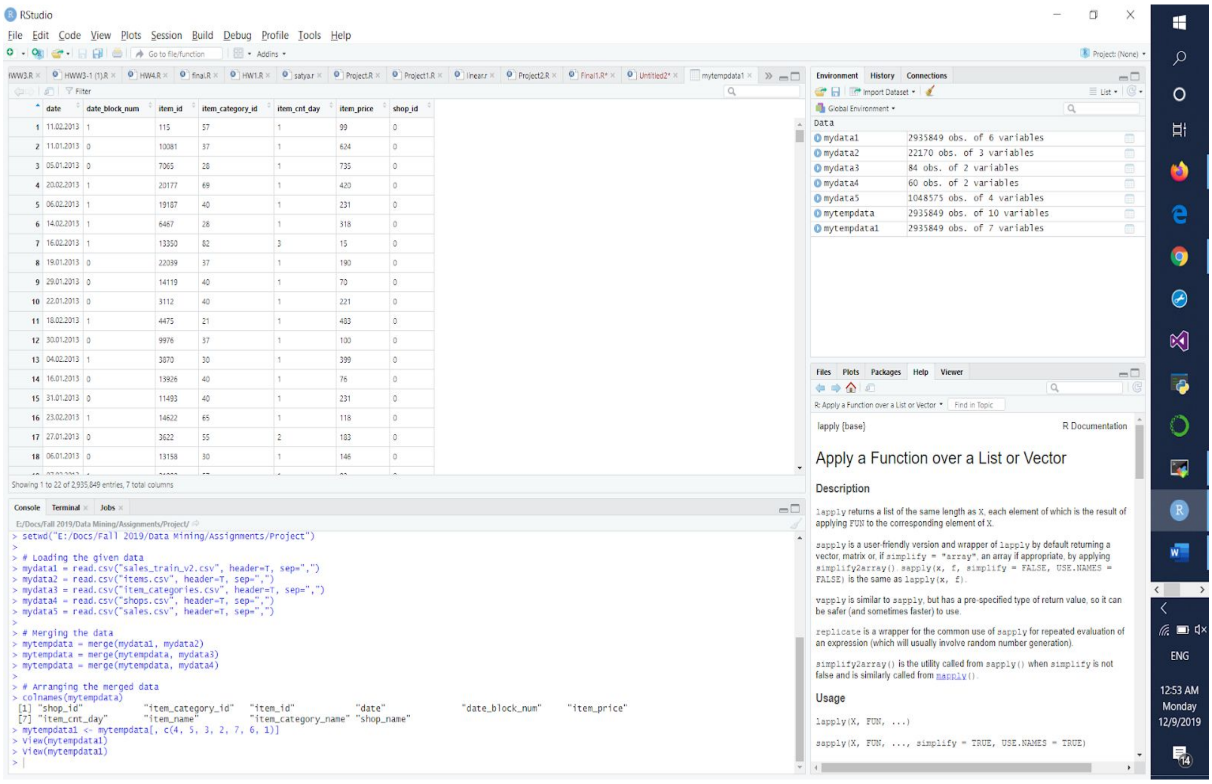
Data Mining Final Project (FALL 2019)

Tasks to be completed:

1. **Data Preparation and Exploratory analysis:**
    - **Merging Data:**
        - Merging all the data items:

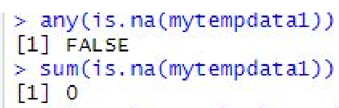
    Merging all the data frames and rearrangement of columns.



- **Cleaning Data:**
    - **Look for any missing data:**

    Identify observation on missing data:

    While checking for the missing values, we found out that there are no missing values in general like NULL or NA values. The output was FALSE.



- **Graph the representation of missing data:**

As we found that there is no missing data and occurrence of negative values like -1 is seen the dataset. So we considered negative values as missing values.

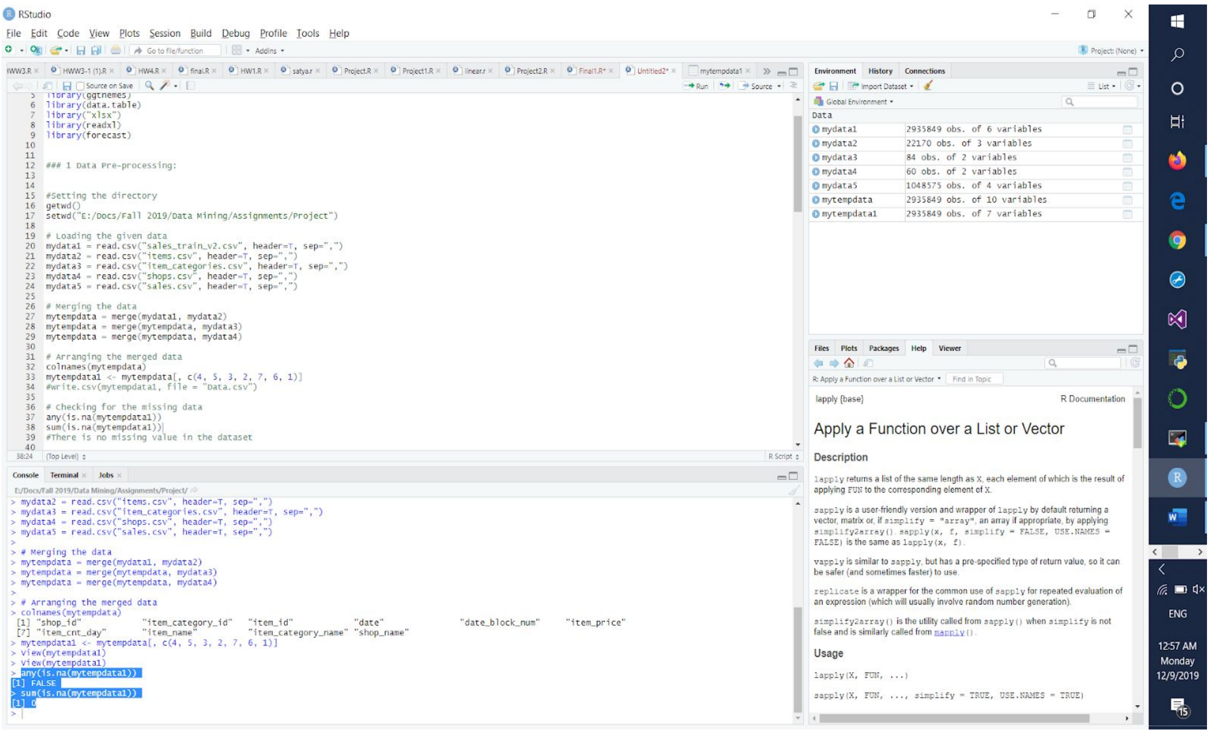
- **Decide whether to populate or remove missing data:**

Since the missing values as very small as compared to the entire dataset they don't affect the dataset as whole. So removal or population of the missing values won't make any difference.

