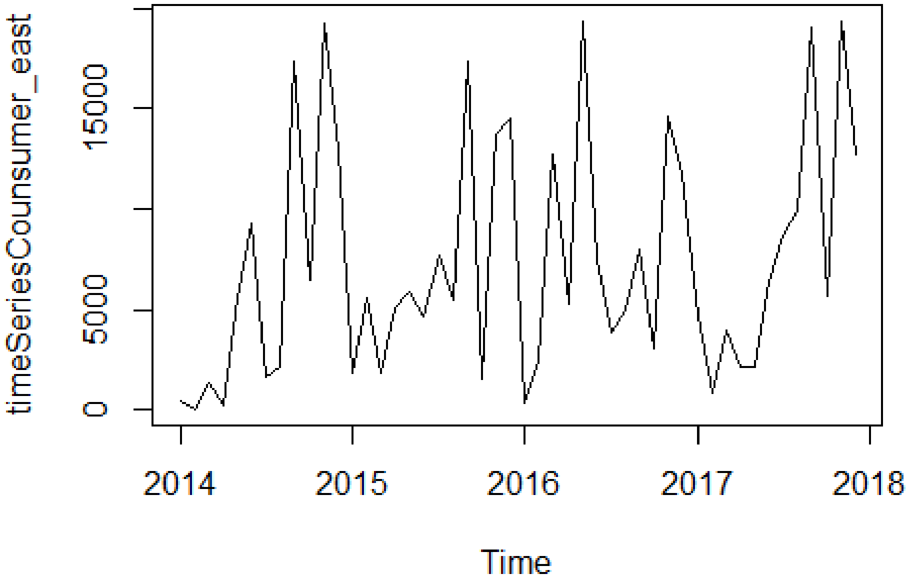
```
##      Order.Date   Sales year month
## 298  2014-01-05  19.536 2014    01
## 112  2014-01-11   9.940 2014    01
## 1418 2014-01-13  37.408 2014    01
## 1419 2014-01-13   3.438 2014    01
## 158  2014-01-16 127.104 2014    01
## 159  2014-01-16 124.200 2014    01
```

```
library(sqldf)
Consumer_East<-sqldf("select year, month, sum(Sales) as Month_Sales from
Consumer_East group by year, month order by year, month")
tail(Consumer_East)
```

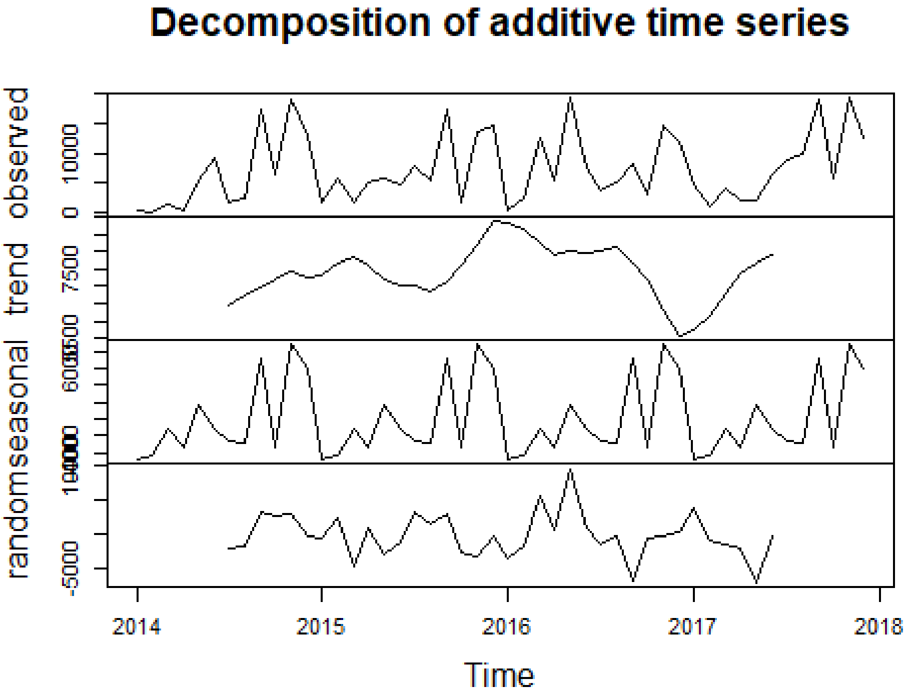
```
##    year month Month_Sales
## 43 2017    07    8672.889
## 44 2017    08    9931.404
## 45 2017    09   19130.260
## 46 2017    10    5649.974
## 47 2017    11   19351.942
## 48 2017    12   12689.418
```

