BIG MARKET SALES PREDICTION

DATA VISUALIZATION
PROJECT REPORT

SLOT: G1

Submitted By: Aime Gupta (17BCE0097)

Pamnani Chintan Rajesh (17BCE0471)

Name of faculty: PROF.RAJKUMAR R.

(SCHOOL OF COMPUTER SCIENCE AND ENGINEERING)

CERTIFICATE

This is to certify that the project report entitled "**BIG MARKET SALES PREDICTION**" is prepared & submitted by Aime Gupta (17BCE0097), Pamnani Chintan Rajesh (17BCE0471) is an authentic work carried out by the members of the team under my supervision & guidance. It has been found satisfactory in terms of scope, quality & presentation as partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Computer Science & Engineering in VIT University, India.

Guide

Prof. RAJKUMAR R

OBJECTIVE:

- The aim is to build a predictive model and find out the sales of each product at a particular store and hence predict the sales of supermarket according to the sample supermarket dataset. The idea is to find out the properties of a product, and store which impacts the sales of a product. Using this model, we will try to understand the properties of products and stores which play a key role in increasing sales
- We will also various visual plots between the data to better understand it.
- Algorithm we will use to build our models are:
  - Linear Regression
  - Ridge Regression
  - Random Forest Regression

SCREEN SHOTS OF THE INTERFACE:

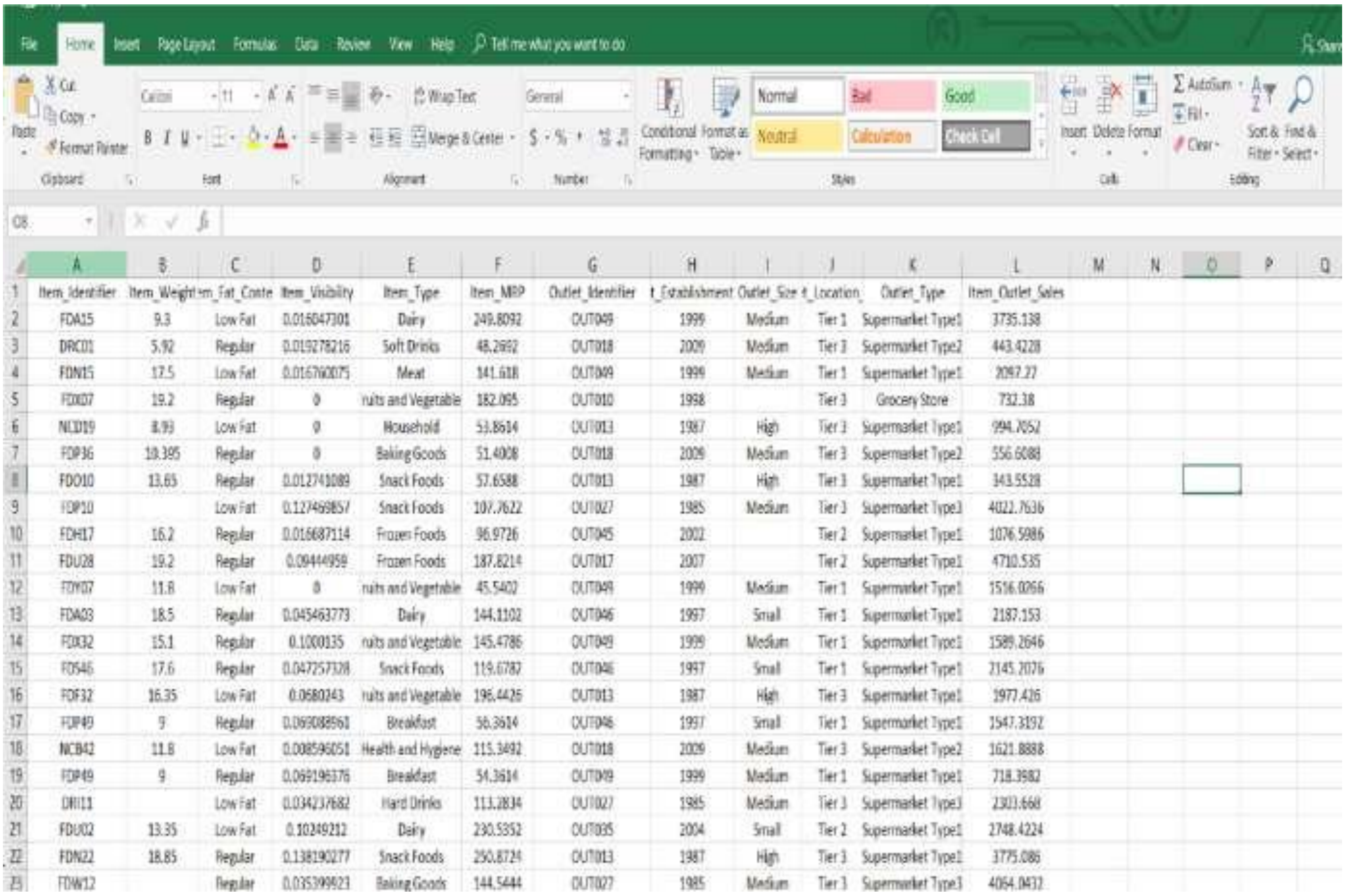
1) Hypothesis Generation:

It involves understanding the problem in detail by brainstorming as many factors as possible which can impact the outcome. It is done by understanding the problem statement thoroughly and before looking at the data

Loading Packages

```
library(data.table) # used for reading and manipulation of data
library(dplyr)      # used for data manipulation and joining
library(ggplot2)    # used for ploting   library(caret)      # used
for modeling library(corrplot) # used for making correlation
plot  library(xgboost)    # used for building XGBoost model
library(cowplot)    # used for combining multiple plots
library(magritrr) # used for pipe operator(%>%)
```

- Dataset used:



	A	B	C	D	E	F	G	H	I	J	K	L
1	Item_Identifier	Item_Weight	em_Fat_Conte	Item_Visibility	Item_Type	Item_MRP	Outlet_Identifier	t_Establishment	Outlet_Size	t_Location	Outlet_Type	Item_Outlet_Sales
2	FDA15	9.3	Low Fat	0.016047301	Dairy	249.8092	OUT049	1999	Medium	Tier 1	Supermarket Type1	3735.138
3	DRC01	5.92	Regular	0.019278216	Soft Drinks	48.2692	OUT018	2009	Medium	Tier 3	Supermarket Type2	443.4228
4	FDN15	17.5	Low Fat	0.016760075	Meat	141.618	OUT049	1999	Medium	Tier 1	Supermarket Type1	2097.27
5	FDX07	19.2	Regular	0	ruits and Vegetable	182.095	OUT010	1998		Tier 3	Grocery Store	732.38
6	NCD19	8.93	Low Fat	0	Household	53.8614	OUT013	1987	High	Tier 3	Supermarket Type1	994.7052
7	FDP36	10.395	Regular	0	Baking Goods	51.4008	OUT018	2009	Medium	Tier 3	Supermarket Type2	556.6088
8	FDO10	13.65	Regular	0.012741089	Snack Foods	57.6588	OUT013	1987	High	Tier 3	Supermarket Type1	343.5528
9	FDP10		Low Fat	0.127469857	Snack Foods	107.7622	OUT027	1985	Medium	Tier 3	Supermarket Type3	4022.7636
10	FDH17	16.2	Regular	0.016687114	Frozen Foods	96.9726	OUT045	2002		Tier 2	Supermarket Type1	1076.5986
11	FDU28	19.2	Regular	0.09444959	Frozen Foods	187.8214	OUT017	2007		Tier 2	Supermarket Type1	4710.535
12	FDY07	11.8	Low Fat	0	ruits and Vegetable	45.5402	OUT049	1999	Medium	Tier 1	Supermarket Type1	1516.0266
13	FDA03	18.5	Regular	0.045463773	Dairy	144.1102	OUT046	1997	Small	Tier 1	Supermarket Type1	2187.153
14	FDX32	15.1	Regular	0.1000135	ruits and Vegetable	145.4786	OUT049	1999	Medium	Tier 1	Supermarket Type1	1589.2646
15	FDS46	17.6	Regular	0.047257328	Snack Foods	119.6782	OUT046	1997	Small	Tier 1	Supermarket Type1	2145.2076
16	FDF32	16.35	Low Fat	0.0680243	ruits and Vegetable	196.4426	OUT013	1987	High	Tier 3	Supermarket Type1	1977.426
17	FDP49	9	Regular	0.069088961	Breakfast	56.3614	OUT046	1997	Small	Tier 1	Supermarket Type1	1547.3192
18	NCB42	11.8	Low Fat	0.008596051	Health and Hygiene	115.3492	OUT018	2009	Medium	Tier 3	Supermarket Type2	1621.8888
19	FDP49	9	Regular	0.069196376	Breakfast	54.3614	OUT049	1999	Medium	Tier 1	Supermarket Type1	718.3982
20	DRI11		Low Fat	0.034237682	Hard Drinks	113.2834	OUT027	1985	Medium	Tier 3	Supermarket Type3	2303.668
21	FDU02	13.35	Low Fat	0.10249212	Dairy	230.5352	OUT035	2004	Small	Tier 2	Supermarket Type1	2748.4224
22	FDN22	18.85	Regular	0.138190277	Snack Foods	250.8724	OUT013	1987	High	Tier 3	Supermarket Type1	3775.086
23	FDW12		Regular	0.035399923	Baking Goods	144.5444	OUT027	1985	Medium	Tier 3	Supermarket Type3	4064.0432

```
> dim(train)
[1] 8523   12
> dim(test)
[1] 5681   12
> names(train)
 [1] "Item_Identifier"           "Item_Weight"               "Item_Fat_Content"
 [4] "Item_Visibility"           "Item_Type"                 "Item_MRP"
 [7] "Outlet_Identifier"         "Outlet_Establishment_Year" "Outlet_Size"
[10] "Outlet_Location_Type"      "Outlet_Type"               "Item_Outlet_Sales"
> names(test)
 [1] "Item_Identifier"           "Item_Weight"               "Item_Fat_Content"
 [4] "Item_Visibility"           "Item_Type"                 "Item_MRP"
 [7] "Outlet_Identifier"         "Outlet_Establishment_Year" "Outlet_Size"
[10] "Outlet_Location_Type"      "Outlet_Type"               "Item_Outlet_Sales"
> |
```

```
> str(train)
'data.frame':	8523 obs. of  12 variables:
 $ Item_Identifier          : Factor w/ 1559 levels "DRA12","DRA24",..: 157 9 663 1122 1298
759 697 739 441 991 ...
 $ Item_Weight              : num  9.3 5.92 17.5 19.2 8.93 ...
 $ Item_Fat_Content         : Factor w/ 5 levels "LF","low fat",..: 3 5 3 5 3 5 5 3 5 5 ...
 $ Item_Visibility          : num  0.016 0.0193 0.0168 0 0 ...
 $ Item_Type                : Factor w/ 16 levels "Baking Goods",..: 5 15 11 7 10 1 14 14 6
6 ...
 $ Item_MRP                 : num  249.8 48.3 141.6 182.1 53.9 ...
 $ Outlet_Identifier        : Factor w/ 10 levels "OUT010","OUT013",..: 10 4 10 1 2 4 2 6 8
3 ...
 $ Outlet_Establishment_Year: int  1999 2009 1999 1998 1987 2009 1987 1985 2002 2007 ...
 $ Outlet_Size              : Factor w/ 4 levels "","High","Medium",..: 3 3 3 1 2 3 2 3 1 1
...
 $ Outlet_Location_Type     : Factor w/ 3 levels "Tier 1","Tier 2",..: 1 3 1 3 3 3 3 3 2 2
...
 $ Outlet_Type              : Factor w/ 4 levels "Grocery Store",..: 2 3 2 1 2 3 2 4 2 2 ...
 $ Item_Outlet_Sales        : num  3735 443 2097 732 995 ...
> str(test)
'data.frame':	5681 obs. of  12 variables:
 $ Item_Identifier          : Factor w/ 1543 levels "DRA12","DRA24",..: 1104 1068 1407 810 1
185 462 605 267 669 171 ...
 $ Item_Weight              : num  20.75 8.3 14.6 7.32 NA ...
 $ Item_Fat_Content         : Factor w/ 5 levels "LF","low fat",..: 3 4 3 3 5 5 5 3 5 3 ...
 $ Item_Visibility          : num  0.00756 0.03843 0.09957 0.01539 0.1186 ...
 $ Item_Type                : Factor w/ 16 levels "Baking Goods",..: 14 5 12 14 5 7 1 1 14 1
...
 $ Item_MRP                 : num  107.9 87.3 241.8 155 234.2 ...
 $ Outlet_Identifier        : Factor w/ 10 levels "OUT010","OUT013",..: 10 3 1 3 6 9 4 6 8 3
...
 $ Outlet_Establishment_Year: int  1999 2007 1998 2007 1985 1997 2009 1985 2002 2007 ...
 $ Outlet_Size              : Factor w/ 4 levels "","High","Medium",..: 3 1 1 1 3 4 3 3 1 1
...
 $ Outlet_Location_Type     : Factor w/ 3 levels "Tier 1","Tier 2",..: 1 2 3 2 3 1 3 3 2 2
...
 $ Outlet_Type              : Factor w/ 4 levels "Grocery Store",..: 2 2 1 2 4 2 3 4 2 2 ...
```

