BigMart Sales Visualization & Prediction

PROJECT REPORT

Data Visualization (CSE3020)

Submitted By:-

Mitarth Jain 17BCE0765 (L5+L6 **)**

Sandesh Pokharkar 17BCE0768 (L43+L44**)**

Submitted to:-

Prof. Annapurna Jonnalagadda

Slot- L5+L6

Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

November,2019

ACKNOWLEDGEMENTS

A deepest gratitude and sincere thanks to Prof. Annapurna Jonnalagadda in helping us complete our Project with several learning outcomes. We feel deeply obliged to thank the SCOPE (School of Computer Science and Engineering) Department and the VIT University for their services rendered and for giving us an opportunity to carry out our studies at the University.

Mitarth Jain (17BCE0765)

Sandesh Pokharkar (17BCE0768)

PROBLEM STATEMENT

The data scientists at BigMart have collected sales data for 1559 products across 10 stores in different cities. Also,certain attributes of each product and store have been defined. The aim is to build a predictive model and find out the sales of each product at a particular store.So the idea is to find out the properties of a product, and store which impacts the sales of a product.

INTRODUCTION

With the rapid development of global malls and stores chains and the increase in the number of electronic payment customers, the competition among the rival organizations is becoming more serious day by day. Each organization is trying to attract more customers using personalized and short-time offers which makes the prediction of future volume of sales of every item an important asset in the planning and inventory management of every organization, transport service, etc. Due to the cheap availability of computing and storage, it has become possible to use sophisticated machine learning algorithms for this purpose. In this project, we are providing forecast for the sales data of big mart in a number of big mart stores across various location types which is based on the historical data of sales volume. According to the characteristics of the data, we can use the method of multiple linear regression analysis and random forest to forecast the sales volume.

Motivation:-

Retail is another industry which extensively uses analytics to optimize business processes. Tasks like product placement, inventory management, customized offers, product bundling, etc. are being smartly handled using data science techniques. Sales prediction is a very common real-life problem that each company faces at least once in its lifetime. If done correctly, it can have a significant impact on the success and performance of that company.

Significance:

Due to the cheap availability of computing and storage, it has become possible to use sophisticated machine learning algorithms for this purpose. In this project, we are providing forecast for the sales data of big mart in a number of big mart stores across various location types which is based on the historical data of sales volume. According to the characteristics of the data, we can use the method of multiple linear regression analysis and random forest to forecast the sales volume. According to a study,companies with accurate sales predictions are 10% more likely to grow their revenue year-over-year and 7.3% more likely to hit quota.

Scope and Applications:

Using this model, we will analyze the properties of products and stores which play a key role in increasing sales. In order to understand the problem statement better , we can brainstorm possible factors that can impact the outcome.We will show our results and knowledge insight from data by the help of graphs and data visualization techniques to have a better and clear understanding about the analysis.

Data Set Information

Variable	Description
Item_Identifier	Unique product ID
Item_Weight	Weight of product
Item_Fat_Content	Whether the product is low fat or not
Item_Visibility	The % of total display area of all products in a store allocated to the particular product
Item_Type	The category to which the product belongs
Item_MRP	Maximum Retail Price (list price) of the product
Outlet_Identifier	Unique store ID
Outlet_Establishment_Year	The year in which store was established
Outlet_Size	The size of the store in terms of ground area covered
Outlet_Location_Type	The type of city in which the store is located
Outlet_Type	Whether the outlet is just a grocery store or some sort of supermarket
Item_Outlet_Sales	Sales of the product in the particulat store. This is the outcome variable to be predicted.