Total Sales in Decemeber 2017 is $12689.418

```
timeSeriesCounsumer_east<-ts(Consumer_East$Month_Sales,
frequency=12,start=c(2014,1),end=c(2017,12))
plot(timeSeriesCounsumer_east)
```



```
decompose_consumer_east<-decompose(timeSeriesCounsumer_east)
plot(decompose_consumer_east)
```



TIME SERIES FORCASTING

Simple Smoothing

For next 2 years:

```
library(forecast)
simple_smoothing1_central <- forecast(timeSeriesCounsumer_east)
summary(simple_smoothing1_central)

##
## Forecast method: ETS(A,N,A)
##
## Model Information:
## ETS(A,N,A)
##
## Call:
##  ets(y = object, lambda = lambda, allow.multiplicative.trend =
allow.multiplicative.trend)
##
##   Smoothing parameters:
##     alpha = 0.0214
##     gamma = 1e-04
##
##   Initial states:
##     l = 6754.4662
##     s=6089.289 8780.633 -3445.884 7380.854 -2388.12 -2557.653
##            -1064.439 1725.577 -3288.694 -1254.146 -4308.31 -5669.105
##
##   sigma:  3359.383
##
##       AIC      AICc       BIC
##  995.2909 1010.2909 1023.3589
##
## Error measures:
##                    ME      RMSE       MAE        MPE     MAPE       MASE
## Training set 469.7282 3359.383 2642.819 -319.2025 352.151 0.6147263
##                   ACF1
## Training set 0.2265991
##
## Forecasts:
##          Point Forecast       Lo 80      Hi 80       Lo 95     Hi 95
## Jan 2018       1568.976 -2736.2464  5874.199 -5015.2936  8153.246
## Feb 2018       2929.399 -1376.8125  7235.611 -3656.3834  9515.182
## Mar 2018       5983.428  1676.2271 10290.629  -603.8673 12570.723
## Apr 2018       3948.984  -359.2053  8257.174 -2639.8231 10537.792
## May 2018       8963.285  4654.1070 13272.463  2372.9660 15553.604
## Jun 2018       6173.864  1863.6981 10484.031  -417.9661 12765.695
## Jul 2018       4680.648   369.4938  8991.803 -1912.6935 11273.990
## Aug 2018       4850.170   538.0277  9162.312 -1744.6825 11445.022
## Sep 2018      14619.156 10306.0257 18932.285  8022.7927 21215.518
## Oct 2018       3792.229  -521.8885  8106.346 -2805.6442 10390.101
## Nov 2018      16018.881 11703.7764 20333.985  9419.4982 22618.263
## Dec 2018      13327.104  9011.0035 17643.204  6726.1979 19928.010
## Jan 2019       1568.976 -2748.1108  5886.063 -5033.4387  8171.391
## Feb 2019       2929.399 -1388.6742  7247.473 -3674.5242  9533.323
## Mar 2019       5983.428  1664.3681 10302.488  -622.0040 12588.860
## Apr 2019       3948.984  -371.0615  8269.030 -2657.9557 10555.924
## May 2019       8963.285  4642.2535 13284.317  2354.8375 15571.733
## Jun 2019       6173.864  1851.8473 10495.882  -436.0904 12783.819
## Jul 2019       4680.648   357.6456  9003.651 -1930.8137 11292.110
## Aug 2019       4850.170   526.1823  9174.158 -1762.7985 11463.138
## Sep 2019      14619.156 10294.1830 18944.128  8004.6809 21233.630
## Oct 2019       3792.229  -533.7286  8118.186 -2823.7520 10408.209
## Nov 2019      16018.881 11691.9391 20345.822  9401.3945 22636.367
## Dec 2019      13327.104  8999.1689 17655.039  6708.0985 19946.109
```