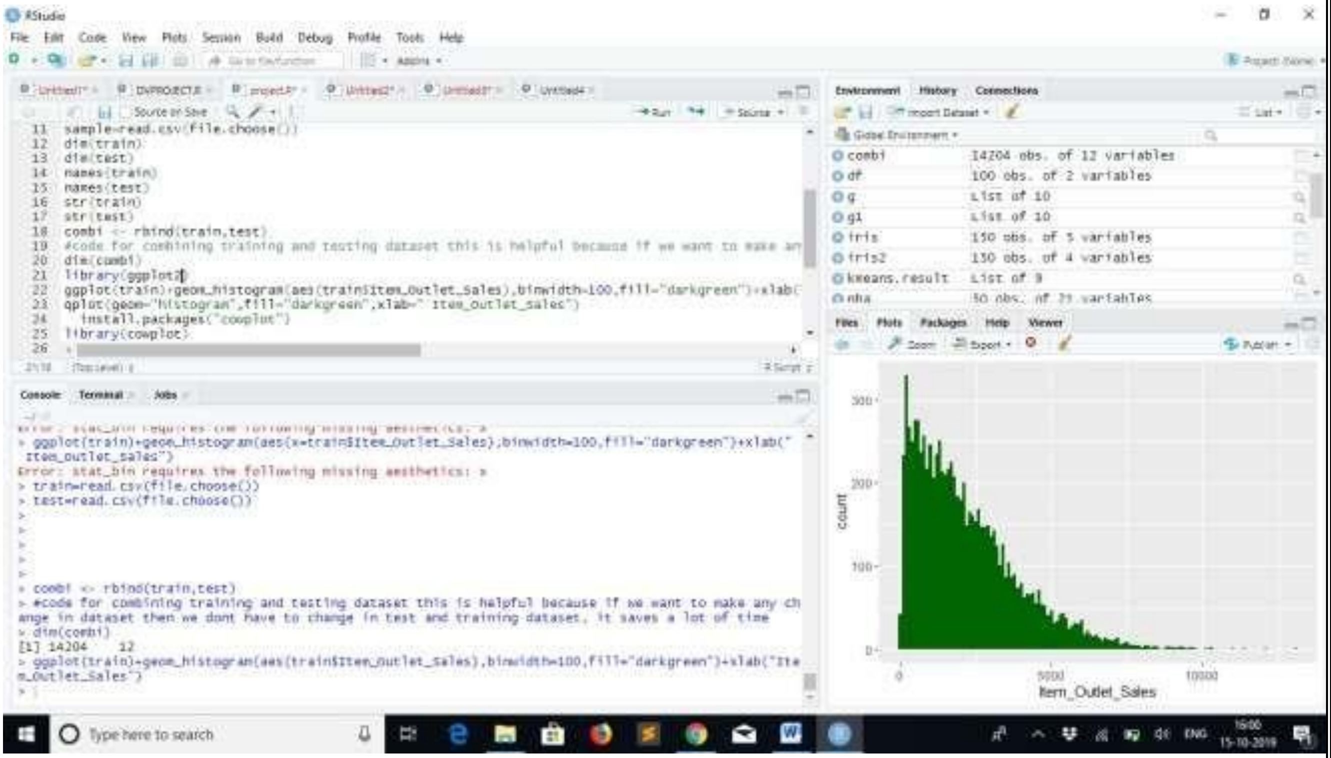
Data visualisation:

```
> combi <- rbind(train,test)
> #code for combining training and testing dataset this is helpful because if we want to make any change in dataset then w
e dont have to change in test and training dataset, it saves a lot of time
> dim(combi)
[1] 14204    12
> ggplot(train)+geom_histogram(aes(train$Item_Outlet_Sales),binwidth=100,fill="darkgreen")+xlab("Item_Outlet_Sales")
> 
```

DATASET:

- The dataset has 8523 rows (instances) of 12 variables (attributes).
- It is taken from kaggle.com.
- Data attributes : Item_Identifier, Item_Weight, Item_Type, Item_MRP, etc.

HISTOGRAM(GGPLOT2)- ITEM_OUTLET_SALES

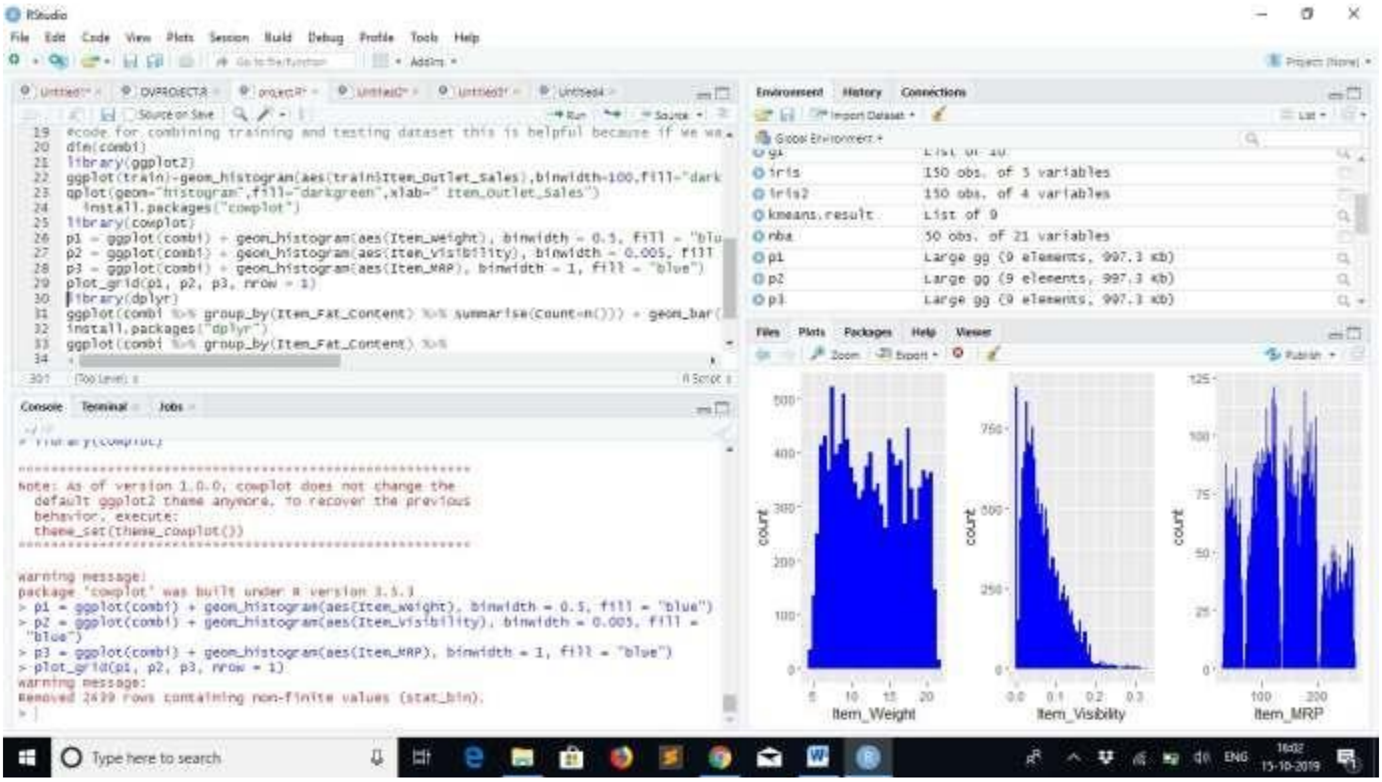


> install.packages("ggplot2")

> library(ggplot2)

> ggplot(train) + geom_histogram(aes(train$Item_Outlet_Sales), binwidth=100, fill="darkgreen") + xlab("Item_Outlet_Sales")

HISTOGRAM(COWPLOT)- ITEM_WEIGHT, ITEM_VISIBILITY, ITEM_MRP

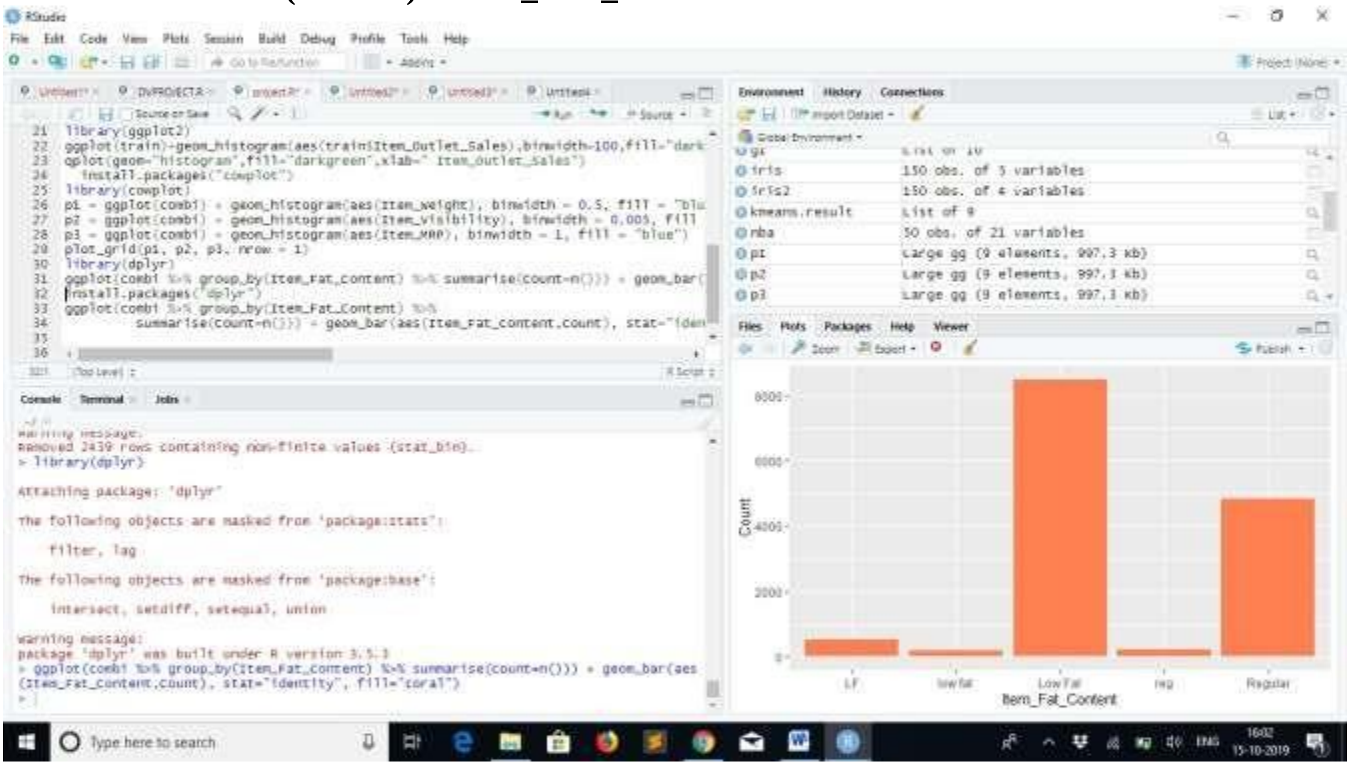


```
> install.packages("cowplot")
> library(cowplot)

>  p1 = ggplot(combi) + geom_histogram(aes(Item_Weight), binwidth = 0.5, fill = "blue") >
p2 = ggplot(combi) + geom_histogram(aes(Item_Visibility), binwidth = 0.005, fill = "blue")
> p3 = ggplot(combi) + geom_histogram(aes(Item_MRP), binwidth = 1, fill = "blue")

> plot_grid(p1, p2, p3, nrow = 1)
```