- **Identify the possible of impact of it while modelling:**

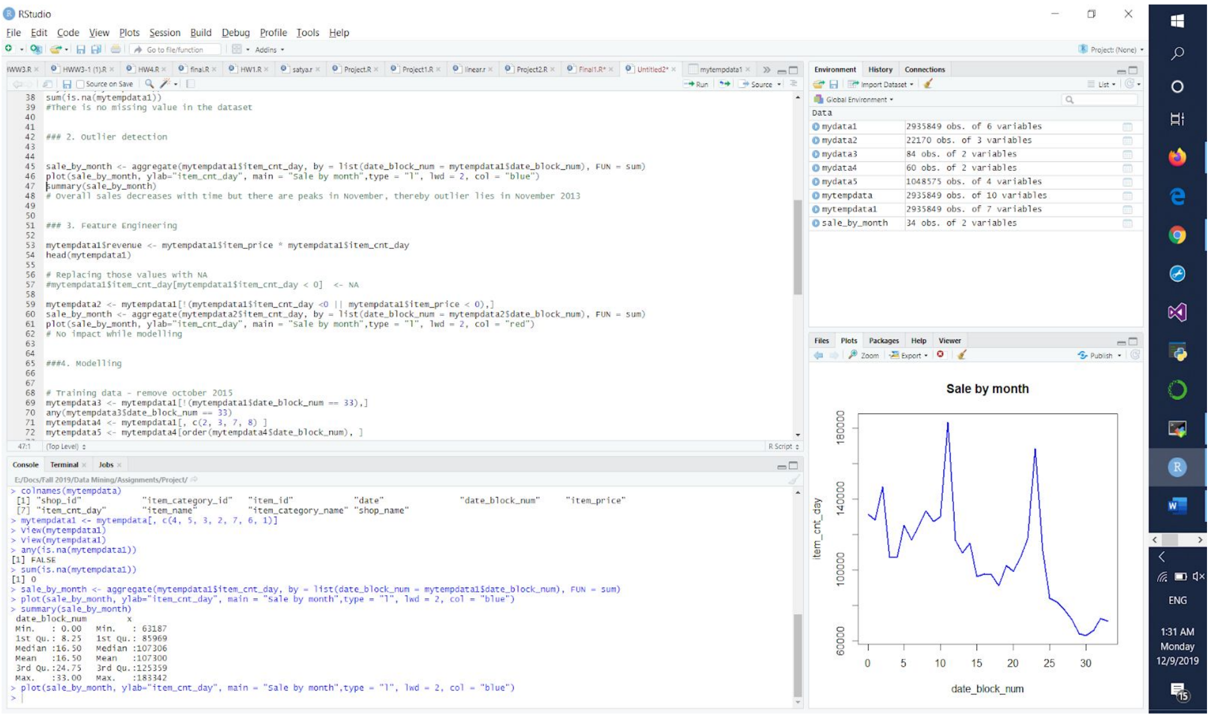
As mentioned above since the missing values as less as compared to the entire dataset it has negligible or no impact while modelling.

- Removed categorial attributes that were not required (item_name, item_category_name, shop_name).

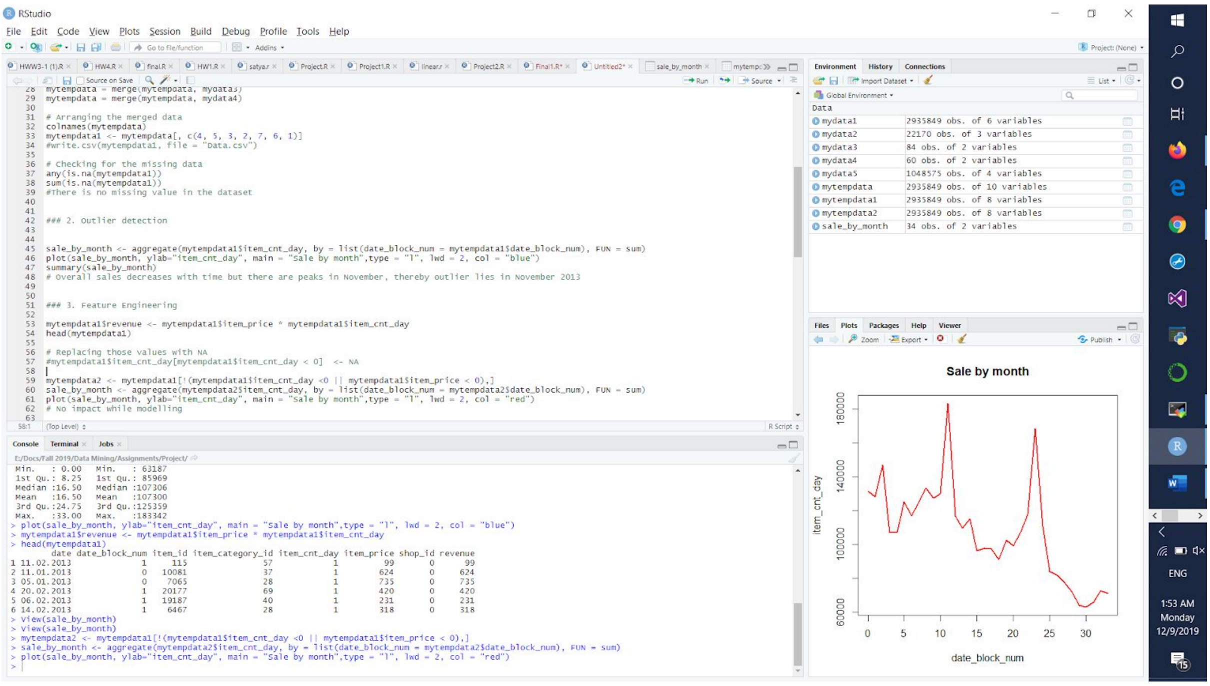


2. OUTLIER DETECTION:

- Graph represents the number of products sold for every consecutive month
- From the graph it is observed that the overall sales decrease with time but there are peaks in November months for consecutive years.
- It is found that there are two peaks in consecutive November 2013 and November 2014 which are considered as Outliers.