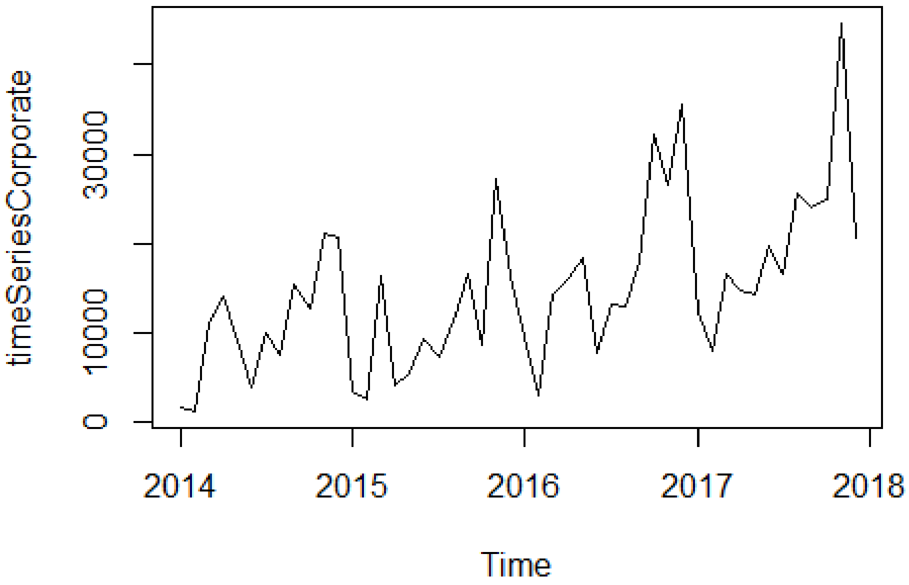
Forecast of Total Sales after 1 year would be $13327.104

Sale in the Central Region is going high.