Observation:

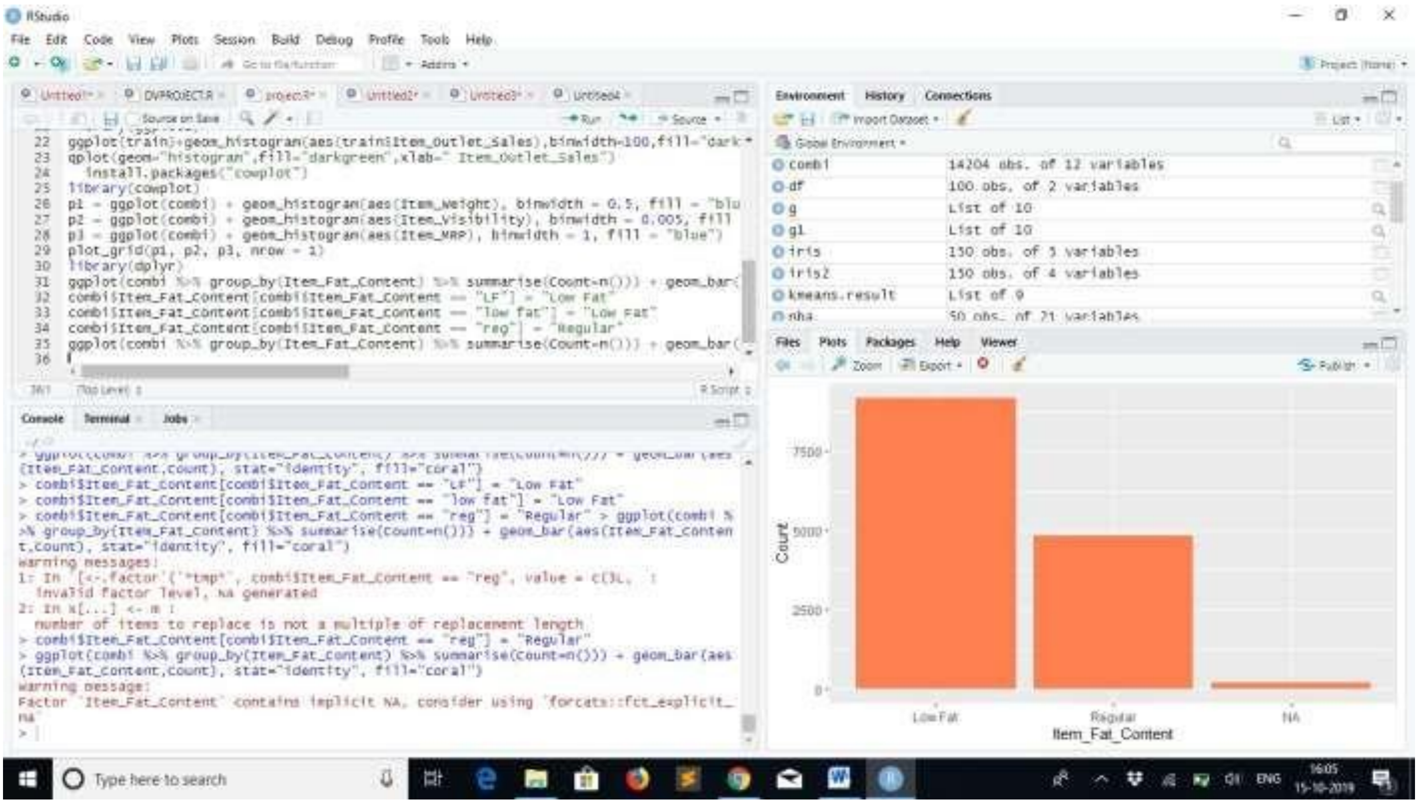
- There seems to be no clear-cut pattern in Item_Weight.
- Item_Visibility is right-skewed and should be transformed to curb its skewness.
- We can clearly see 4 different distributions for Item_MRP. It is an interesting insight.

BAR GRAPH(DPLYR)- ITEM_FAT_CONTENT




```
> install.packages("dplyr")
> library(dplyr)

> ggplot(combi %>% group_by(Item_Fat_Content) %>% summarise(Count=n())) +
geom_bar(aes(Item_Fat_Content,Count), stat="identity",  fill="coral")
```

BAR GRAPH-ITEM_FAT_CONTENT

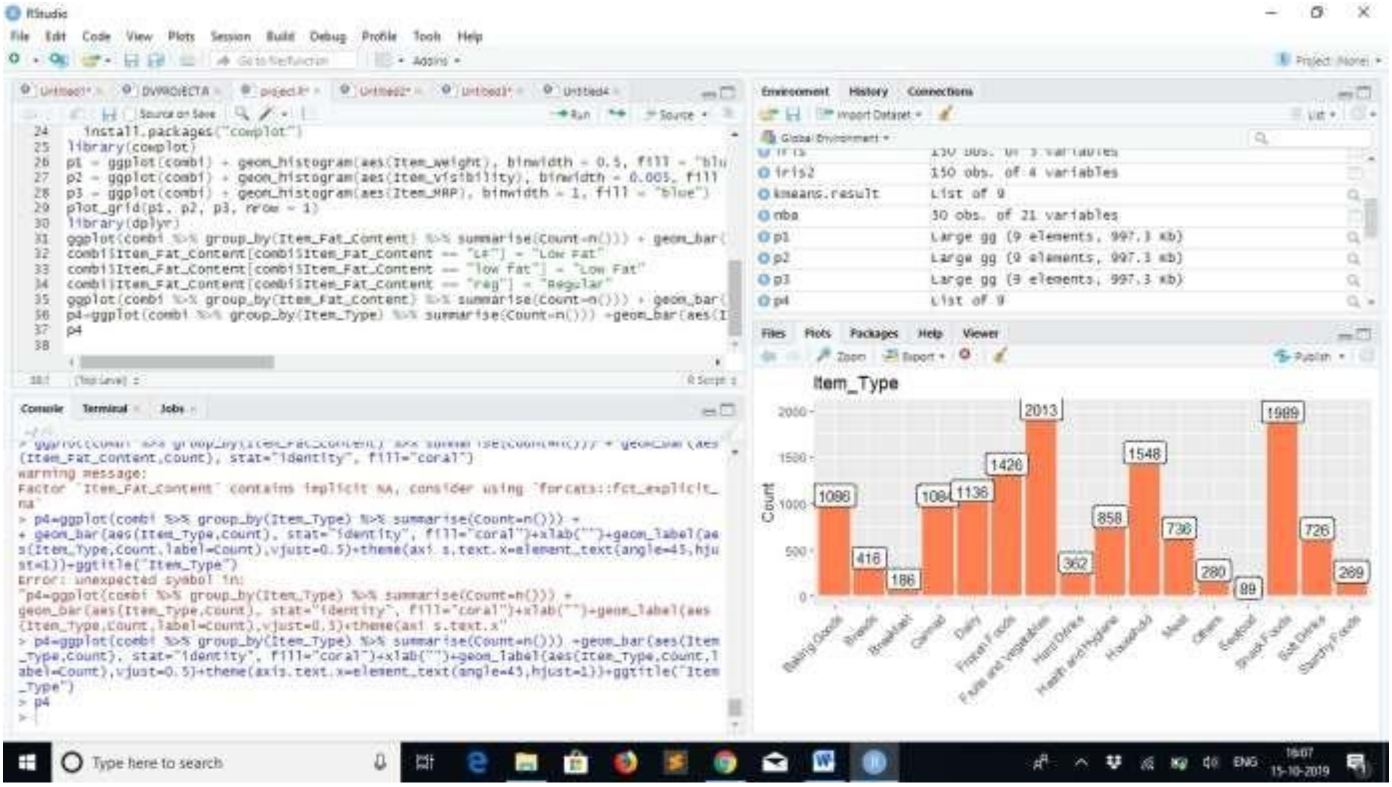


>combi$Item_Fat_Content[combi$Item_Fat_Content == "LF"] = "Low Fat"

>combi$Item_Fat_Content[combi$Item_Fat_Content == "low fat"] = "Low Fat"

>combi$Item_Fat_Content[combi$Item_Fat_Content == "reg"] = "Regular" > ggplot(combi %>% group_by(Item_Fat_Content) %>% summarise(Count=n())) + geom_bar(aes(Item_Fat_Content,Count), stat="identity", fill="coral")

BAR GRAPH- ITEM_TYPE

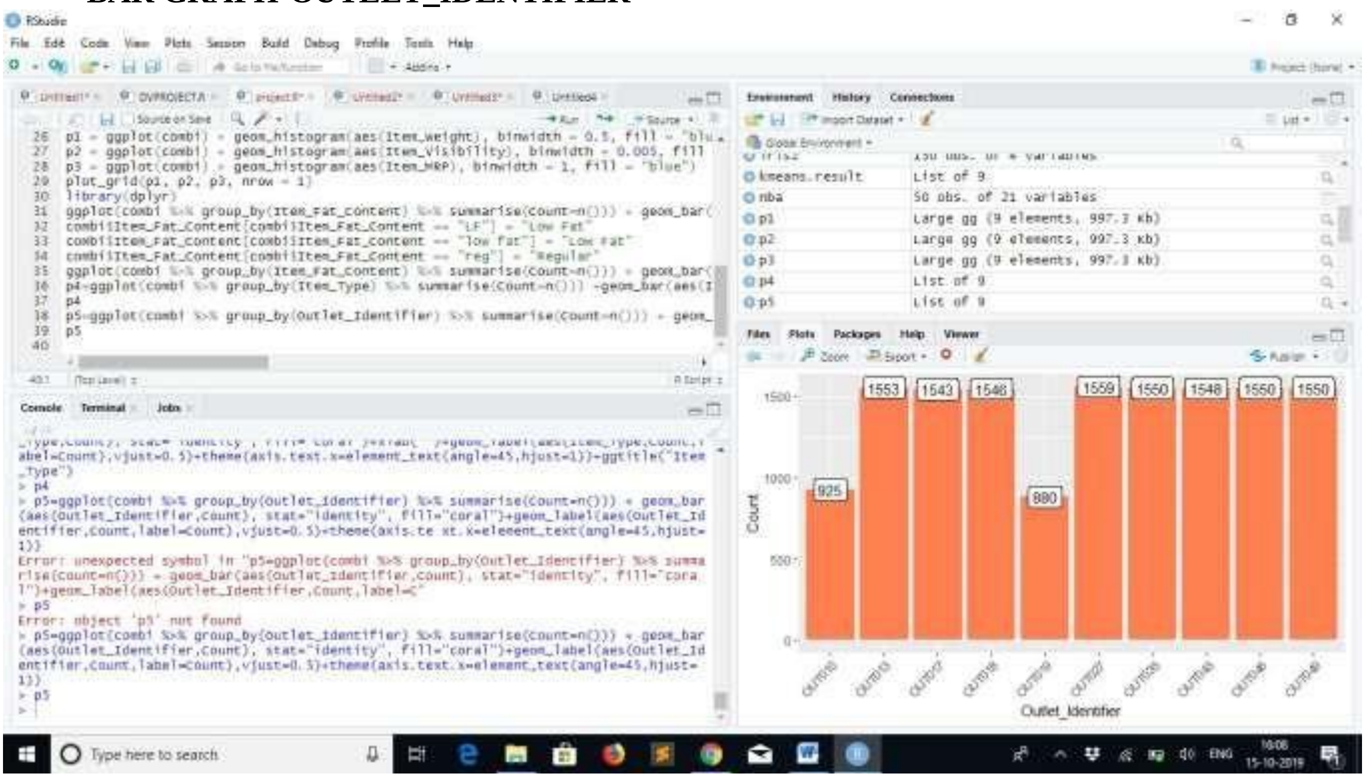


> p4=ggplot(combi %>% group_by(Item_Type) %>% summarise(Count=n())) +

geom_bar(aes(Item_Type,Count), stat="identity", fill="coral")+xlab("")+geom_label(aes(Item_Type,Count,label=Count),vjust=0.5)+theme(axis.text.x=element_text(angle=45,hjust=1))+ggtitle("Item_Type")