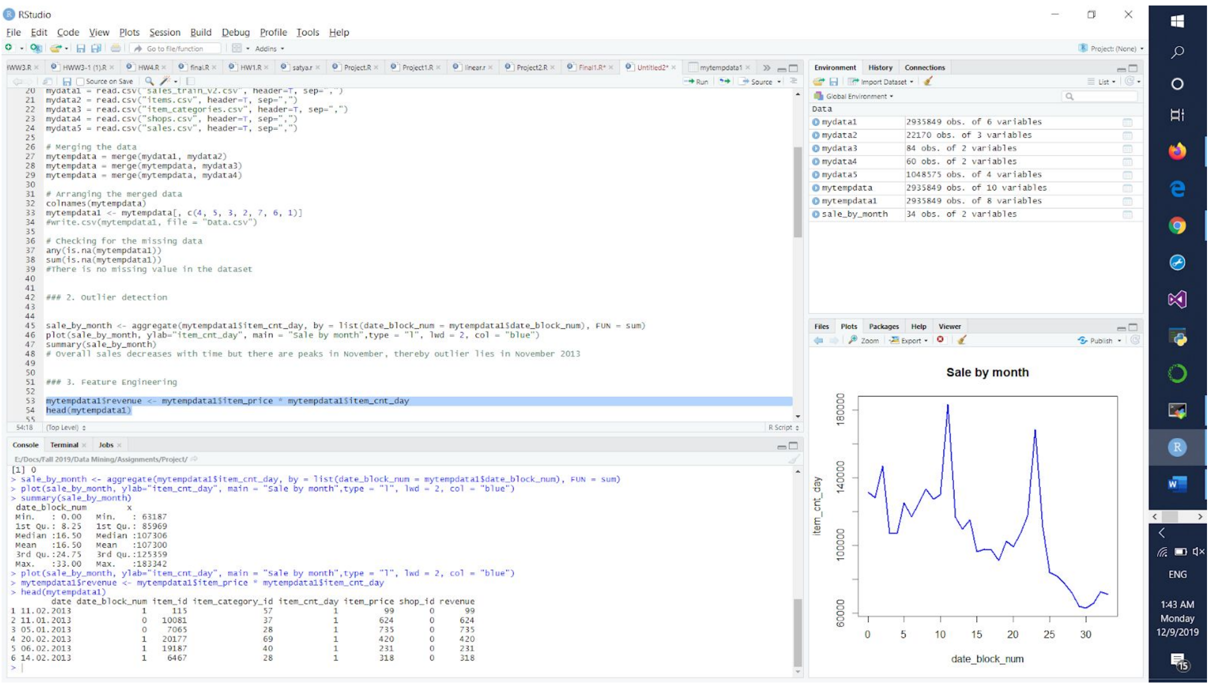


- Replacing the negative values with NA and then plotting the graph (represented by red line) and we found out that it remains the same.

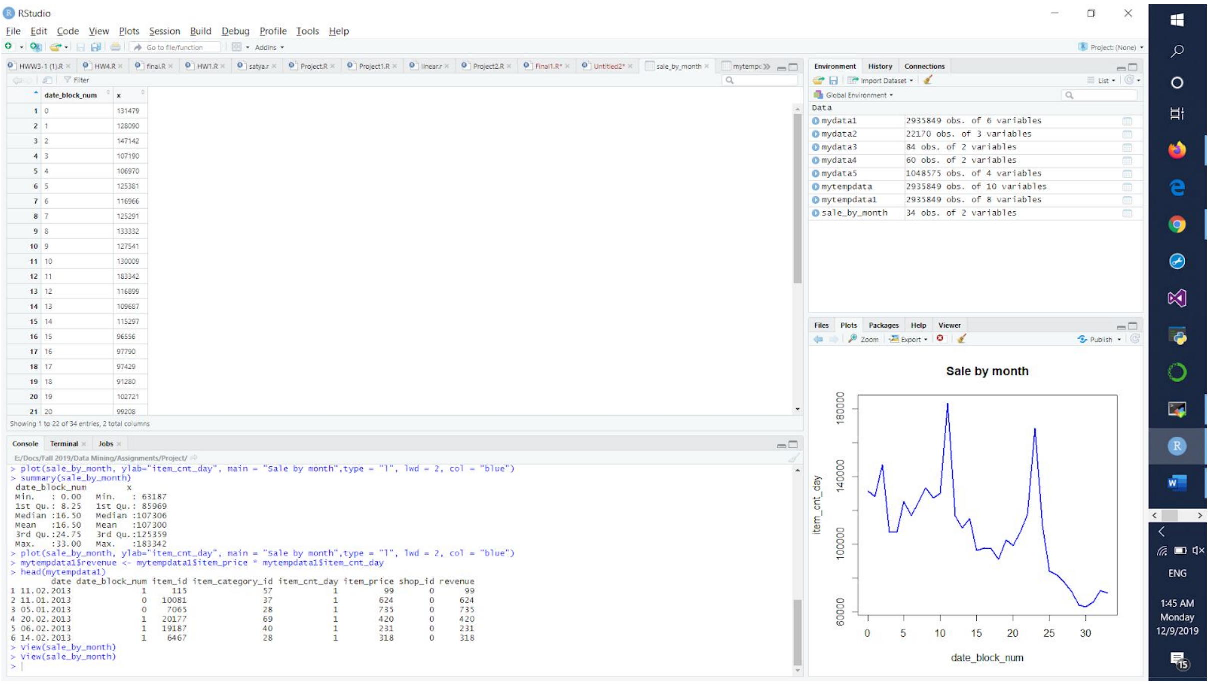


3. Feature Selection/ Engineering:

- We created a new data field Revenue which is the product of no. of products sold (item_cnt_day) and current price of an item (item_price).



- Created Sales_by_month dataset which consists of all the aggregated date_block_num and number of items sold for every consecutive month.



4. Modelling:

- **K Nearest Neighbours (KNN):**

  - **Training data:** The merged dataset(mytempdata1) dataset except October 2015.
  - **Testing data:** The merged dataset(mytemdata1) dataset for the month of October 2015.

```
# Training data - remove october 2015
mytempdata3 <- mytempdata1[!(mytempdata1$date_block_num == 33),]
any(mytempdata3$date_block_num == 33)
colnames(mytempdata3)
# Testing data (October 2015)
mytempdata4 <- mytempdata1[(mytempdata1$date_block_num == 33),]
colnames(mytempdata4)
```