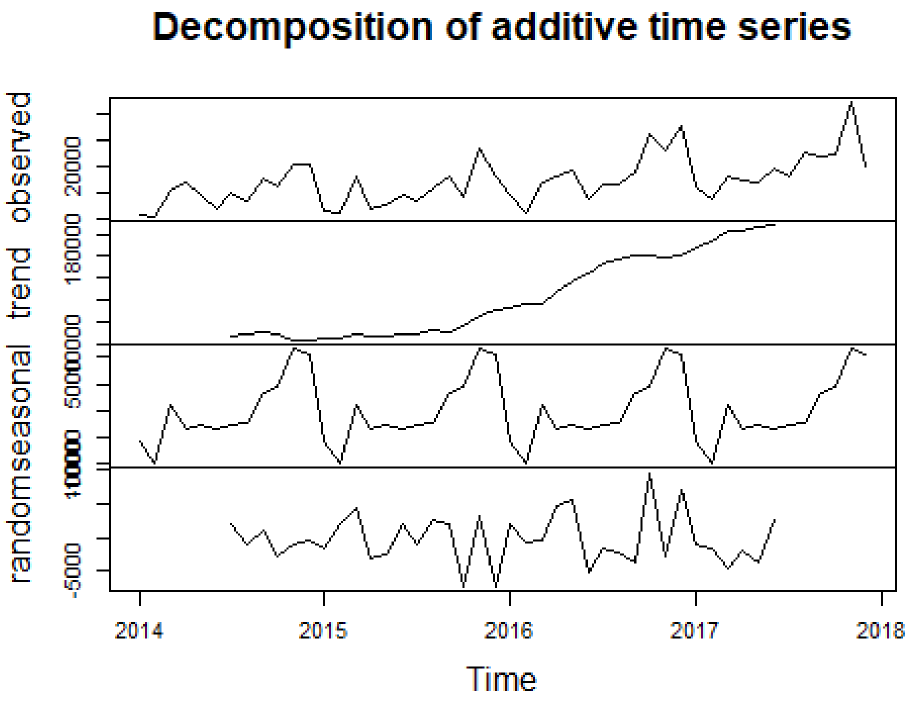
CORPORATE

```
library(dplyr)
Corporate<-dataNW %>%
  select(Order.Date, Sales) %>%
  filter(dataNW$Segment == "Corporate")

Corporate<-Corporate[order(as.Date(Corporate$Order.Date, "%m/%d/%Y"),
decreasing = FALSE),]
tail(Corporate)

##      Order.Date   Sales
## 1984 2017-12-28 113.372
## 1985 2017-12-28   2.960
## 1241 2017-12-29 209.700
## 1452 2017-12-29 101.120
## 365  2017-12-30  13.904
## 366  2017-12-30  20.720

Corporate$year <- format(as.Date(Corporate$Order.Date), "%Y")
Corporate$month <- format(as.Date(Corporate$Order.Date), "%m")

head(Corporate)

##      Order.Date  Sales year month
## 2115 2014-01-06  12.78 2014    01
## 252  2014-01-10   2.89 2014    01
## 253  2014-01-10  51.94 2014    01
## 214  2014-01-13  11.36 2014    01
## 215  2014-01-13  50.94 2014    01
## 216  2014-01-13 646.74 2014    01
```

```
library(sqldf)
Corporate<-sqldf("select year, month, sum(Sales) as Month_Sales from
Corporate group by year, month order by year, month")
tail(Corporate)

##    year month Month_Sales
## 43 2017    07    16525.60
## 44 2017    08    25678.44
## 45 2017    09    24105.10
## 46 2017    10    24877.17
## 47 2017    11    44644.08
## 48 2017    12    20524.43

timeSeriesCorporate<-ts(Corporate$Month_Sales,
frequency=12,start=c(2014,1),end=c(2017,12))
plot(timeSeriesCorporate)
```



```
head(timeSeriesCorporate)

##           Jan       Feb       Mar       Apr       May       Jun
## 2014  1701.528  1183.668 11106.799 14131.729  9142.000  3970.914

timeSeriesCorporate_decompose<-decompose(timeSeriesCorporate)
plot(timeSeriesCorporate_decompose)
```



```
simple_smoothing2 <- forecast(timeSeriesCorporate)
summary(simple_smoothing2)

##
## Forecast method: ETS(M,N,M)
##
## Model Information:
## ETS(M,N,M)
##
## Call:
##  ets(y = object, lambda = lambda, allow.multiplicative.trend =
allow.multiplicative.trend)
##
##   Smoothing parameters:
##     alpha = 0.1973
##     gamma = 3e-04
##
##   Initial states:
##     l = 11069.1814
##     s=1.5103 1.8084 1.2989 1.1727 0.8978 0.7807
##            0.7659 0.8646 0.9993 1.1011 0.2643 0.536
##
##   sigma:  0.3688
##
##      AIC     AICc      BIC
## 1018.479 1033.479 1046.547
##
```

```
## Error measures:
##                    ME     RMSE      MAE      MPE     MAPE     MASE
## Training set 1030.707 4919.005 3845.256 -9.17516 35.99377 0.607545
##                    ACF1
## Training set -0.2638244
##
## Forecasts:
##          Point Forecast     Lo 80     Hi 80      Lo 95     Hi 95
## Jan 2018      10798.512  5694.347 15902.676  2992.3659 18604.658
## Feb 2018       5324.184  2752.537  7895.831  1391.1886  9257.179
## Mar 2018      22187.978 11245.125 33130.831  5452.3290 38923.628
## Apr 2018      20134.176 10002.442 30265.910  4639.0259 35629.326
## May 2018      17420.910  8482.434 26359.387  3750.6903 31091.131
## Jun 2018      15430.908  7363.146 23498.670  3092.3301 27769.486
## Jul 2018      15732.429  7355.765 24109.093  2921.4267 28543.431
## Aug 2018      18092.392  8287.418 27897.366  3096.9780 33087.806
## Sep 2018      23632.259 10603.370 36661.149  3706.2923 43558.226
## Oct 2018      26170.974 11499.829 40842.119  3733.3940 48608.554
## Nov 2018      36442.858 15679.323 57206.393  4687.7720 68197.945
## Dec 2018      30433.043 12817.617 48048.470  3492.5729 57373.514
## Jan 2019      10798.586  4450.902 17146.271  1090.6409 20506.532
## Feb 2019       5324.221  2147.187  8501.255   465.3665 10183.075
## Mar 2019      22188.131  8752.842 35623.421  1640.6289 42735.634
## Apr 2019      20134.315  7767.015 32501.615  1220.1613 39048.468
## May 2019      17421.031  6569.715 28272.347   825.3748 34016.686
## Jun 2019      15431.014  5686.969 25175.060   528.7824 30333.246
## Jul 2019      15732.537  5664.331 25800.744   334.5447 31130.530
## Aug 2019      18092.517  6361.408 29823.625   151.3347 36033.699
## Sep 2019      23632.422  8111.452 39153.393  -104.8538 47369.698
## Oct 2019      26171.155  8765.388 43576.921  -448.6687 52790.978
## Nov 2019      36443.110 11905.102 60981.117 -1084.5348 73970.754
## Dec 2019      30433.253  9692.508 51173.999 -1286.9798 62153.486
```