> p4

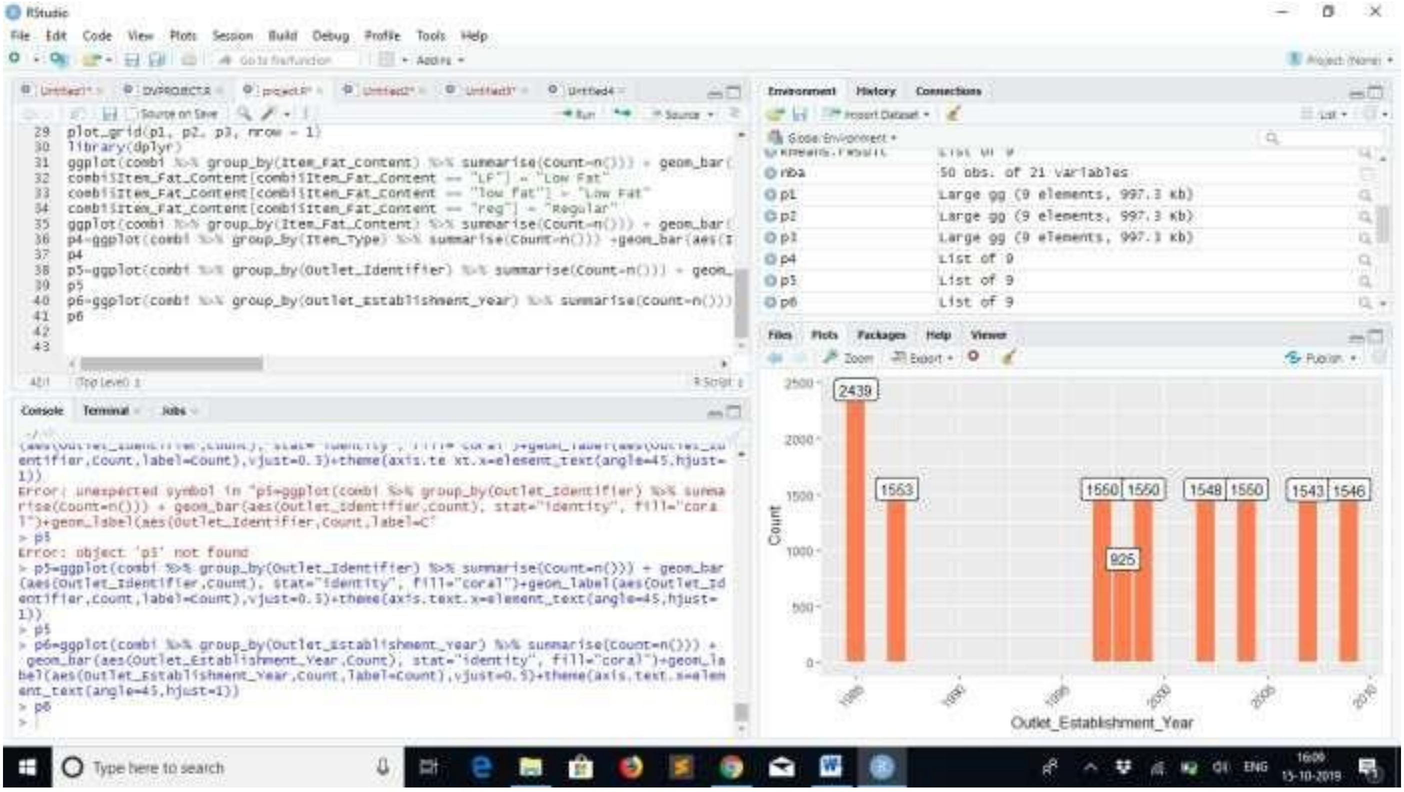
BAR GRAPH-OUTLET_IDENTIFIER



> p5=ggplot(combi %>% group_by(Outlet_Identifier) %>% summarise(Count=n())) + geom_bar(aes(Outlet_Identifier,Count), stat="identity", fill="coral")+geom_label(aes(Outlet_Identifier,Count,label=Count),vjust=0.5)+theme(axis.text.x=element_text(angle=45,hjust=1))

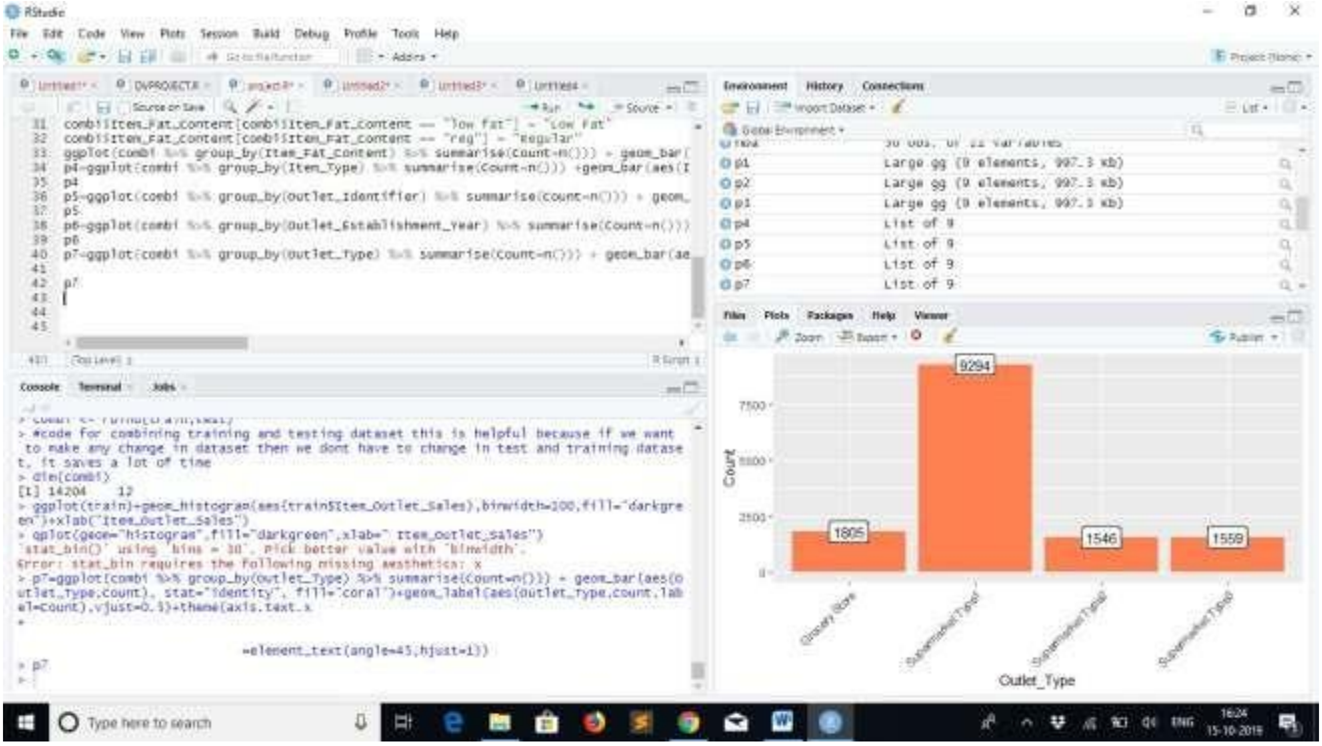
> p5

BAR GRAPH-OUTLET_ESTABLISHMENT_YEAR




```
> p6=ggplot(combi %>% group_by(Outlet_Establishment_Year) %>% summarise(Count=n()))
+ geom_bar(aes(Outlet_Establishment_Year,Count), stat="identity",
fill="coral")+geom_label(aes(Outlet_Establishment_Year,Count,label=Count),vjust=0.5)+the
me(axis.text.x=element_text(angle=45,hjust=1))
> p6
```

BAR GRAPH-OUTET_TYPE




```
> p7=ggplot(combi %>% group_by(Outlet_Type) %>% summarise(Count=n())) +
geom_bar(aes(Outlet_Type,Count), stat="identity",
fill="coral")+geom_label(aes(Outlet_Type,Count,label=Count),vjust=0.5)+theme(axis.text.x
=element_text(angle=45,hjust=1))
> p7
```

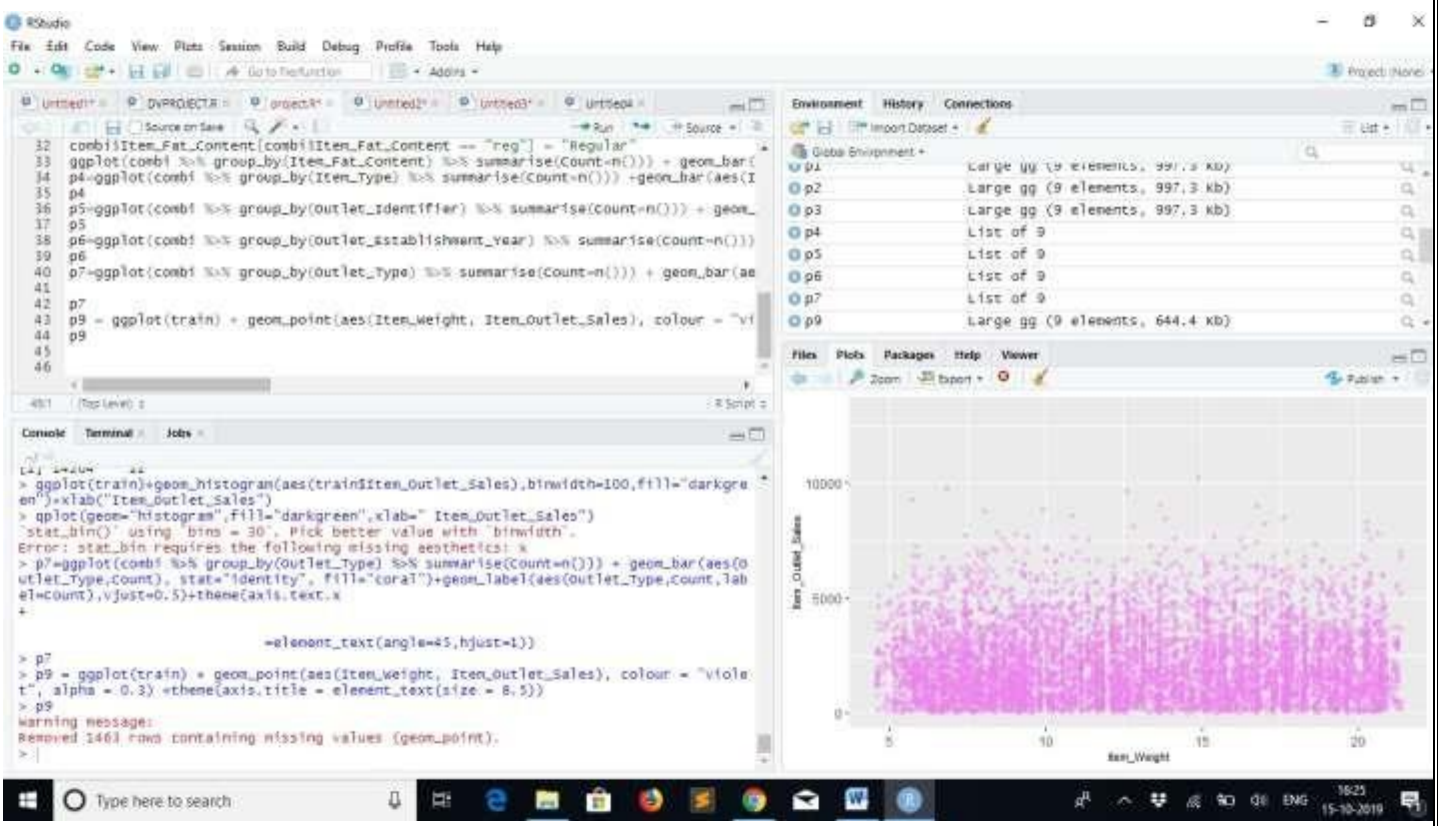
Observations:

- Lesser number of observations in the data for the outlets established in the year 1998 as compared to the other years.
- Supermarket Type 1 seems to be the most popular category of Outlet_Type.