  - **Normalizing data: We defined the normalize function to normalize data.**

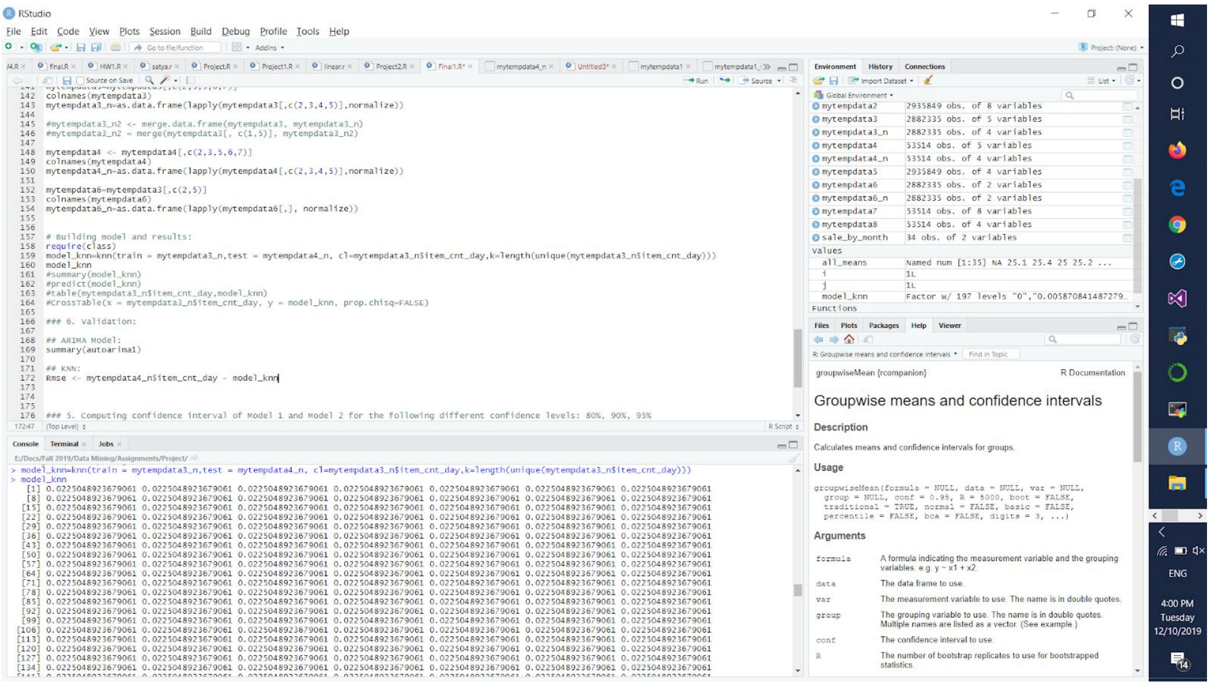
```
# Normalize function
normalize=function(x){
  return ((x-min(x))/(max(x)-min(x)))
}
```

    - Normalizing the datasets mytempdata3, mytempdata4, mytempdata6 using the normalizing function to mytempdata3_n, mytempdata4_n, mytempdata6_n.

```r
# Normalizing data
mytempdata3=mytempdata3[,c(2,3,5,6,7)]
colnames(mytempdata3)
mytempdata3_n=as.data.frame(lapply(mytempdata3[,c(2,3,4,5)],normalize))

mytempdata4 <- mytempdata4[,c(2,3,5,6,7)]
colnames(mytempdata4)
mytempdata4_n=as.data.frame(lapply(mytempdata4[,c(2,3,4,5)],normalize))

mytempdata6=mytempdata3[,c(2,5)]
colnames(mytempdata6)
mytempdata6_n=as.data.frame(lapply(mytempdata6[,], normalize))
```

- **Building Model and output:**
  - Following values shown are the predicted item_cnt_day values.

```r
# Building model and results:

require(class)
model_knn=knn(train = mytempdata3_n,test = mytempdata4_n, cl=mytempdata3_n$item_cnt_day,k=length(unique(mytempdata3_n$item_cnt_day)))
model_knn
```



- **ARIMA(Auto Regressive Integrated Moving Average):**
  - Arima is useful in our case since sales_by_month plot consists of moving averages and is not constant.
  - **Integrated** in the ARIMA refers to the difference in the timestamps.

e.g. Timestamp_1= Sales(February 2013 – January 2013)

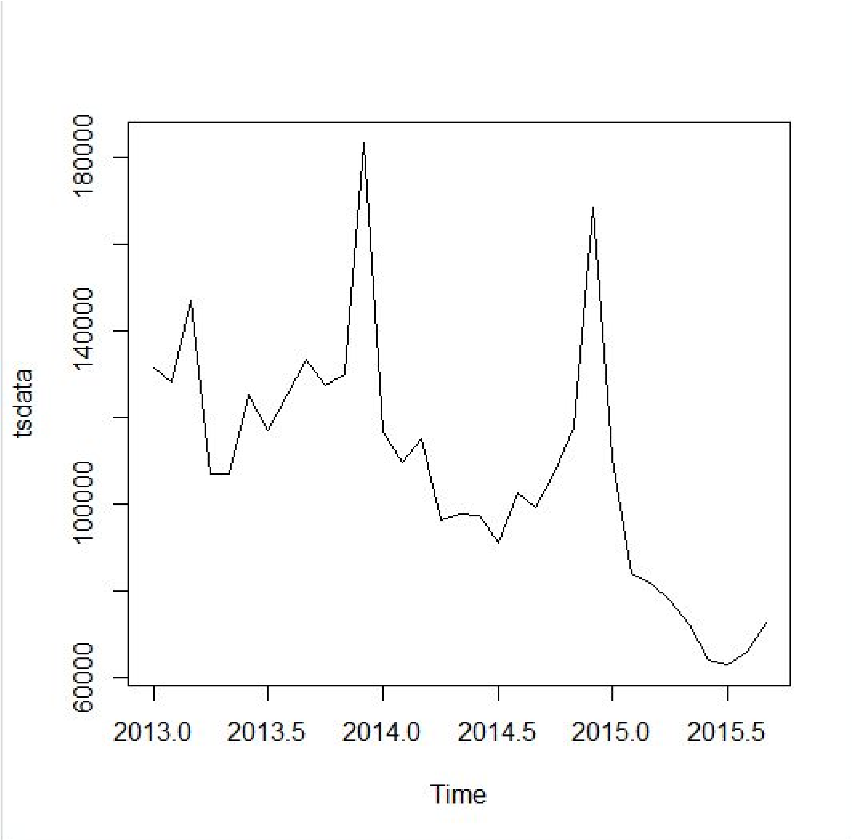
  - **Training data:** Sales_by_month dataset excluding October 2015.

```
train_data_arima <- sale_by_month [-34,]
```

- **Testing data:** Aggregated number of items sold count for the month of October 2015.

```
tsdata <- ts(train_data_arima$x, frequency = 12, start = c(2013,1))
```