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	Item_Iden	Item_Wei	Item_Fat_	Item_Visib	Item_Type	Item_MRP	Outlet_Ide	Outlet_Est	Outlet_Siz	Outlet_Lo	Outlet_Ty	Item_Outlet_Sales	
2	FDA15	9.3	Low Fat	0.016047	Dairy	249.8092	OUT049	1999	Medium	Tier 1	Supermark	3735.138	
3	DRC01	5.92	Regular	0.019278	Soft Drink	48.2692	OUT018	2009	Medium	Tier 3	Supermark	443.4228	
4	FDN15	17.5	Low Fat	0.01676	Meat	141.618	OUT049	1999	Medium	Tier 1	Supermark	2097.27	
5	FDX07	19.2	Regular	0	Fruits and	182.095	OUT010	1998		Tier 3	Grocery St	732.38	
6	NCD19	8.93	Low Fat	0	Household	53.8614	OUT013	1987	High	Tier 3	Supermark	994.7052	
7	FDP36	10.395	Regular	0	Baking Go	51.4008	OUT018	2009	Medium	Tier 3	Supermark	556.6088	
8	FDO10	13.65	Regular	0.012741	Snack Foo	57.6588	OUT013	1987	High	Tier 3	Supermark	343.5528	
9	FDP10		Low Fat	0.12747	Snack Foo	107.7622	OUT027	1985	Medium	Tier 3	Supermark	4022.764	
10	FDH17	16.2	Regular	0.016687	Frozen Foo	96.9726	OUT045	2002		Tier 2	Supermark	1076.599	
11	FDU28	19.2	Regular	0.09445	Frozen Foo	187.8214	OUT017	2007		Tier 2	Supermark	4710.535	
12	FDY07	11.8	Low Fat	0	Fruits and	45.5402	OUT049	1999	Medium	Tier 1	Supermark	1516.027	
13	FDA03	18.5	Regular	0.045464	Dairy	144.1102	OUT046	1997	Small	Tier 1	Supermark	2187.153	
14	FDX32	15.1	Regular	0.100014	Fruits and	145.4786	OUT049	1999	Medium	Tier 1	Supermark	1589.265	
15	FDS46	17.6	Regular	0.047257	Snack Foo	119.6782	OUT046	1997	Small	Tier 1	Supermark	2145.208	
16	FDF32	16.35	Low Fat	0.068024	Fruits and	196.4426	OUT013	1987	High	Tier 3	Supermark	1977.426	
17	FDP49	9	Regular	0.069089	Breakfast	56.3614	OUT046	1997	Small	Tier 1	Supermark	1547.319	
18	NCB42	11.8	Low Fat	0.008596	Health and	115.3492	OUT018	2009	Medium	Tier 3	Supermark	1621.889	
19	FDP49	9	Regular	0.069196	Breakfast	54.3614	OUT049	1999	Medium	Tier 1	Supermark	718.3982	
20	DRI11		Low Fat	0.034238	Hard Drink	113.2834	OUT027	1985	Medium	Tier 3	Supermark	2303.668	
21	FDU02	13.35	Low Fat	0.102492	Dairy	230.5352	OUT035	2004	Small	Tier 2	Supermark	2748.422	
22	FDN22	18.85	Regular	0.13819	Snack Foo	250.8724	OUT013	1987	High	Tier 3	Supermark	3775.086	
23	FDW12		Regular	0.0354	Baking Go	144.5444	OUT027	1985	Medium	Tier 3	Supermark	4064.043	
24	NCB30	14.6	Low Fat	0.025698	Household	196.5084	OUT035	2004	Small	Tier 2	Supermark	1587.267	
25	FDC37		Low Fat	0.057557	Baking Go	107.6938	OUT019	1985	Small	Tier 1	Grocery St	214.3876	
26	FDR28	13.85	Regular	0.025896	Frozen Foo	165.021	OUT046	1997	Small	Tier 1	Supermark	4078.025	
27	NCD06	13	Low Fat	0.099887	Household	45.906	OUT017	2007		Tier 2	Supermark	838.908	
28	FDV10	7.645	Regular	0.066693	Snack Foo	42.3112	OUT035	2004	Small	Tier 2	Supermark	1065.28	
29	DRJ59	11.65	low fat	0.019356	Hard Drink	39.1164	OUT013	1987	High	Tier 3	Supermark	308.9312	

Literature Survey

1.) Tanu jain, AK Sharma [1] interprets that the algorithms which are frequently used in the field of association rule mining are Eclat and Apriori (market basket analysis) algorithms. Both of these algorithms are mainly used for mining of primarily data sets and to find fraternity(associations) between these regular data sets using R which is a domain based language for data exploration, analysis and analytics. Several packages and libraries of R has been used by the authors to examine the performance of Eclat and Apriori algorithms on different item sets on the basis of execution time taken by both of the algorithms.