We will make use of **scatter plots** for the continuous or numeric variables and **violin plots** for the categorical variables.

SCATTER PLOT-ITEM_WEIGHT

```
> p9 = ggplot(train) + geom_point(aes(Item_Weight, Item_Outlet_Sales), colour = "violet", alpha = 0.3) +theme(axis.title = element_text(size = 8.5)) > p9
```



MISSING VALUE TREATEMENT:

There are different methods to treat missing values based on the problem and the data. Some of the common techniques are as follows:

1. **Deletion of rows**: In train dataset, observations having missing values in any variable are deleted.
2. The downside of this method is the loss of information and drop in prediction power of model.
3. **Mean/Median/Mode Imputation**: In case of continuous variable, missing values can be replaced with
4. mean or median of all known values of that variable. For categorical variables, we can use mode of the given values to replace the missing values.

5.**Building Prediction Model**: We can even make a predictive model to impute missing data in a variable.

Here we will treat the variable having missing data as the target variable and the other variables as predictors.

We will divide our data into 2 datasets—one without any missing value for that variable and the other with

missing values for that variable.

The former set would be used as training set to build the predictive model and it would then be applied to

the latter set to predict the missing values.

Treatment 1:we have missing values in Item_Weight and

Item_Outlet_Sales. Missing data in Item_Outlet_Sales can be ignored since they belong to the test dataset. We'll now impute *Item_Weight* with mean weight based on the *Item_Identifier* variable.

```
> sum(is.na(combi$Item_Weight))
[1] 2439
> missing_index=which(is.na(combi$Item_Weight))
> for(i in missing_index){
+   item=combi$Item_Identifier[i]
+   combi$Item_Weight[i]=mean(combi)
+ }
```

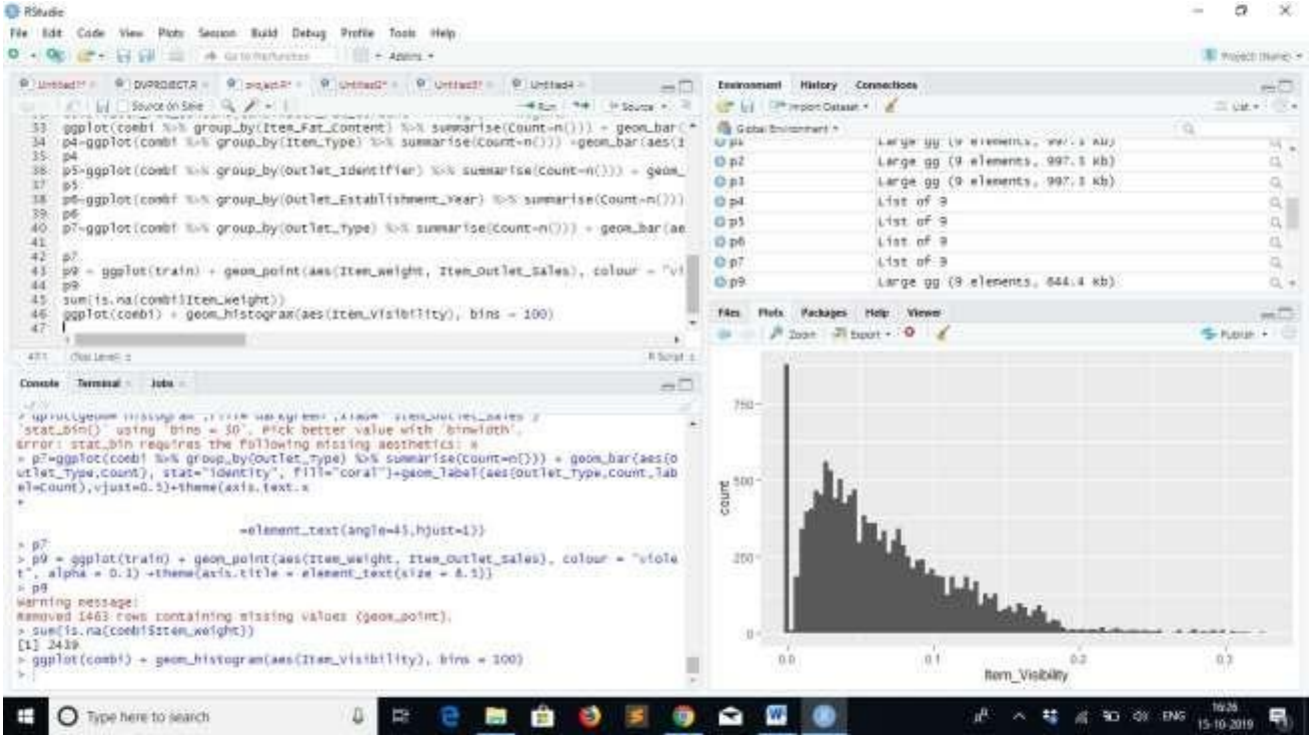
Treatment2:

Replacing 0's in Item_Visibility variable

Similarly, zeroes in Item_Visibility variable can be replaced with *Item_Identifier* wise mean values of *Item_Visibility*. It can be visualized in the plot below.

Before: Since item visibility can not be zero so we replace with the mean of the Item_Identifier.

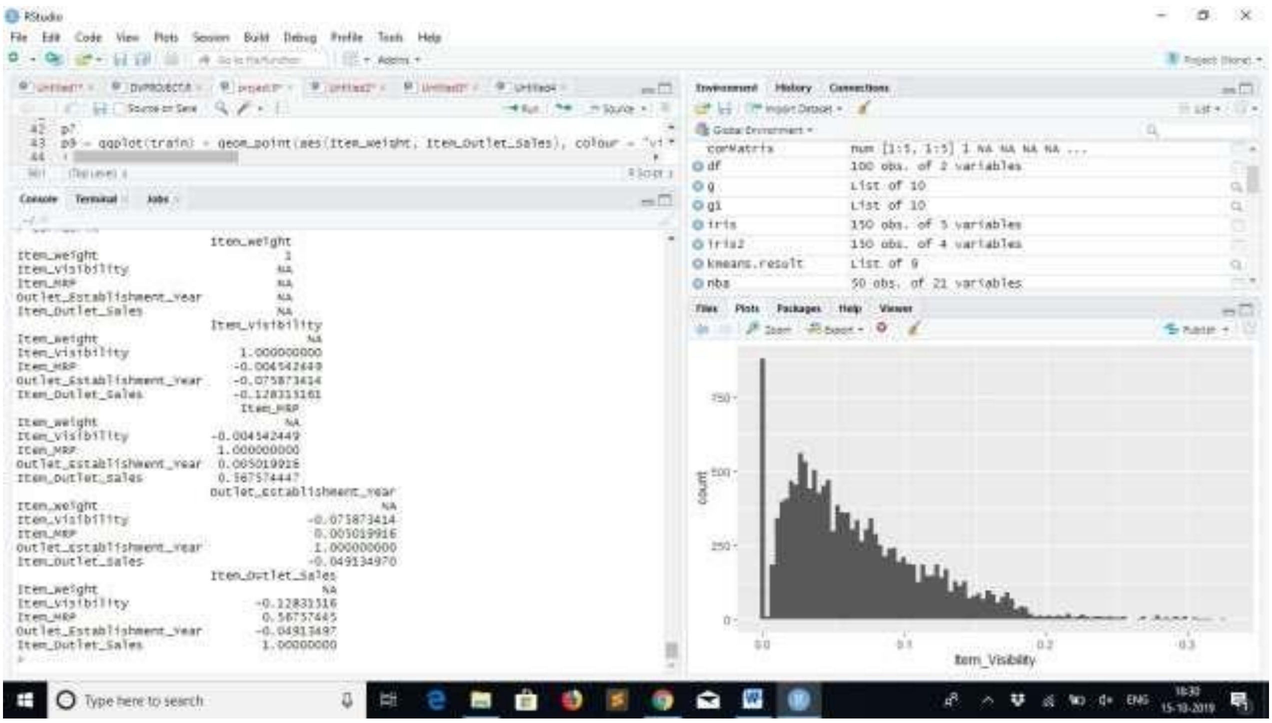
```
> ggplot(combi) + geom_histogram(aes(Item_Visibility), bins = 100)
```