- **Plot of test data:**

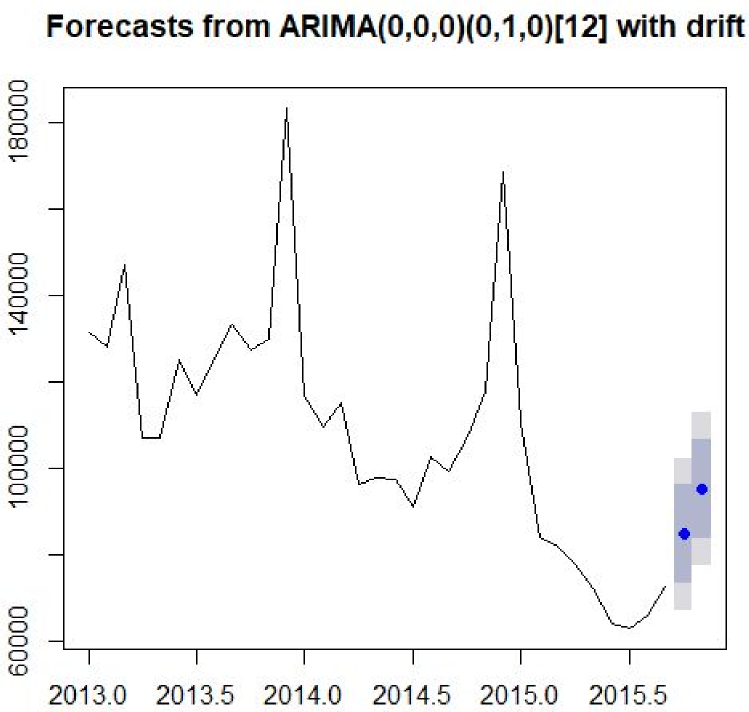


- **Training time for the data set :** 2.75 seconds.

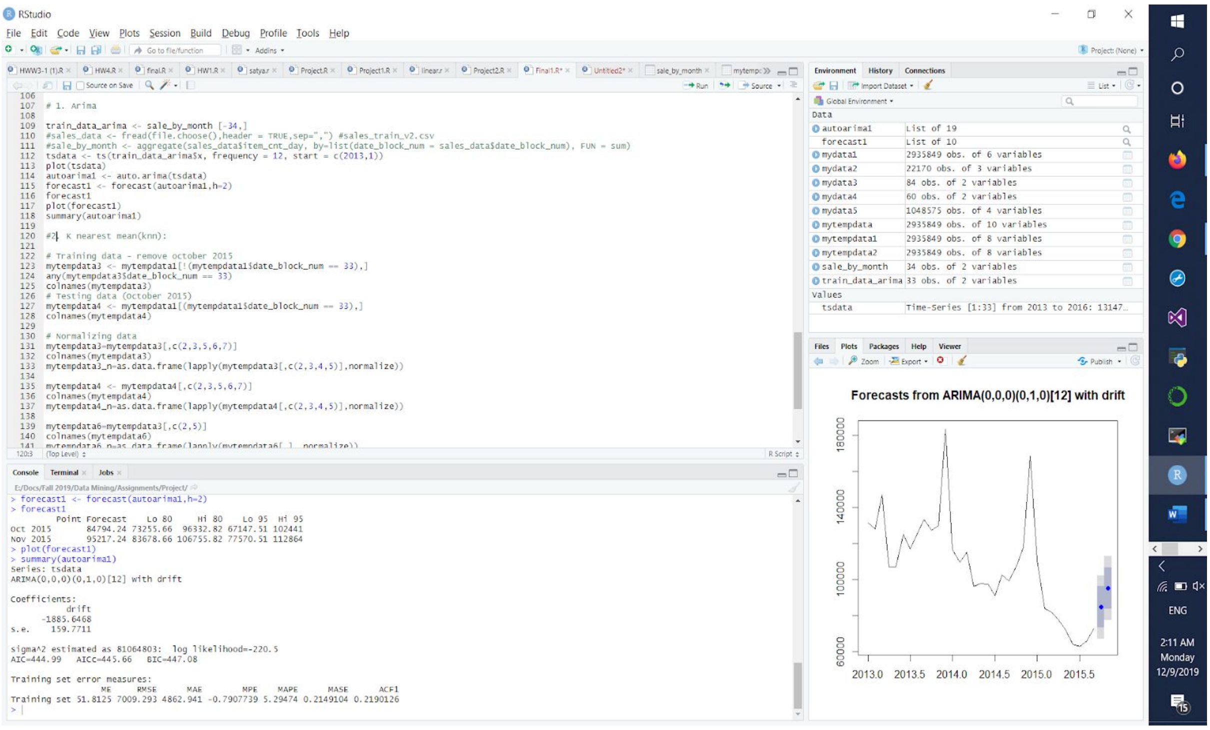
- **Actual forecasting results:**

```
> forecast1 <- forecast(autoarima1,h=2)
> forecast1
         Point Forecast    Lo 80     Hi 80    Lo 95  Hi 95
Oct 2015       84794.24 73255.66  96332.82 67147.51 102441
Nov 2015       95217.24 83678.66 106755.82 77570.51 112864
```

- **Plotting the Forecasts from ARIMA(0,0,0)(0,1,0)[12] with drift:**



- Here we can observe the actual and predicted values for months October 2015 and November 2015 in the graph.