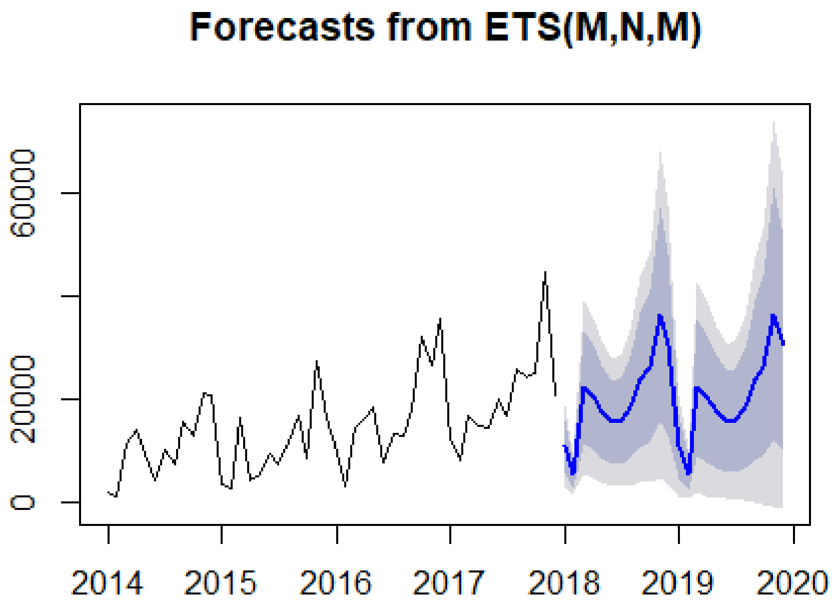
30433.043

20524.43

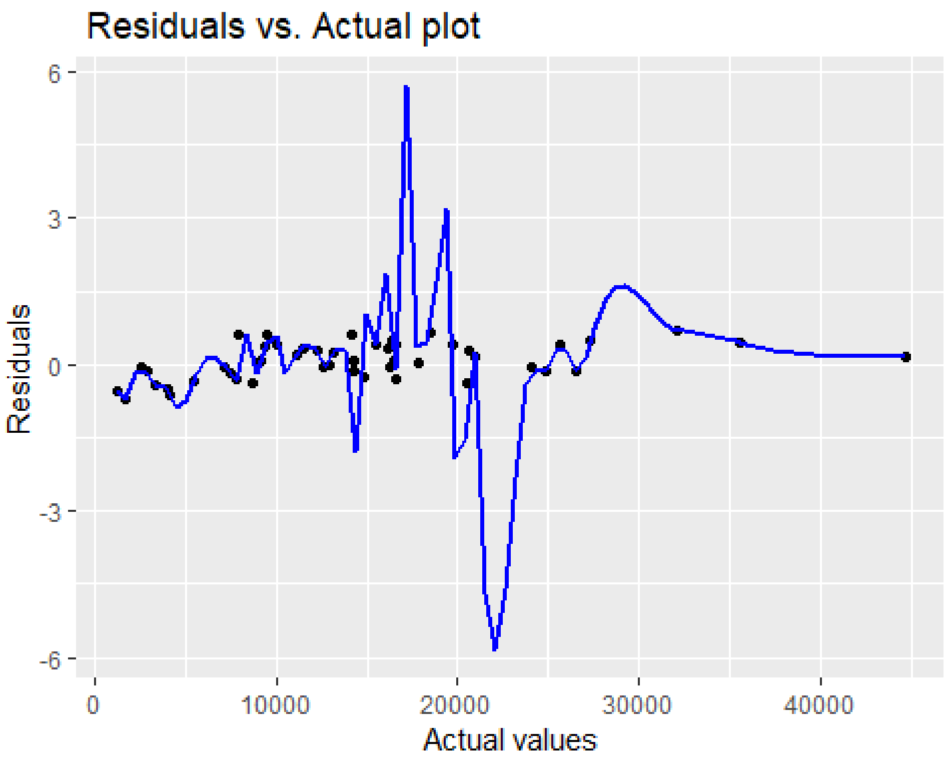
```
accuracy(simple_smoothing2)
```

```
##                    ME     RMSE      MAE      MPE     MAPE     MASE
## Training set 1030.707 4919.005 3845.256 -9.17516 35.99377 0.607545
##                    ACF1
## Training set -0.2638244
```

```
plot(simple_smoothing2)
```



```
library(ggplot2)
qplot(y = simple_smoothing2$residuals, x = timeSeriesCorporate,
      ylab = "Residuals", xlab = "Actual values",
      main = " Residuals vs. Actual plot") +
  stat_smooth(method = "loess", span = 0.1, colour = I("blue"), se = FALSE)

## Don't know how to automatically pick scale for object of type ts.
Defaulting to continuous.
## Don't know how to automatically pick scale for object of type ts.
Defaulting to continuous.
```