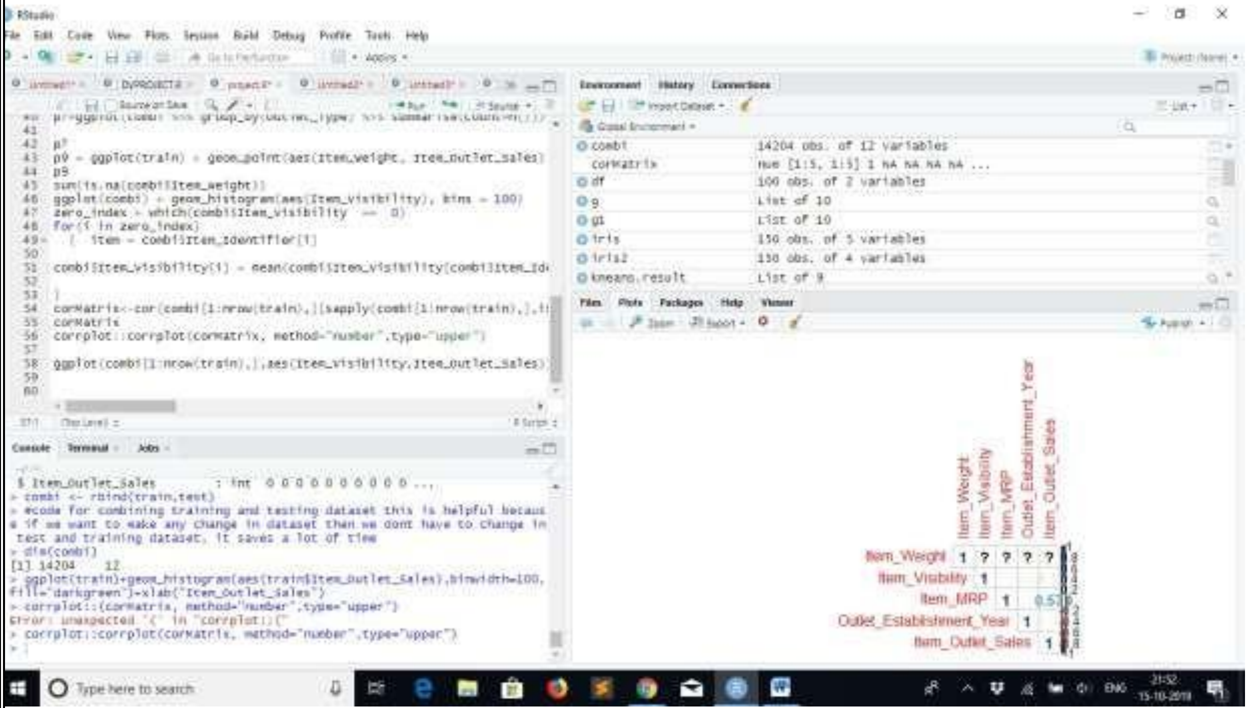


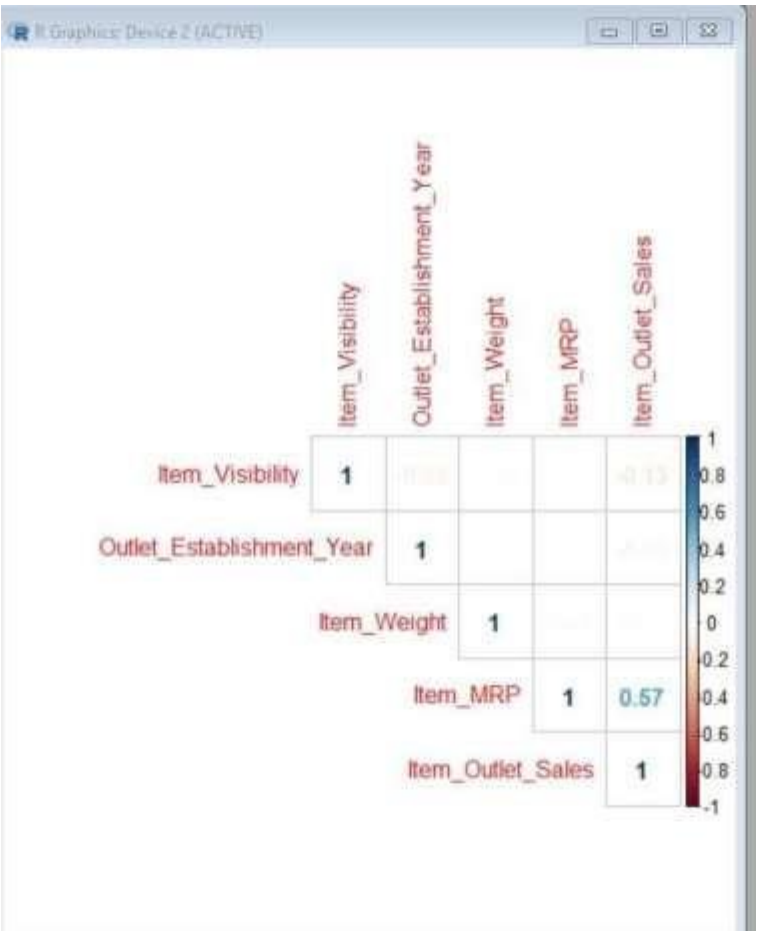
```
zero_index = which(combi$Item_Visibility == 0)

for(i in zero_index){ item = combi$Item_Identifier[i]

  combi$Item_Visibility[i] = mean(combi$Item_Visibility[combi$Item_Identifier == item],
na.rm = T)

 }
```

```
> corMatrix<-cor(combi[1:nrow(train),][sapply(combi[1:nrow(train),],is.numeric)])
> corMatrix
                          Item_Weight Item_Visibility     Item_MRP Outlet_Establishment_Year Item_Outlet_Sales
Item_Weight                         1              NA           NA                        NA                NA
Item_Visibility                    NA     1.000000000 -0.001314848              -0.074833504       -0.12862461
Item_MRP                           NA    -0.001314848  1.000000000               0.005019916        0.56757445
Outlet_Establishment_Year          NA    -0.074833504  0.005019916               1.000000000       -0.04913497
Item_Outlet_Sales                  NA    -0.128624612  0.567574447              -0.049134970        1.00000000
> corrplot::corrplot(corMatrix, method="number",type="upper")
> |
```

```
corrplot::corrplot(corMatrix, method="number",type="upper")
corrplot::corrplot(corMatrix, method="number",type="upper",order="hclust")
corrplot::corrplot(corMatrix, method="number",type="upper")
corrplot::corrplot(corMatrix, method="number",type="upper",order = "hclust")
```



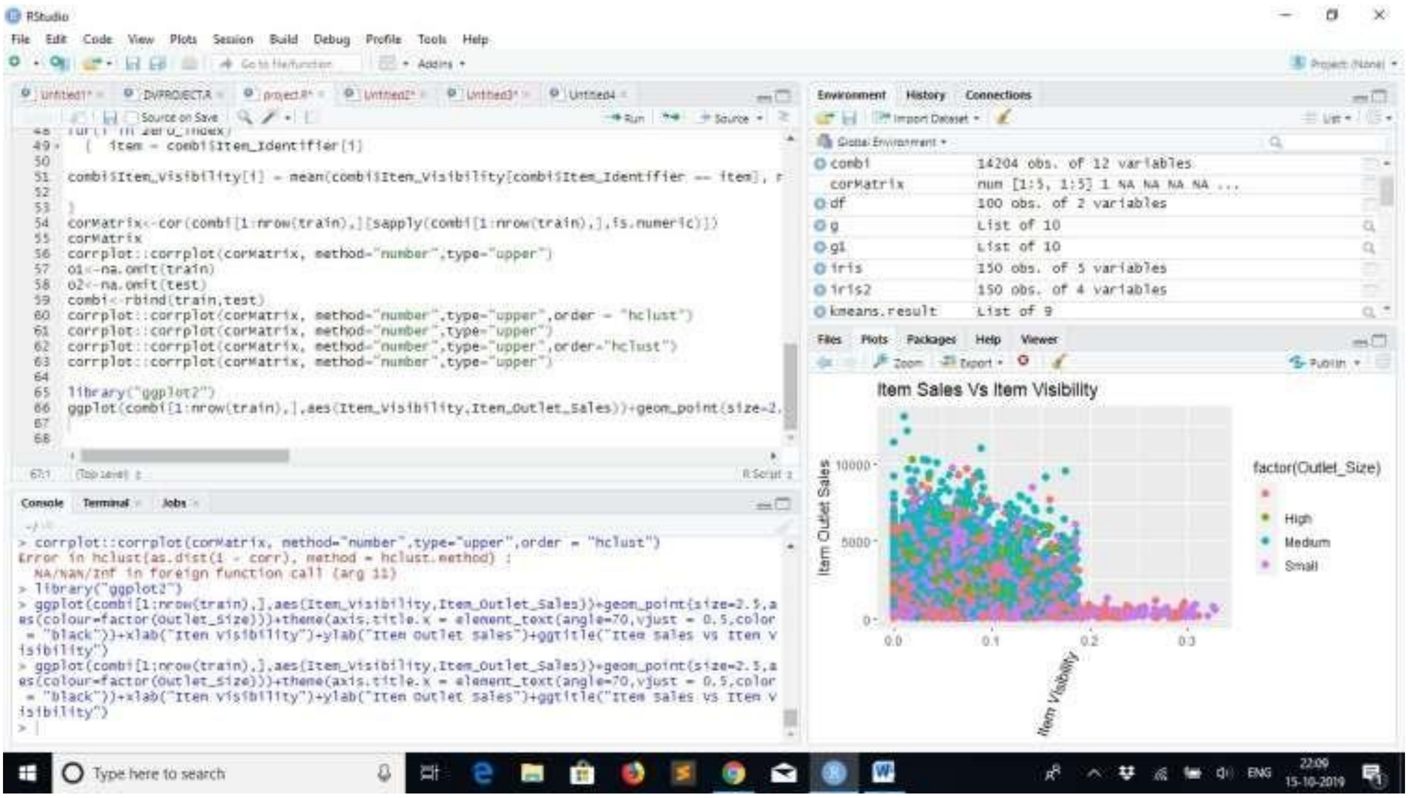

RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Environment History Connections
combi 14204 obs. of 12 variables
df 100 obs. of 2 variables
iris 150 obs. of 5 variables
iris2 150 obs. of 4 variables
Files Plots Packages Help Viewer
> corrplot::corrplot(cormatrix, method="number",type="upper")
Item_Weight
Item_Visibility
Item_MRP
Outlet_Establishment_Year
Item_Outlet_Sales
Type here to search


R Graphics Device 2 (ACTIVE)
Item_Visibility
Outlet_Establishment_Year
Item_Weight
Item_MRP
Item_Outlet_Sales
1
0.57
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1

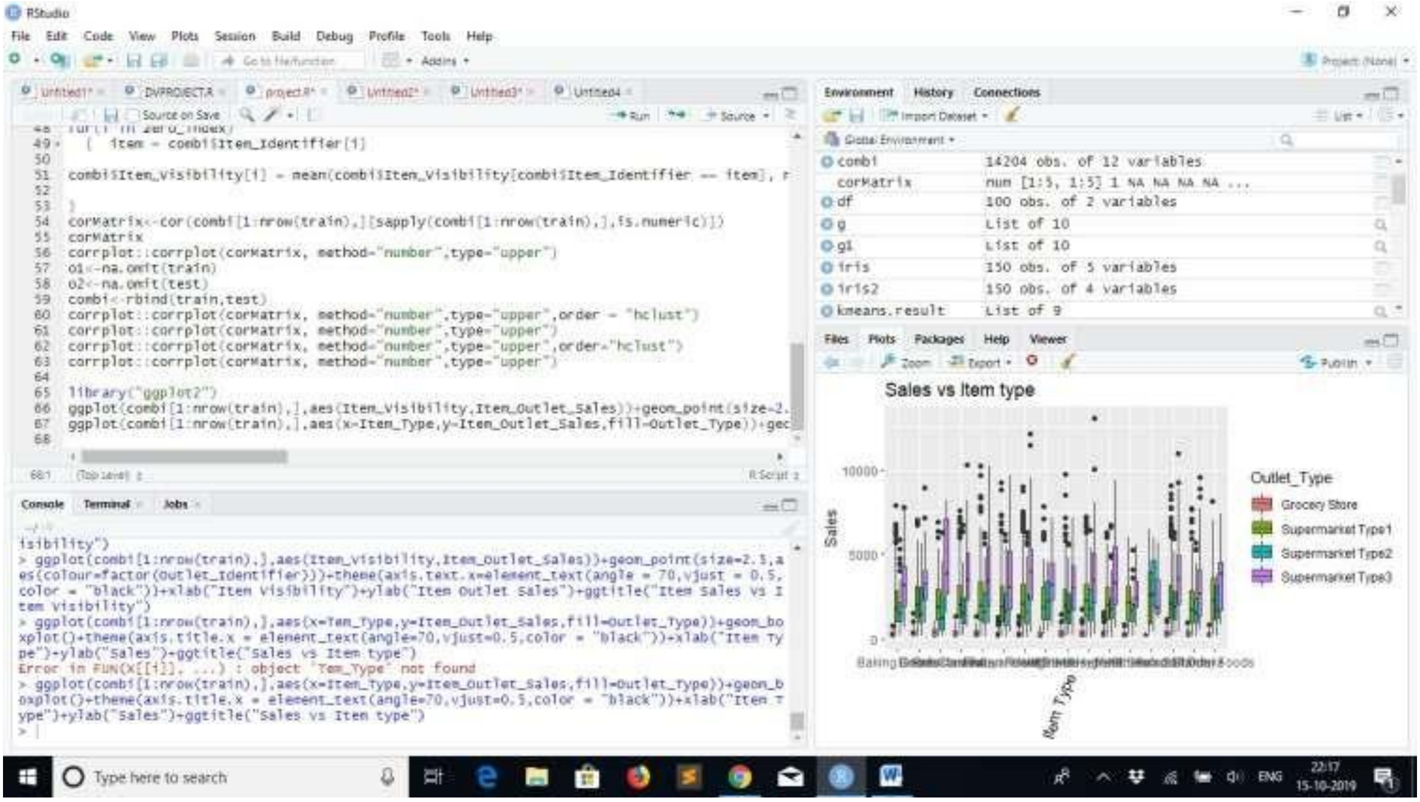
Conclusion:

Item_outlet_Sales has a strong positive correlation with Item_MRP and a somewhat weaker negative with item_Visibility.