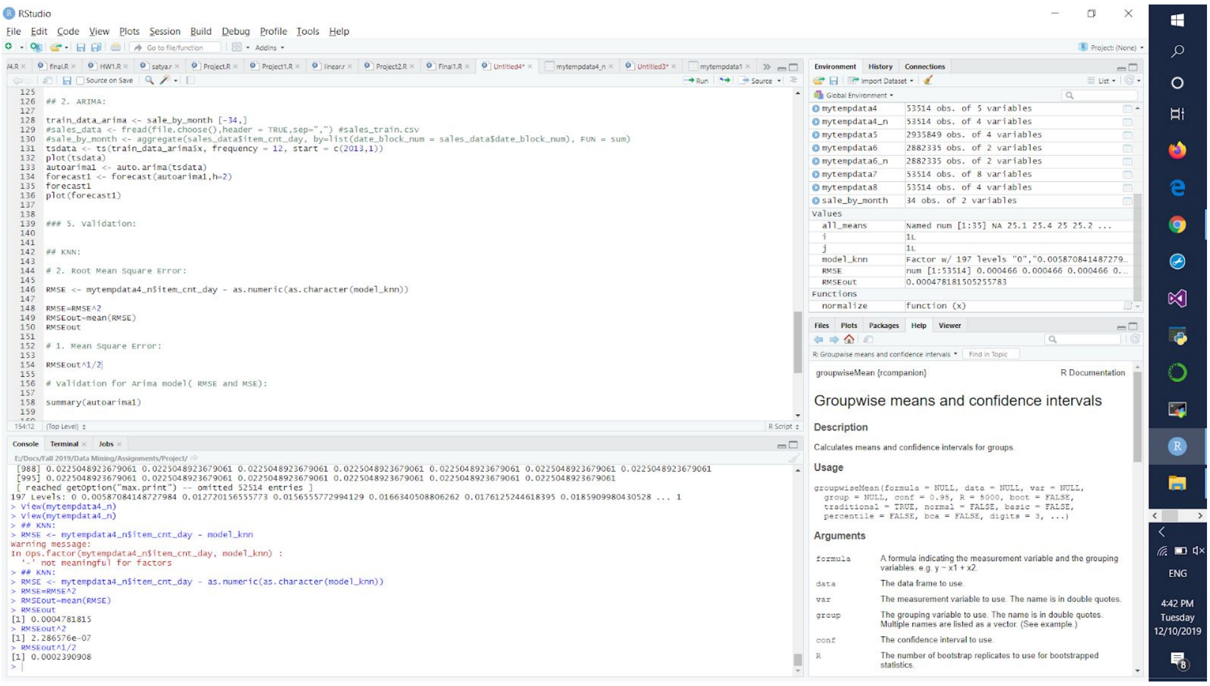


5. Validation:

- **KNN Model:**

- **Root Mean Square Error(RMSE):** We got the value around 4.781.
- **Mean Square Error(MSE):** We got this around 2.391

```
> ## KNN:
> RMSE <- mytempdata4_n$item_cnt_day - as.numeric(as.character(model_knn))
> RMSE=RMSE^2
> RMSEout=mean(RMSE)
> RMSEout
[1] 0.0004781815
> RMSEout^2
[1] 2.286576e-07
> RMSEout^1/2
[1] 0.0002390908
```



- **ARIMA Model:**

- ME (Mean Error): The mean error is the term that usually refers to the average of all the errors in a set.
- RMSE (Root Mean Square Error: It is a frequently used measure of the differences between values predicted by a model or an estimator and the values observed.
- MPE (Mean Percentage Error): It is the computed average of percentage errors by which forecasts of a model differ from actual values.
- MAPE (Mean Absolute Percentage Error): It is a measure of prediction accuracy of a forecasting method
- MASE (Mean Absolute Scaled Error): It is a measure of the accuracy of forecasts
- ACF1 (Autocorrelation of errors at lag 1): It is a measure of how much is the current value influenced by the previous values in a time series.
- Here, we observed following validation values for the Arima model:

```
Training set error measures:
                  ME     RMSE      MAE        MPE    MAPE      MASE      ACF1
Training set 51.8125 7009.293 4862.941 -0.7907739 5.29474 0.2149104 0.2190126
```

6. **Compute confidence interval of Model 1 and Model 2 for the following different confidence levels: 90%, 95%**

- **KNN Model:**

  - Confidence level (80%):

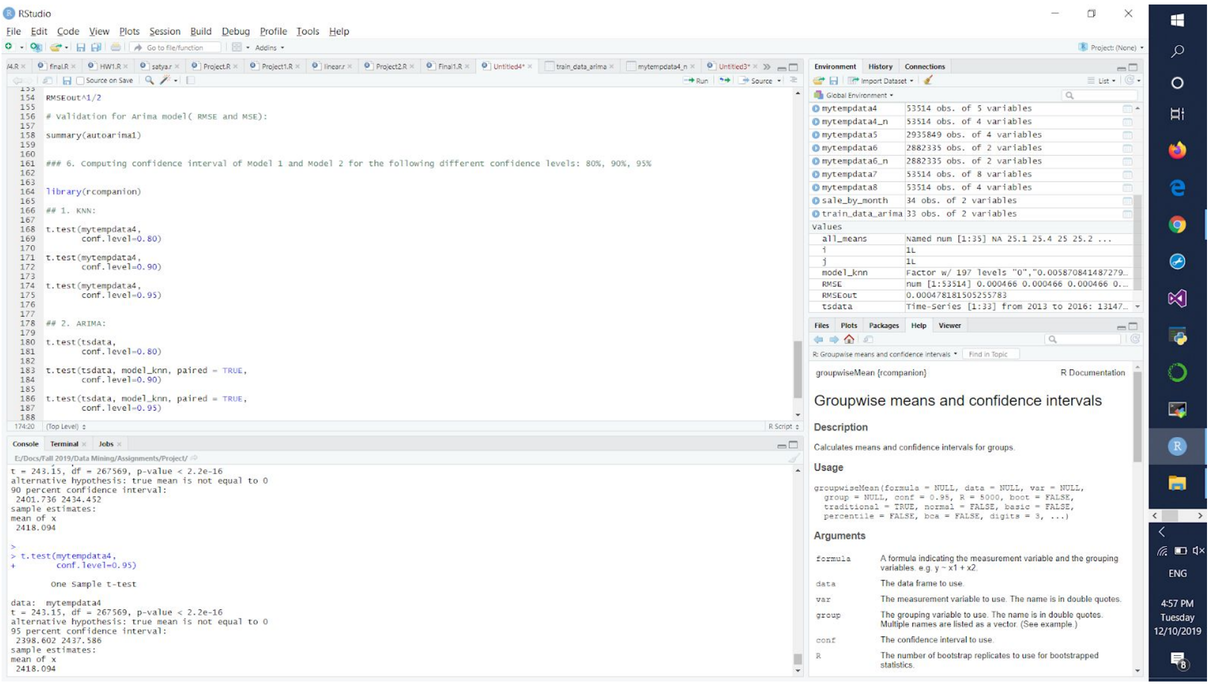
```
> t.test(mytempdata4,
+         conf.level=0.80)

        One Sample t-test

data:  mytempdata4
t = 243.15, df = 267569, p-value < 2.2e-16
alternative hypothesis: true mean is not equal to 0
80 percent confidence interval:
 2405.349 2430.839
sample estimates:
mean of x
 2418.094
```

  - Confidence level (90%):

```
> t.test(mytempdata4,
+         conf.level=0.90)

        One Sample t-test

data:  mytempdata4
t = 243.15, df = 267569, p-value < 2.2e-16
alternative hypothesis: true mean is not equal to 0
90 percent confidence interval:
 2401.736 2434.452
sample estimates:
mean of x
 2418.094
```

  - Confidence level (95%):

```
> t.test(mytempdata4,
+         conf.level=0.95)

        One Sample t-test

data:  mytempdata4
t = 243.15, df = 267569, p-value < 2.2e-16
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
 2398.602 2437.586
sample estimates:
mean of x
 2418.094
```