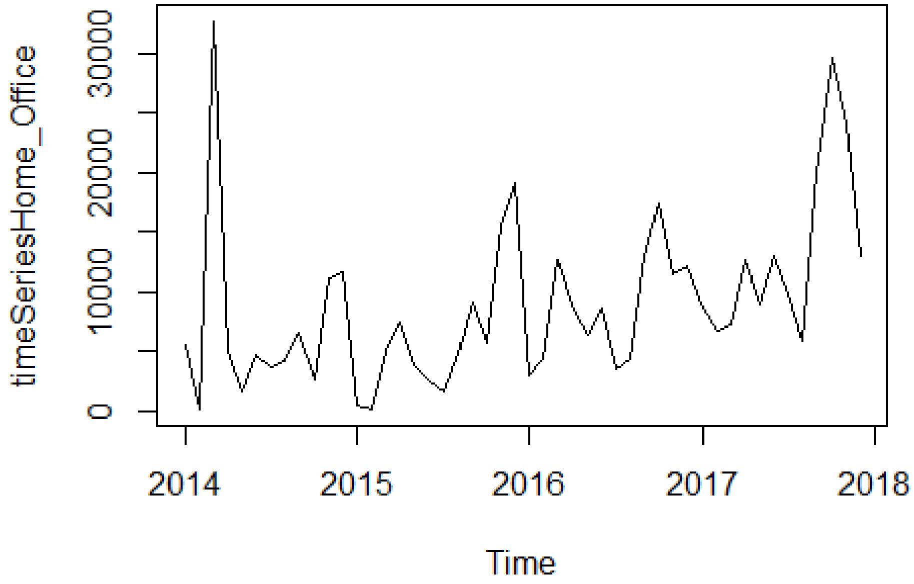
3. Home Office

```
library(dplyr)
Home_Office<-dataNW %>%
  select(Order.Date, Sales) %>%
  filter(dataNW$Segment == "Home Office")

Home_Office<-Home_Office[order(as.Date(Home_Office$Order.Date, "%m/%d/%Y"),
decreasing = FALSE),]
tail(Home_Office)

##      Order.Date   Sales
## 903  2017-12-25  90.480
## 704  2017-12-26   3.132
## 779  2017-12-28  64.784
## 1330 2017-12-28   2.480
## 1331 2017-12-28  25.900
## 95   2017-12-30 209.300

Home_Office$year <- format(as.Date(Home_Office$Order.Date), "%Y")
Home_Office$month <- format(as.Date(Home_Office$Order.Date), "%m")

head(Home_Office)

##      Order.Date    Sales year month
## 120  2014-01-04   11.784 2014    01
## 121  2014-01-04  272.736 2014    01
## 122  2014-01-04    3.540 2014    01
## 1321 2014-01-06 2573.820 2014    01
## 1322 2014-01-06  609.980 2014    01
## 1323 2014-01-06    5.480 2014    01
```

```
library(sqldf)
Home_Office<-sqldf("select year, month, sum(Sales) as Month_Sales from
Home_Office group by year, month order by year, month")
tail(Home_Office)
```

```
##    year month Month_Sales
## 43 2017    07    9747.444
## 44 2017    08    5813.300
## 45 2017    09   19904.346
## 46 2017    10   29705.515
## 47 2017    11   24013.684
## 48 2017    12   13072.431
```

```
timeSeriesHome_Office<-ts(Home_Office$Month_Sales,
frequency=12,start=c(2014,1),end=c(2017,12))
plot(timeSeriesHome_Office)
```

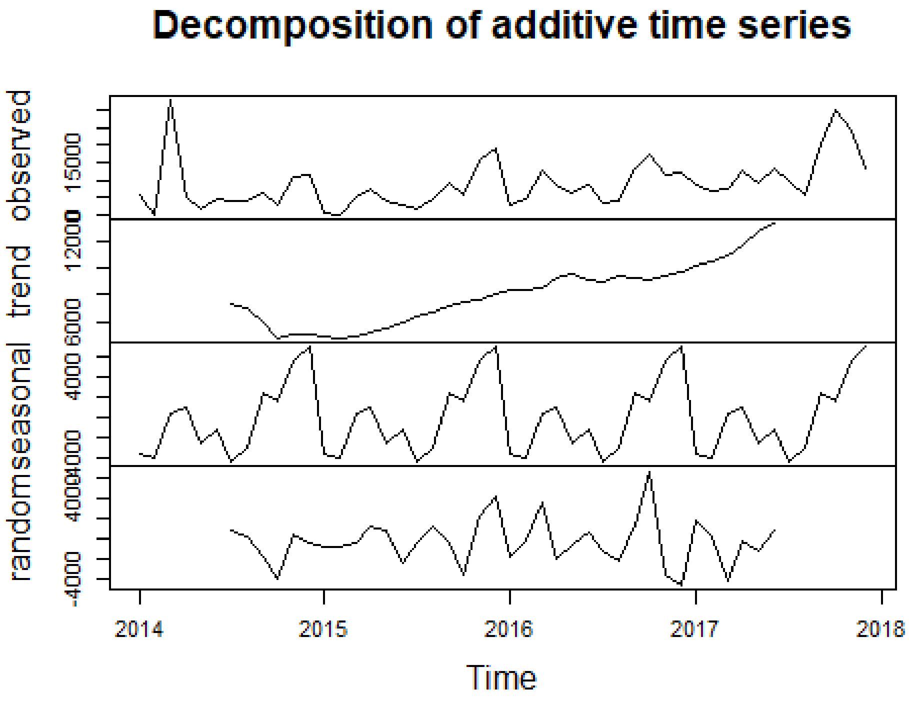


```
head(timeSeriesHome_Office)
```

```
##              Jan       Feb       Mar       Apr       May       Jun
## 2014  5607.550   168.370 32714.906  5055.004  1687.124  4700.466
```