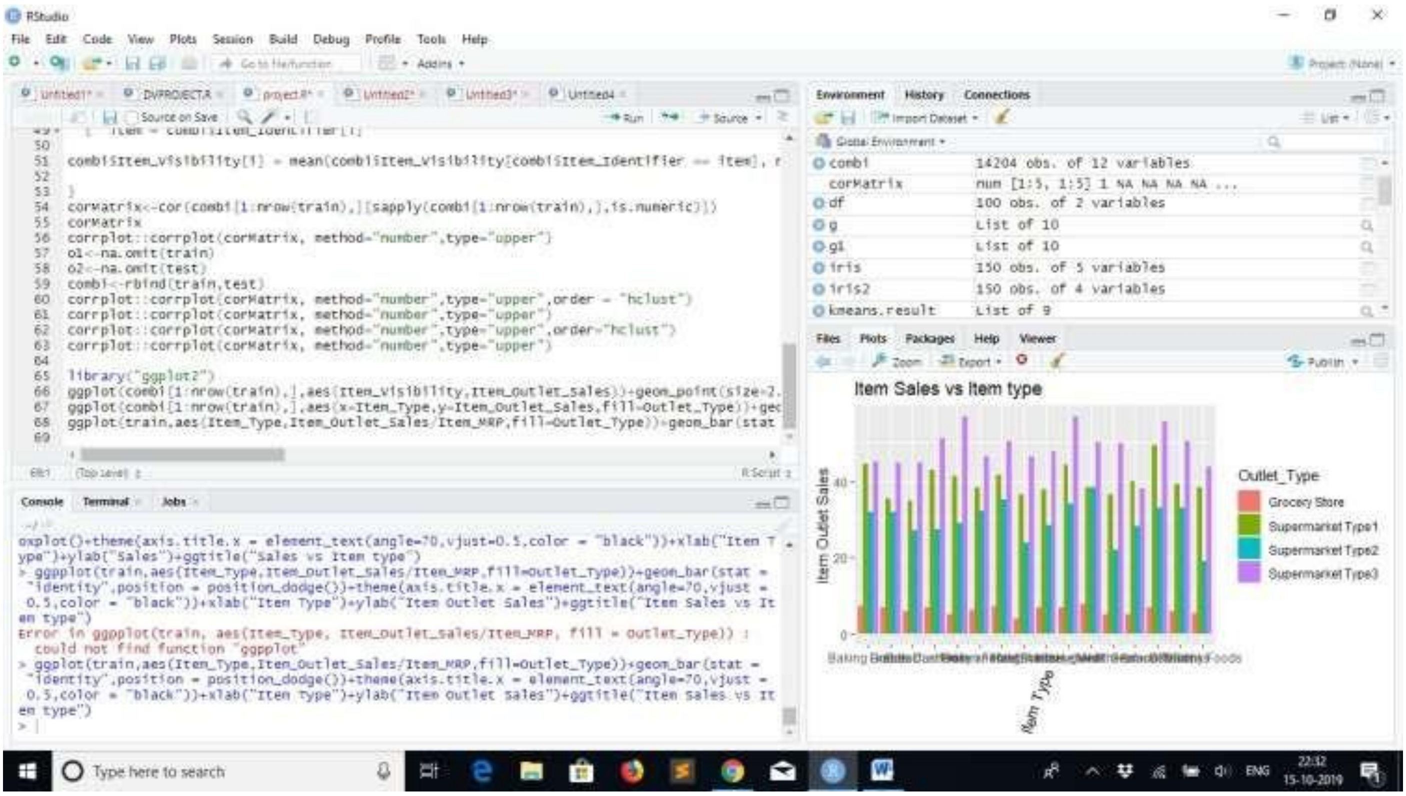
```
library("ggplot2")
ggplot(combi[1:nrow(train),],aes(Item_Visibility,Item_Outlet_Sales))+geom_point(size=2.5,
aes(colour=factor(Outlet_Identifier)))+theme(axis.text.x=element_text(angle = 70,vjust = 0.5,color = "black"))+
xlab("Item Visibility")+ylab("Item Outlet Sales")+ggtitle("Item Sales Vs Item Visibility")
```



ggplot(combi[1:nrow(train),],aes(x=Item_Type,y=Item_Outlet_Sales,fill=Outlet_Type))+geom_boxplot()+theme(axis.title.x = element_text(angle=70,vjust=0.5,color = "black"))+xlab("Item Type")+ylab("Sales")+ggtitle("Sales vs Item type")

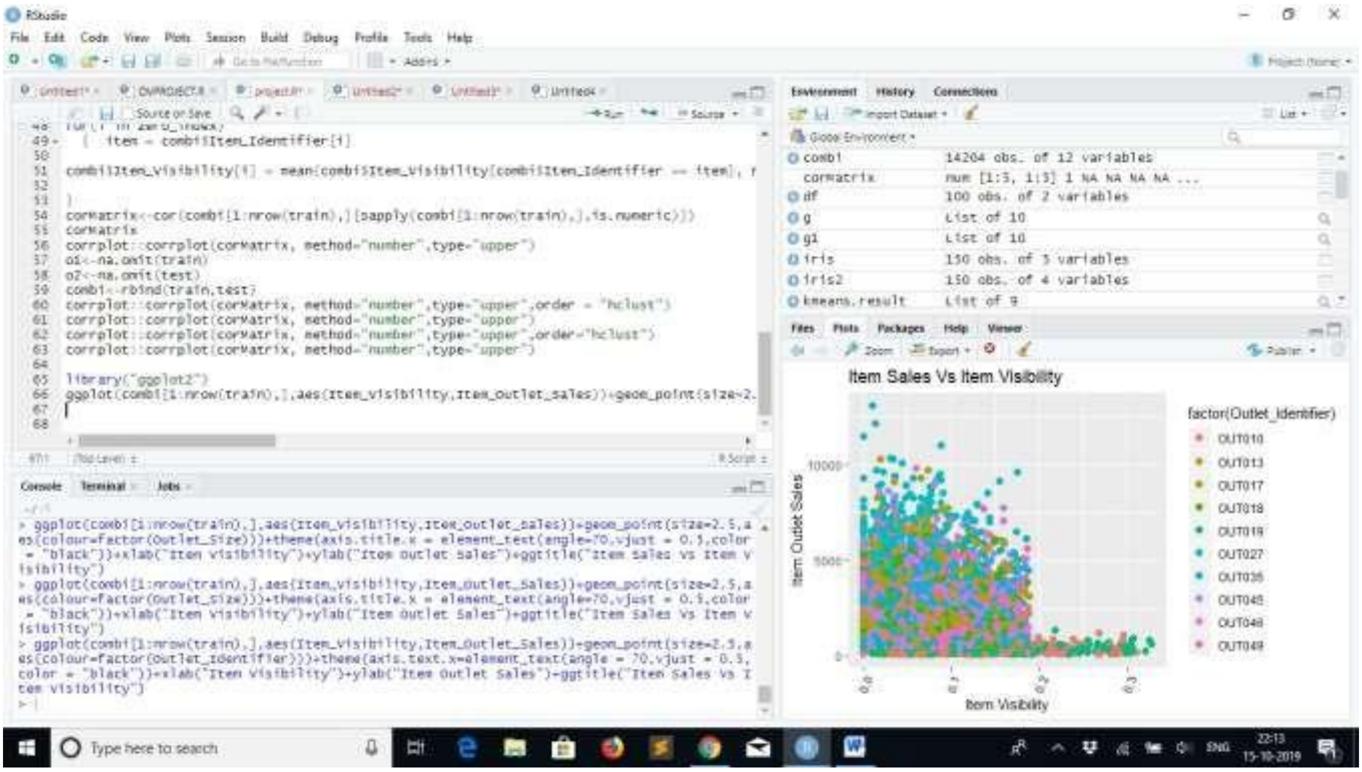


ggplot(train,aes(Item_Type,Item_Outlet_Sales/Item_MRP,fill=Outlet_Type))+geom_bar(stat = "identity",position = position_dodge())+theme(axis.title.x = element_text(angle=70,vjust = 0.5,color = "black"))+xlab("Item Type")+ylab("Item Outlet Sales")+ggtitle("Item Sales vs Item type")



Observation: only very few outlet which are medium in size have item Visibility low and high outlet sales.

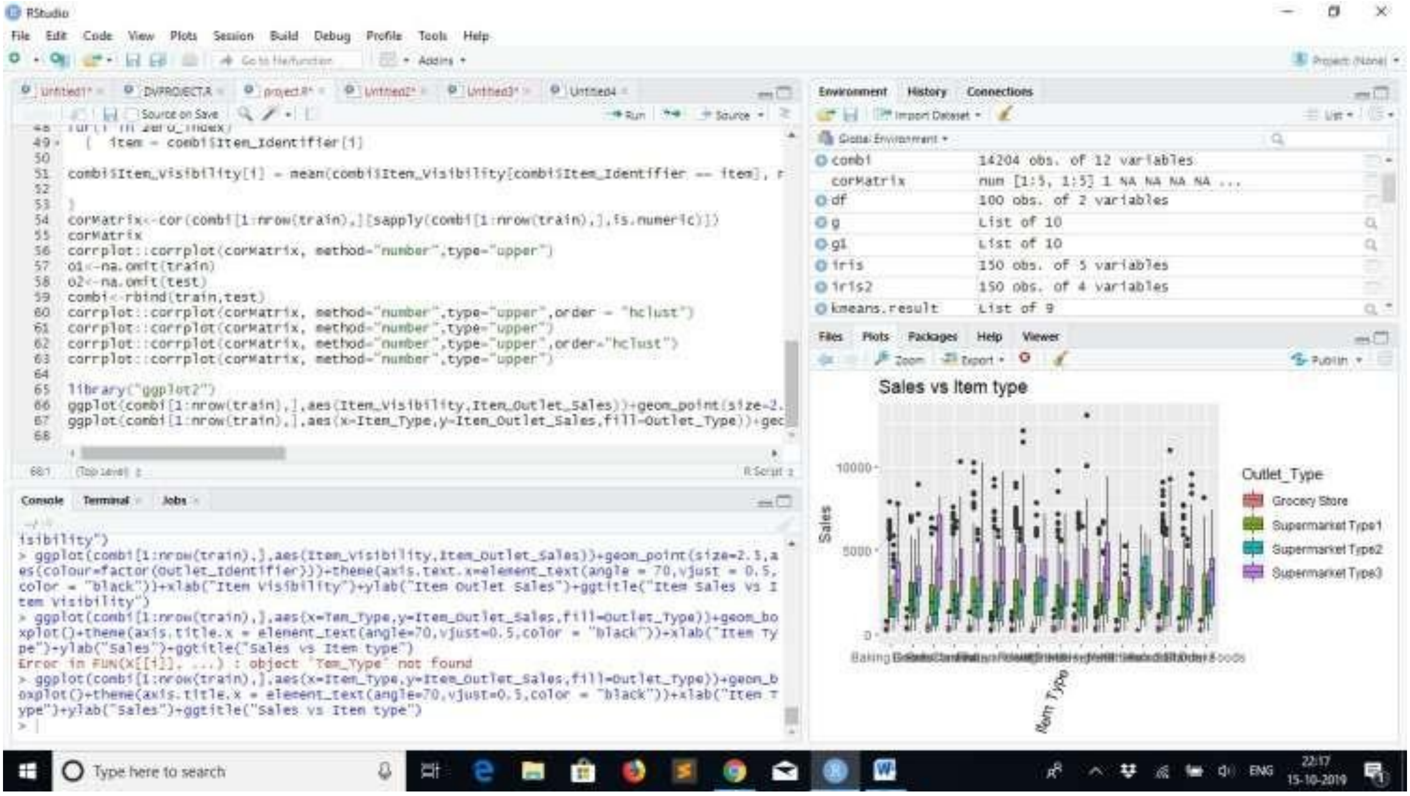
```
ggplot(combi[1:nrow(train),],aes(Item_Visibility,Item_Outlet_Sales))+geom_point(size=2.5,aes(colour=factor(Outlet_Identifier)))+
theme(a xis.text.x=element_text(angle = 70,vjust = 0.5,color = "black"))+xlab("Item Visibility")+ylab("Item Outlet Sales")+
ggtitle("Item Sales Vs Item Visibility")
```



Observations: outlet 27 and 35 are belongs to the outlet which have low item visibility but have more outlet sales.

BAR PLOT:

ggplot(combi[1:nrow(train),],aes(x=Item_Type,y=Item_Outlet_Sales,fill=Outlet_Type))+geom_boxplot()+ theme(axis.text.x=el ement_text(angle = 70,vjust = 0.5,color = "black"))+xlab("Item type")+ylab(" Sales")+ ggtitle("Sales Vs Item Visibility")