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
RMSEout^1/2
# Validation for Arima model(RMSE and MSE):
summary(autoarima1)
6. Computing confidence interval of Model 1 and Model 2 for the following different confidence levels: 80%, 90%, 95%
library(rcompanion)
1. KNN:
t.test(mytempdata4,
conf.level=0.80)
t.test(mytempdata4,
conf.level=0.90)
t.test(mytempdata4,
conf.level=0.95)
2. ARIMA:
t.test(tsdata,
conf.level=0.80)
t.test(tsdata, model_knn, paired = TRUE,
conf.level=0.90)
t.test(tsdata, model_knn, paired = TRUE,
conf.level=0.95)
Console Terminal Jobs
E:/Docs/Fall 2019/Data Mining/Assignments/Project/
t = 243.15, df = 267569, p-value < 2.2e-16
alternative hypothesis: true mean is not equal to 0
90 percent confidence interval:
2401.736 2434.452
sample estimates:
mean of x
2418.094
> t.test(mytempdata4,
+ conf.level=0.95)
One Sample t-test
data: mytempdata4
t = 243.15, df = 267569, p-value < 2.2e-16
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
2398.602 2437.586
sample estimates:
mean of x
2418.094
Environment History Connections
Global Environment
mytempdata4 53514 obs. of 5 variables
mytempdata4_n 53514 obs. of 4 variables
mytempdata5 2935849 obs. of 4 variables
mytempdata6 2882335 obs. of 2 variables
mytempdata6_n 2882335 obs. of 2 variables
mytempdata7 53514 obs. of 8 variables
mytempdata8 53514 obs. of 4 variables
sale_by_month 34 obs. of 2 variables
train_data_arima 33 obs. of 2 variables
Values
Files Plots Packages Help Viewer
groupwiseMean {rcompanion} R Documentation
Groupwise means and confidence intervals
Description
Calculates means and confidence intervals for groups.
Usage
Arguments
4:57 PM
Tuesday
12/10/2019

- **ARIMA Model:**
  - Confidence level (80%):

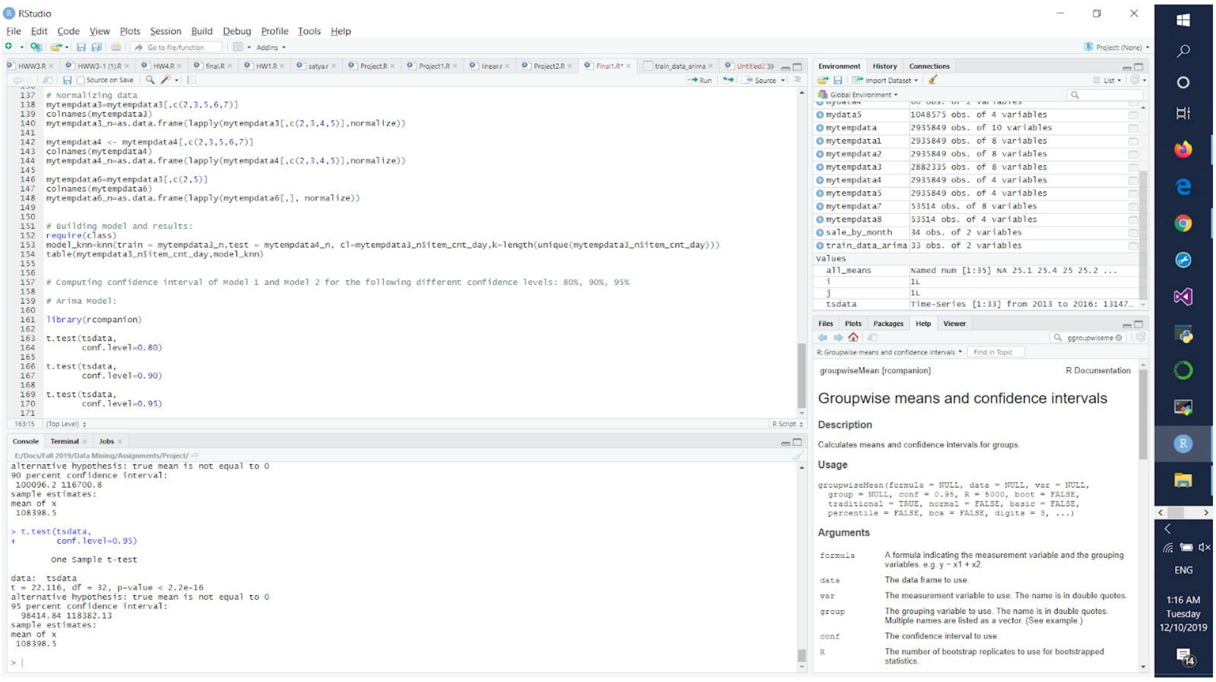
```
> t.test(tsdata,
+        conf.level=0.80)

        One Sample t-test

data:  tsdata
t = 22.116, df = 32, p-value < 2.2e-16
alternative hypothesis: true mean is not equal to 0
80 percent confidence interval:
 101984.8 114812.2
sample estimates:
mean of x
 108398.5
```

  - Confidence level (90%):

```
> t.test(tsdata,
+        conf.level=0.90)

        One Sample t-test

data:  tsdata
t = 22.116, df = 32, p-value < 2.2e-16
alternative hypothesis: true mean is not equal to 0
90 percent confidence interval:
 100096.2 116700.8
sample estimates:
mean of x
 108398.5
```

  - Confidence level (95%):