2.) Author's of this paper [2] uses one of the most effective data processing and analyzation tool which is known as R Studio to analyze the RF(Random Forest) and LDA(latent dirichlet allocation) algorithms based on the outcome of large data sets to come up with more improved results which provides help to predicts some outcome in advance.

3.) The author's from Renmin university of china [3] explained about a system known as novel trigger system which would give better better prediction results as compared to single prediction model for various different types of items or products. After research the authors come at a point to conclude that the accuracy of novel trigger system is more than that of single prediction model. various enterprises can use this for better future prediction which would affect their sales.

4.) Author's [4] basically interprets that what is data analysis and how we can do it efficiently?.In this paper author recommends R for data analysis because of its tremendous capability of data exploration, several inbuilt packages, easy to

implement several machine learning algorithms etc. As we know that R is a statistical language as well as programming language which helps in effective model prediction and better visualization techniques. So after survey authors found that the with R data analysis is much more efficient.

Tools and language that will be used are:-

R studio and R programming language will be used.We will handle this problem in a structured way.We will be following the table of content given below.

1. Hypothesis Generation
2. Loading Packages and Data
3. Data Structure and Content
4. Exploratory Data Analysis
   (i) Univariate Analysis
   (ii) Bivariate Analysis
5. Missing Value Treatment
6. Pre-Processing Data
7. Modeling
   (i) Linear Regression
   (ii) Random Forest
8. Result