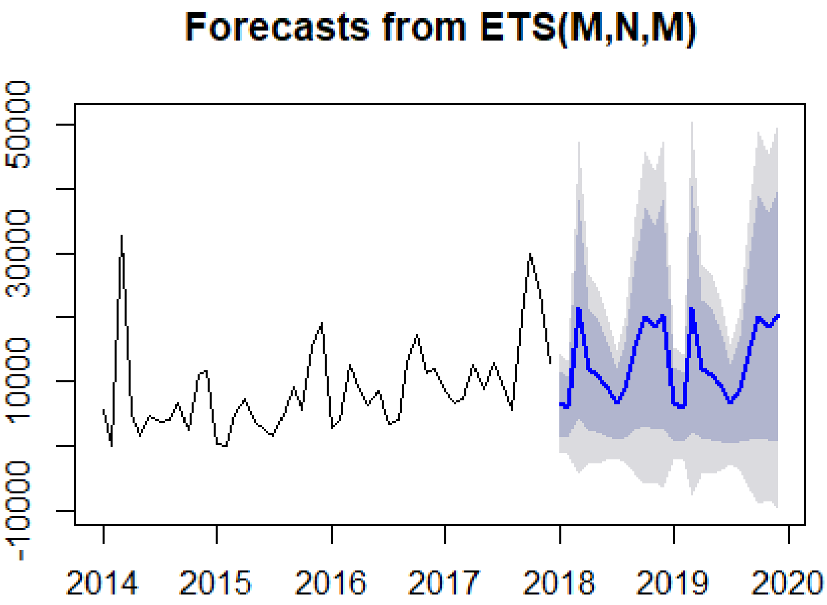
```
timeSeriesHome_Office_decompose<-decompose(timeSeriesHome_Office)
plot<-timeSeriesHome_Office_decompose
```

```r
plot(timeSeriesHome_Office_decompose)
```