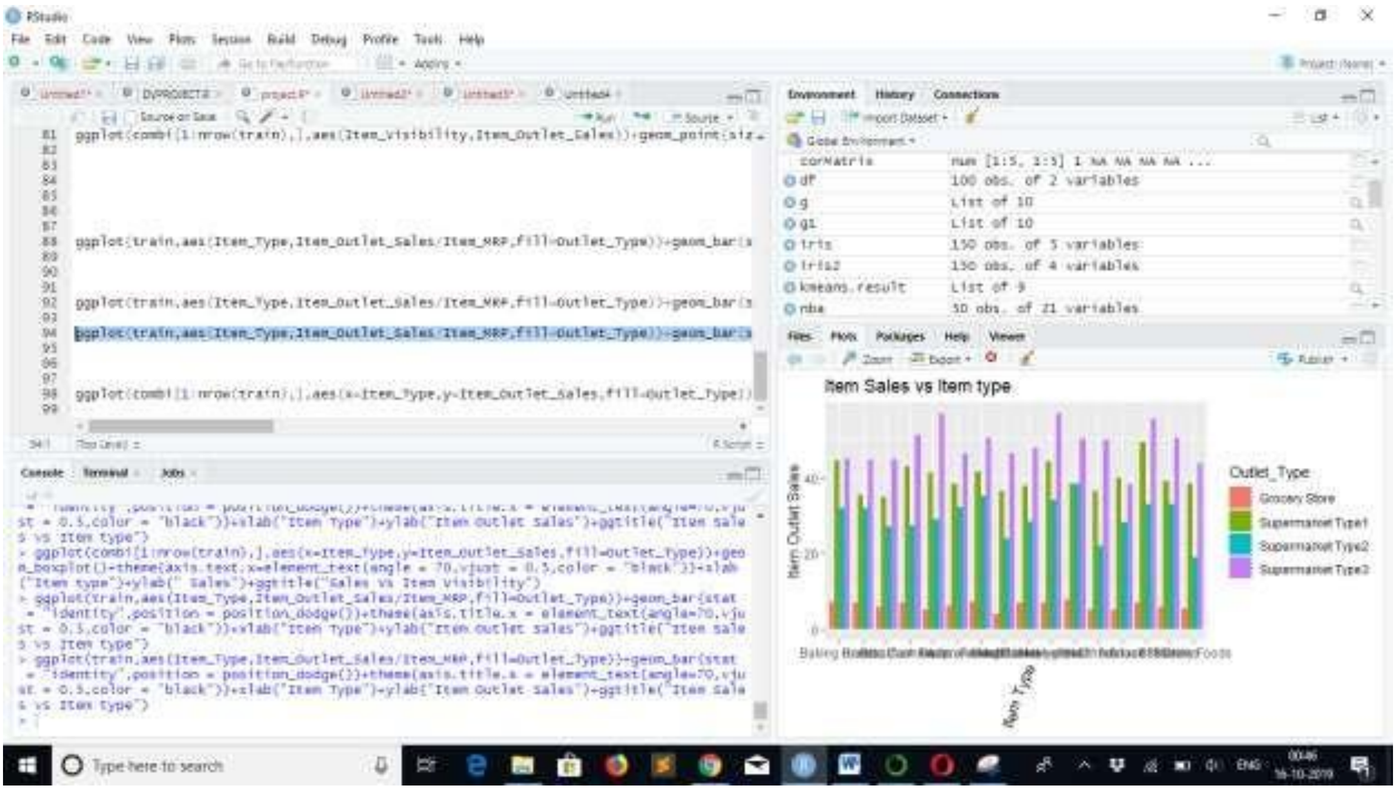
Conclusion: In most of the cases the supermarket type3 has most amount of sales irrespective of item_type.

BOX PLOT:

ggplot(train,aes(Item_Type,Item_Outlet_Sales/Item_MRP,fill=Outlet_Type))+

geom_bar(stat = "identity",position = position_dodge())+

theme(axis.title.x = element_text(angle=70,vjust = 0.5,color = "black"))+xlab("Item Type")+
ylab("Item Outlet Sales")+ggtitle("Item Sales vs Item type")



Python code:

```
import pandas as pd #import
numpy as np

train=pd.read_csv("train.csv",na_values={"Item_Visibility":[0]})

test=pd.read_csv("test.csv",na_values={"Item_Visibility":[0]})



train['source']='train'

test['source']='test'

data=pd.concat([train,test],ignore_index=True)

#the one thing we have to focus is item_outlet_Sales



discpt=data.describe()



#Lets find out how many zero'es values are

nan_descript=data.apply(lambda x: sum(x.isnull()))

#Now lets find out the unique values in each of the catogorical columns

uniq=data.apply(lambda x: len(x.unique()))

#let do grouping in each catogorical columns
```

```
col=["Item_Fat_Content","Item_Type","Outlet_Location_Type","Outlet_Size"]

for i in col:
   print("The frequency distribution of each catogorical columns is--" + i+"\n")
print(data[i].value_counts())

#Replacing the minimum nan values in the Item_Weight with its mean value

data.fillna({"Item_Weight":data["Item_Weight"].mean()},inplace=True)

#checking the current status of nan values in the dataframe
 nan_descript=data.apply(lambda x: sum(x.isnull()))
#Now we have 0 nan valuesin Item_Weight


data["Outlet_Size"].fillna(method="ffill",inplace=True) nan_descript=data.apply(lambda x:
sum(x.isnull()))


#Now working on the item_visibility


visibilty_avg=data.pivot_table(values="Item_Visibility",index="Item_Identifier")


itm_visi=data.groupby('Item_Type')

data_frames=[] for item,item_df in itm_visi:
data_frames.append(itm_visi.get_group(item))
for i in data_frames:
   i["Item_Visibility"].fillna(value=i["Item_Visibility"].mean(),inplace=True)
i["Item_Outlet_Sales"].fillna(value=i["Item_Outlet_Sales"].mean(),inplace=True)
```

```
new_data=pd.concat(data_frames)

nan_descript=new_data.apply(lambda x: sum(x.isnull()))

#Now we have successfully cleaned our complete dataset.
new_data["Item_Fat_Content"].replace({'LF':'Low Fat','reg':'Regular','low fat':'Low Fat'},inplace=True)

new_data["Item_Fat_Content"].value_counts()

#Implementing one-hot-Coding method for getting the categorical variables
from sklearn.preprocessing import LabelEncoder le = LabelEncoder()
data=new_data
data['Outlet'] = le.fit_transform(data['Outlet_Identifier'])
var_mod = ['Item_Fat_Content','Outlet_Location_Type','Outlet_Size','Item_Type','Outlet_Type']
le = LabelEncoder() for i in var_mod:
    data[i] = le.fit_transform(data[i]) #One
Hot Coding:
data = pd.get_dummies(data, columns=['Item_Fat_Content','Outlet_Location_Type','Outlet_Size','Outlet_Type',
                          'Item_Type'])

#Exporting the datas

train = data.loc[data['source']=="train"]

test = data.loc[data['source']=="test"]

#Drop unnecessary columns:
test.drop(['Item_Outlet_Sales','source'],axis=1,inplace=True)
```

#here we are droping the "Item_Outlet_Sales because this only we want to be predicted from the model that we are going to built train.drop(['source'],axis=1,inplace=True)

#Export files as modified versions:
train.to_csv("train_modified.csv",index=False)
test.to_csv("test_modified.csv",index=False)

#Let's start building the baseline model as it is non -predicting model and also commenly known as informed guess

#Mean based:
mean_sales = train['Item_Outlet_Sales'].mean()

#Define a dataframe with IDs for submission: base1
= test[['Item_Identifier','Outlet_Identifier']]
base1['Item_Outlet_Sales'] = mean_sales

#Export submission file
base1.to_csv("alg0.csv",index=False)

'''Very Important Note for creating baseline model making
baseline models helps in setting a benchmark.
If your predictive algorithm is below this, there is something going seriously wrong and we should check your data.'''

#Define target and ID columns: target
= 'Item_Outlet_Sales'
IDcol = ['Item_Identifier','Outlet_Identifier']

''' Now from this I have to learn machine learning data_Analytics''' import
numpy as np

```
from sklearn import cross_validation, metrics def modelfit(alg,
dtrain, dtest, predictors, target, IDcol, filename):
    #Fit the algorithm on the data
    alg.fit(dtrain[predictors], dtrain[target])

    #Predict training set:
    dtrain_predictions = alg.predict(dtrain[predictors])

    #Perform cross-validation: cv_score = cross_validation.cross_val_score(alg,
dtrain[predictors], dtrain[target], cv=20, scoring='mean_squared_error') cv_score =
np.sqrt(np.abs(cv_score))

    #Print model report:
    print ("\nModel Report")
    print ("RMSE : %.4g" % np.sqrt(metrics.mean_squared_error(dtrain[target].values,
dtrain_predictions)))
    print ("CV Score : Mean - %.4g | Std - %.4g | Min - %.4g | Max - %.4g" %
(np.mean(cv_score),np.std(cv_score),np.min(cv_score),np.max(cv_score)))

    #Predict on testing data: dtest[target] =
alg.predict(dtest[predictors])

    #Export submission file:
IDcol.append(target)
    submission = pd.DataFrame({ x: dtest[x] for x in IDcol})
submission.to_csv(filename, index=False)



#Liner Regression model
print("Creating the models and processing") from
sklearn.linear_model import LinearRegression, Ridge
predictors = [x for x in train.columns if x not in [target]+IDcol]
# print predictors
```

```
alg1 = LinearRegression(normalize=True)
modelfit(alg1, train, test, predictors, target, IDcol, 'alg1.csv') coef1
= pd.Series(alg1.coef_, predictors).sort_values()
coef1.plot(kind='bar', title='Model Coefficients')

#Ridge Regression Model
predictors = [x for x in train.columns if x not in [target]+IDcol] alg2
= Ridge(alpha=0.05,normalize=True)
modelfit(alg2, train, test, predictors, target, IDcol, 'alg2.csv') coef2
= pd.Series(alg2.coef_, predictors).sort_values()
coef2.plot(kind='bar', title='Model Coefficients')
print("Model has been successfully created and trained. The predicted result is in alg2.csv")

# Decision Tree Model

from sklearn.tree import DecisionTreeRegressor
predictors = [x for x in train.columns if x not in [target]+IDcol]
alg3 = DecisionTreeRegressor(max_depth=15, min_samples_leaf=100) modelfit(alg3,
train, test, predictors, target, IDcol, 'alg3.csv')
coef3 = pd.Series(alg3.feature_importances_, predictors).sort_values(ascending=False)
coef3.plot(kind='bar', title='Feature Importances')

print("Model has been successfully created and trained. The predicted result is in alg3.csv")
```

RESULT:

Liner Regression model:

Item_Identifier	Outlet_Identifier	Item_Outlet_Sales
FDL48	OUT018	582
FDC48	OUT027	2742
FDA36	OUT017	3086
FDM24	OUT049	2498
FDD48	OUT010	-74
FDW12	OUT035	2406
FDC37	OUT027	3162
FDZ36	OUT045	3006
FDK60	OUT017	1722
FDG12	OUT046	2062
FDX24	OUT013	1490
FDS60	OUT027	4266
FDC60	OUT049	1482
FDA48	OUT018	3174
FDZ36	OUT027	4434
FDW23	OUT018	350
FDW60	OUT018	2594
FDY59	OUT010	-382
FDR60	OUT049	1250
FDV24	OUT013	2402
FDZ48	OUT010	-146
FDS24	OUT027	2894
FDS48	OUT045	2394
FDM60	OUT010	-1186

Ridge Regression Model:

Item_Identifier	Outlet_Identifier	Item_Outlet_Sales
FDL48	OUT018	657.8721748
FDC48	OUT027	2756.411826
FDA36	OUT017	3012.743399
FDM24	OUT049	2496.634157
FDD48	OUT010	68.73456657
FDW12	OUT035	2409.049018
FDC37	OUT027	3151.715131
FDZ36	OUT045	3052.338255
FDK60	OUT017	1688.679221
FDG12	OUT046	2045.118054
FDX24	OUT013	1576.155234
FDS60	OUT027	4197.250969
FDC60	OUT049	1530.734423
FDA48	OUT018	3124.994325
FDZ36	OUT027	4335.83567
FDW23	OUT018	452.1847847
FDW60	OUT018	2578.304215
FDY59	OUT010	-268.6900384
FDR60	OUT049	1321.291561
FDV24	OUT013	2419.483633
FDZ48	OUT010	-14.0282151
FDS24	OUT027	2875.715392
FDS48	OUT045	2484.270288
FDM60	OUT010	-1007.011644

Decision Tree Model:

	A	B	C	D
1	Item_Identifier	Outlet_Identifier	Item_Outlet_Sales	
2	FDL48	OUT018	711.0628209	
3	FDC48	OUT027	2517.882088	
4	FDA36	OUT017	2778.989278	
5	FDM24	OUT049	2490.526887	
6	FDD48	OUT010	281.4820818	
7	FDW12	OUT035	2490.526887	
8	FDC37	OUT027	2517.882088	
9	FDZ36	OUT045	3387.204723	
10	FDK60	OUT017	1559.342975	
11	FDG12	OUT046	2282.978881	
12	FDX24	OUT013	1498.239409	
13	FDS60	OUT027	5329.314724	
14	FDC60	OUT049	1335.178675	
15	FDA48	OUT018	3327.952583	
16	FDZ36	OUT027	5329.314724	
17	FDW23	OUT018	507.0464037	
18	FDW60	OUT018	2888.49101	
19	FDY59	OUT010	219.2185209	
20	FDR60	OUT049	1335.178675	
21	FDV24	OUT013	2490.526887	
22	FDZ48	OUT010	281.4820818	
23	FDS24	OUT027	2517.882088	
24	FDS48	OUT045	2660.950403	
25	FDM60	OUT010	92.67548155	