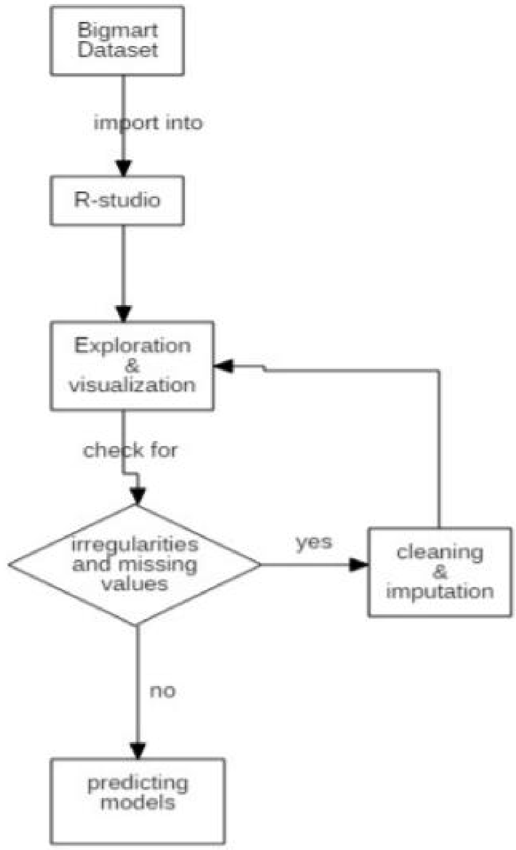


Diagram 1: Architecture diagram

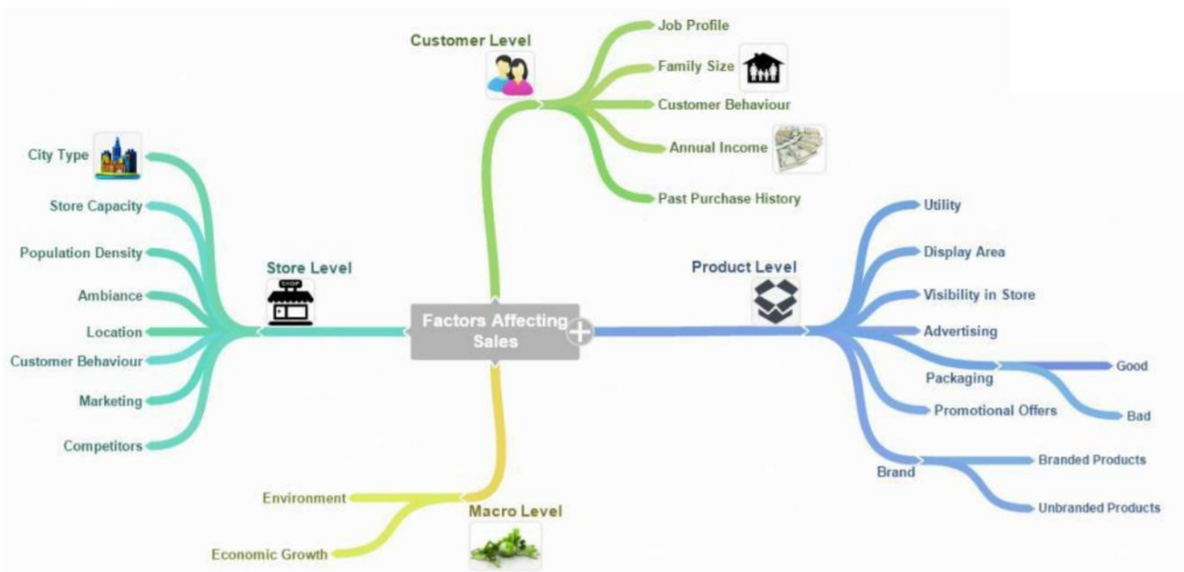
1. Hypothesis Generation

What is hypothesis generation?

This is a very important stage in any machine learning process. 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.

How to do hypothesis generation?

One very effective technique to generate hypotheses is by creating mind maps. You can draw it even using a pen and paper. The General Methodology is as follows: Write the main idea in the center. Draw branches from the center such they are connected with one another with final output shown towards the end.