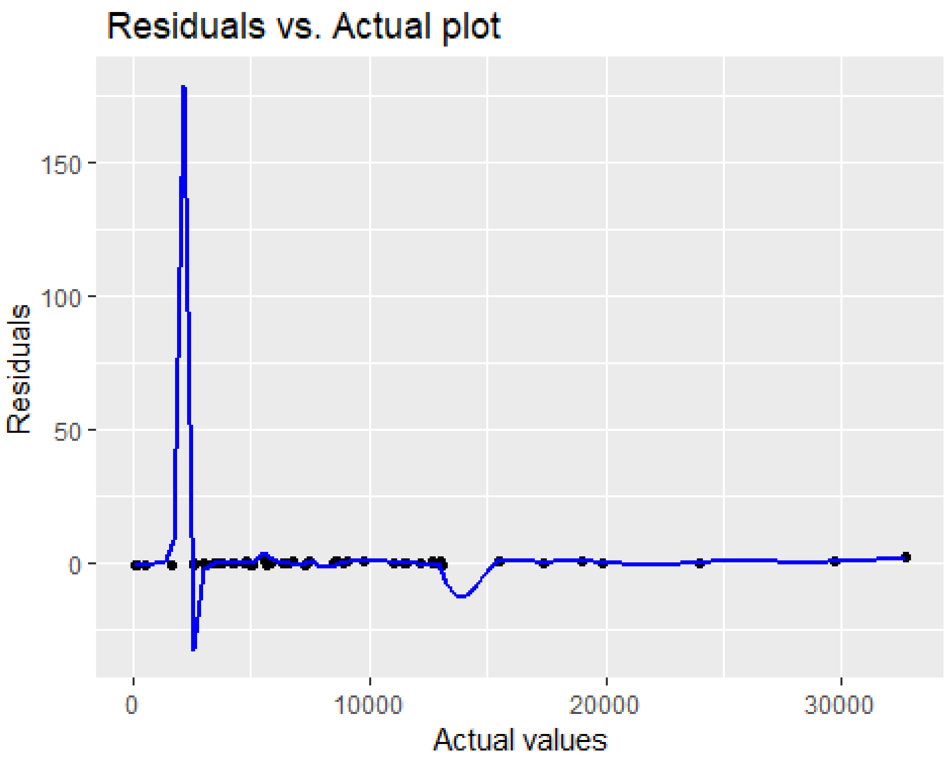


```
simple_smoothing3 <- forecast(timeSeriesHome_Office)
summary(simple_smoothing3)

##
## Forecast method: ETS(M,N,M)
##
## Model Information:
## ETS(M,N,M)
##
## Call:
##  ets(y = object, lambda = lambda, allow.multiplicative.trend =
allow.multiplicative.trend)
##
##   Smoothing parameters:
##     alpha = 0.1286
##     gamma = 1e-04
##
##   Initial states:
##     l = 7008.2519
##     s=1.5737 1.4167 1.5365 1.1958 0.674 0.5056
##            0.6981 0.8596 0.9213 1.6465 0.4608 0.5113
##
##   sigma:  0.6006
##
##      AIC     AICc      BIC
## 1012.907 1027.907 1040.975
##
## Error measures:
##                     ME     RMSE      MAE       MPE     MAPE      MASE
## Training set 1128.939 5245.177 3613.119 -106.5412 140.5028 0.7314989
##                     ACF1
## Training set -0.07726769
##
## Forecasts:
##          Point Forecast     Lo 80    Hi 80     Lo 95    Hi 95
## Jan 2018       6655.676 1532.8435 11778.51 -1179.020 14490.37
## Feb 2018       5998.195 1329.8021 10666.59 -1141.496 13137.89
## Mar 2018      21435.206 4568.6607 38301.75 -4359.949 47230.36
## Apr 2018      11993.674 2454.1208 21533.23 -2595.813 26583.16
## May 2018      11188.493 2194.4888 20182.50 -2566.649 24943.63
## Jun 2018       9088.627 1705.9068 16471.35 -2202.269 20379.52
## Jul 2018       6581.047 1179.9356 11982.16 -1679.240 14841.33
## Aug 2018       8773.585 1499.6247 16047.55 -2350.977 19898.15
## Sep 2018      15567.802 2531.1909 28604.41 -4369.974 35505.58
## Oct 2018      20000.426 3085.8865 36914.96 -5868.130 45868.98
## Nov 2018      18444.767 2693.3525 34196.18 -5644.942 42534.48
## Dec 2018      20486.942 2822.7900 38151.09 -6528.047 47501.93
## Jan 2019       6655.710  862.3327 12449.09 -2204.496 15515.92
## Feb 2019       5998.226  728.0834 11268.37 -2061.761 14058.21
## Mar 2019      21435.315 2427.1375 40443.49 -7635.184 50505.81
## Apr 2019      11993.734 1260.5987 22726.87 -4421.180 28408.65
## May 2019      11188.549 1085.3351 21291.76 -4262.983 26640.08
## Jun 2019       9088.673  808.2357 17369.11 -3575.163 21752.51
## Jul 2019       6581.080  532.2446 12629.92 -2669.816 15831.98
## Aug 2019       8773.629  639.1123 16908.15 -3667.041 21214.30
## Sep 2019      15567.880 1009.3608 30126.40 -6697.454 37833.21
## Oct 2019      20000.527 1136.9980 38864.06 -8848.751 48849.80
## Nov 2019      18444.860  901.5989 35988.12 -8385.243 45274.96
## Dec 2019      20487.046  838.5843 40135.51 -9562.683 50536.78
```

```
plot(simple_smoothing3)
```


Forecasts from ETS(M,N,M)

```
library(ggplot2)
qplot(y = simple_smoothing3$residuals, x = timeSeriesHome_Office,
      ylab = "Residuals", xlab = "Actual values",
      main = " Residuals vs. Actual plot") +
  stat_smooth(method = "loess", span = 0.1, colour = I("blue"), se = FALSE)

## Don't know how to automatically pick scale for object of type ts.
Defaulting to continuous.
## Don't know how to automatically pick scale for object of type ts.
Defaulting to continuous.

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : Chernobyl! trL>n 48


## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : Chernobyl! trL>n 48

## Warning in sqrt(sum.squares/one.delta): NaNs produced
```


Residuals vs. Actual plot
Residuals
150
100
50
0
0
10000
20000
30000
Actual values