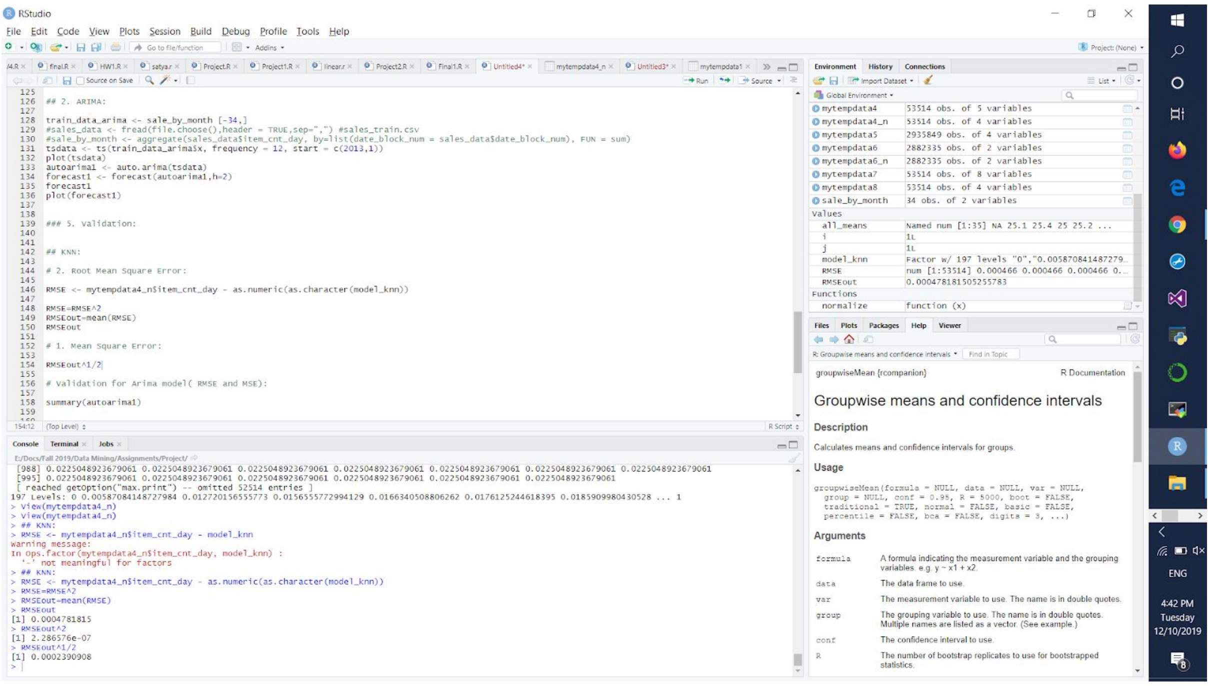
```
> t.test(tsdata,
+        conf.level=0.95)

        One Sample t-test

data:  tsdata
t = 22.116, df = 32, p-value < 2.2e-16
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
  98414.84 118382.13
sample estimates:
mean of x
 108398.5
```


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
HWW3.R HWW3-1 (1).R HW4.R final.R HW1.R satya.r Project.R Project1.R linear.r Project2.R Final1.R* train_data_arima Untitled2
# Normalizing data
mytempdata3=mytempdata3[,c(2,3,5,6,7)]
colnames(mytempdata3)
mytempdata3_n=as.data.frame(lapply(mytempdata3[,c(2,3,4,5)],normalize))
mytempdata4 <- mytempdata4[,c(2,3,5,6,7)]
colnames(mytempdata4)
mytempdata4_n=as.data.frame(lapply(mytempdata4[,c(2,3,4,5)],normalize))
mytempdata6=mytempdata3[,c(2,5)]
colnames(mytempdata6)
mytempdata6_n=as.data.frame(lapply(mytempdata6[,], normalize))
# Building model and results:
require(class)
model_knn=knn(train = mytempdata3_n,test = mytempdata4_n, cl=mytempdata3_n$item_cnt_day,k=length(unique(mytempdata3_n$item_cnt_day)))
table(mytempdata3_n$item_cnt_day,model_knn)
# Computing confidence interval of Model 1 and Model 2 for the following different confidence levels: 80%, 90%, 95%
# Arima Model:
library(rcompanion)
t.test(tsdata,
conf.level=0.80)
t.test(tsdata,
conf.level=0.90)
t.test(tsdata,
conf.level=0.95)
Console Terminal Jobs
E:/Docs/Fall 2019/Data Mining/Assignments/Project/
alternative hypothesis: true mean is not equal to 0
90 percent confidence interval:
100096.2 116700.8
sample estimates:
mean of x
108398.5
> t.test(tsdata,
+ conf.level=0.95)
One Sample t-test
data: tsdata
t = 22.116, df = 32, p-value < 2.2e-16
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
98414.84 118382.13
sample estimates:
mean of x
108398.5
Environment History Connections
Import Dataset
Global Environment
mydata5 1048575 obs. of 4 variables
mytempdata 2935849 obs. of 10 variables
mytempdata1 2935849 obs. of 8 variables
mytempdata2 2935849 obs. of 8 variables
mytempdata3 2882335 obs. of 8 variables
mytempdata4 2935849 obs. of 4 variables
mytempdata5 2935849 obs. of 4 variables
mytempdata7 53514 obs. of 8 variables
mytempdata8 53514 obs. of 4 variables
sale_by_month 34 obs. of 2 variables
train_data_arima 33 obs. of 2 variables
Values
all_means Named num [1:35] NA 25.1 25.4 25 25.2 ...
i 1L
j 1L
tsdata Time-Series [1:33] from 2013 to 2016: 13147...
Files Plots Packages Help Viewer
R: Groupwise means and confidence intervals
groupwiseMean {rcompanion} R Documentation
Groupwise means and confidence intervals
Description
Calculates means and confidence intervals for groups.
Usage
groupwiseMean(formula = NULL, data = NULL, var = NULL, group = NULL, conf = 0.95, R = 5000, boot = FALSE, traditional = TRUE, normal = FALSE, basic = FALSE, percentile = FALSE, bca = FALSE, digits = 3, ...)
Arguments
formula A formula indicating the measurement variable and the grouping variables. e.g. y ~ x1 + x2.
data The data frame to use.
var The measurement variable to use. The name is in double quotes.
group The grouping variable to use. The name is in double quotes. Multiple names are listed as a vector. (See example.)
conf The confidence interval to use.
R The number of bootstrap replicates to use for bootstrapped statistics.
ENG
1:16 AM
Tuesday
12/10/2019

7. Comparison between ARIMA and KNN model:

1. KNN Model:

- **Error:**
  a. **Root Mean Square Error(RMSE):** We got the value around 4.781.
  b. **Mean Square Error(MSE):** We got this around 2.391



a. Efficiency in training time (scalability):

Training time for the data set : 5 minutes.

2. ARIMA Model:

b. Errors:

a. ME (Mean Error): The mean error is the term that usually refers to the average of all the errors in a set.
b. RMSE (Root Mean Square Error: It is a frequently used measure of the differences between values predicted by a model or an estimator and the values observed.

c. MPE (Mean Percentage Error): It is the computed average of percentage errors by which forecasts of a model differ from actual values.
d. MAPE (Mean Absolute Percentage Error): It is a measure of prediction accuracy of a forecasting method
e. MASE (Mean Absolute Scaled Error): It is a measure of the accuracy of forecasts
f. ACF1 (Autocorrelation of errors at lag 1): It is a measure of how much is the current value influenced by the previous values in a time series.

g. Here, we observed following validation values for the Arima model:

```
Training set error measures:
                    ME     RMSE      MAE        MPE    MAPE      MASE      ACF1
Training set 51.8125 7009.293 4862.941 -0.7907739 5.29474 0.2149104 0.2190126
```

c. Efficiency in training time (scalability):

Training time for the data set : 2.75 seconds.