2. 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

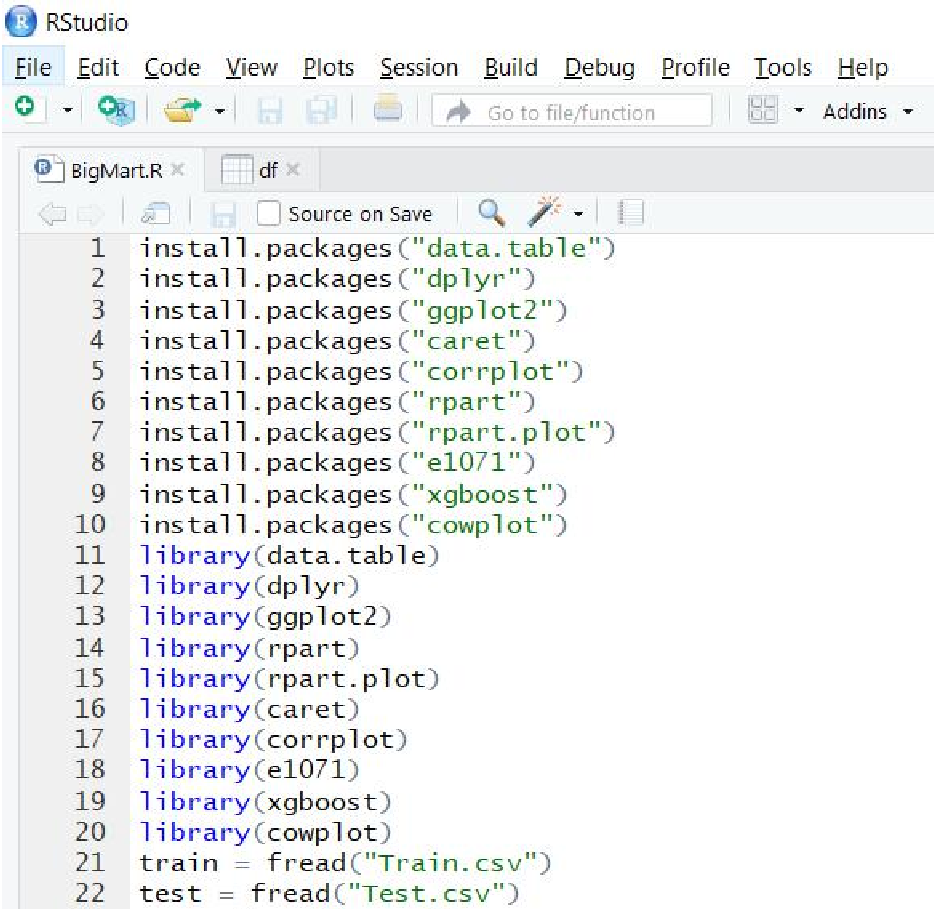
library(caret)- used for modeling

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(%>%)




```
install.packages("data.table")
install.packages("dplyr")
install.packages("ggplot2")
install.packages("caret")
install.packages("corrplot")
install.packages("rpart")
install.packages("rpart.plot")
install.packages("e1071")
install.packages("xgboost")
install.packages("cowplot")
library(data.table)
library(dplyr)
library(ggplot2)
library(rpart)
library(rpart.plot)
library(caret)
library(corrplot)
library(e1071)
library(xgboost)
library(cowplot)
train = fread("Train.csv")
test = fread("Test.csv")
```

```
> dim(train)
[1] 8523   12
> dim(test)
[1] 5681   12
> names(train)
 [1] "Item_Identifier"
 [2] "Item_Weight"
 [3] "Item_Fat_Content"
 [4] "Item_Visibility"
 [5] "Item_Type"
 [6] "Item_MRP"
 [7] "Outlet_Identifier"
 [8] "Outlet_Establishment_Year"
 [9] "Outlet_Size"
[10] "Outlet_Location_Type"
[11] "Outlet_Type"
[12] "Item_Outlet_Sales"
> names(test)
 [1] "Item_Identifier"
 [2] "Item_Weight"
 [3] "Item_Fat_Content"
 [4] "Item_Visibility"
 [5] "Item_Type"
 [6] "Item_MRP"
 [7] "Outlet_Identifier"
 [8] "Outlet_Establishment_Year"
 [9] "Outlet_Size"
[10] "Outlet_Location_Type"
[11] "Outlet_Type"
[12] "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 1185 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 ...
 $ Item_Outlet_Sales        : logi  NA NA NA NA NA NA ...
```

combi <- rbind(train, test)

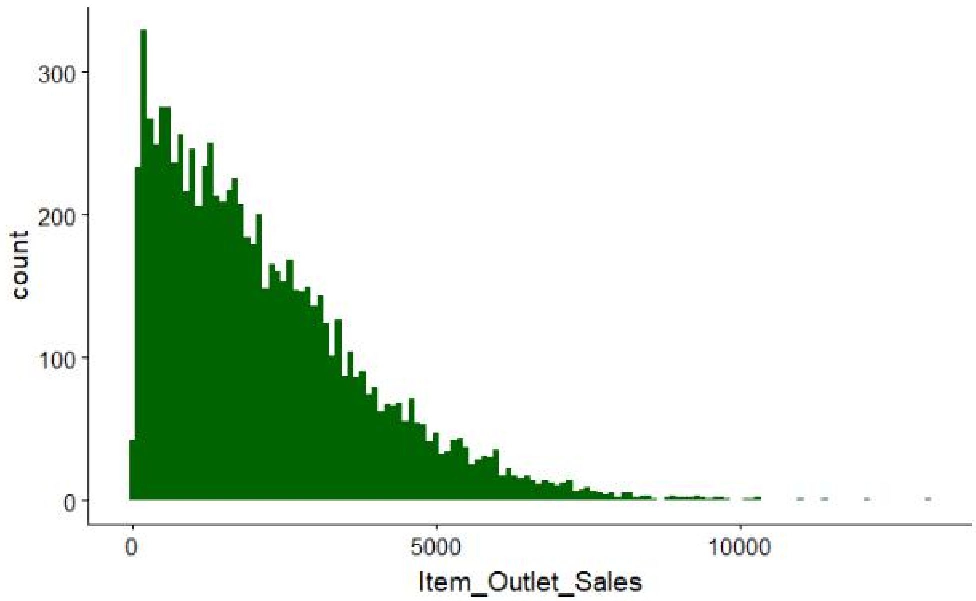
dim(combi)

```
> dim(combi)
[1] 14204    12
```

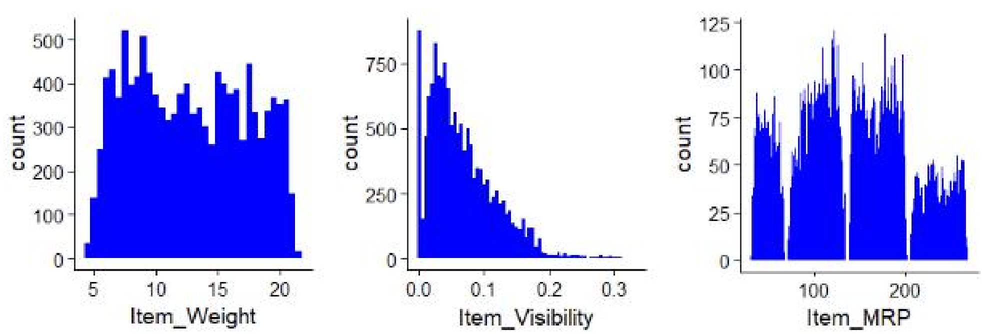
4.) Exploratory Data Visualization

(i) Univariate Analysis

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



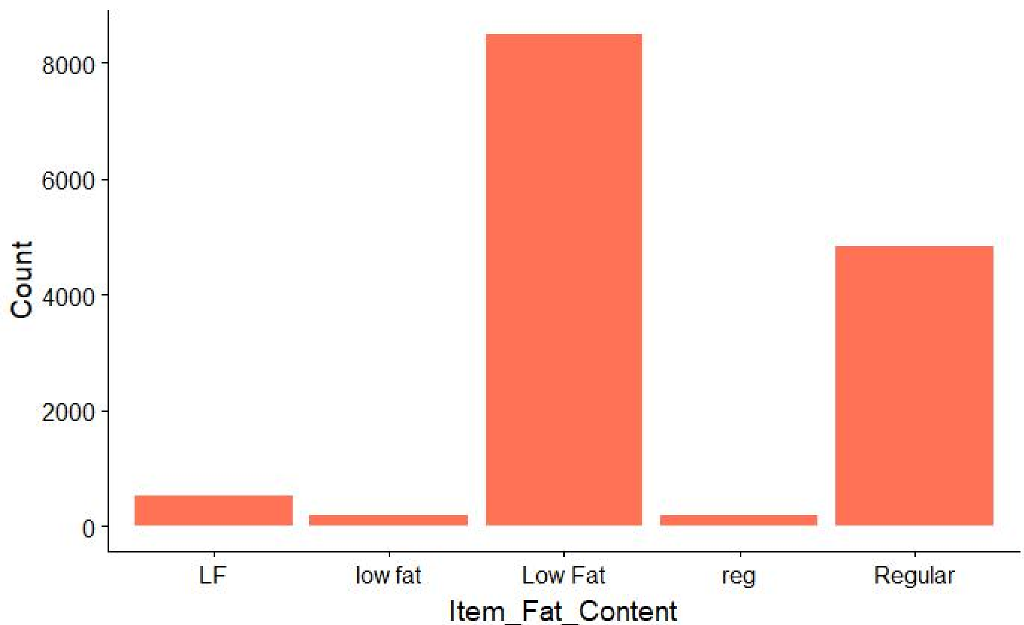

```
p1 = ggplot(combi) + geom_histogram(aes(Item_Weight), binwidth = 0.5, fill =
"blue")
p2 = ggplot(combi) + geom_histogram(aes(Item_Visibility), binwidth = 0.005,
fill = "blue")
p3 = ggplot(combi) + geom_histogram(aes(Item_MRP), binwidth = 1, fill =
"blue")
plot_grid(p1, p2, p3, nrow = 1)
```



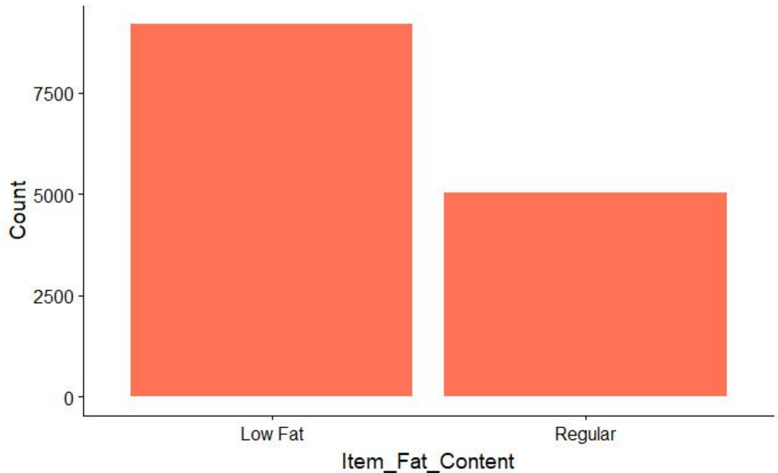
Observation:

- 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.

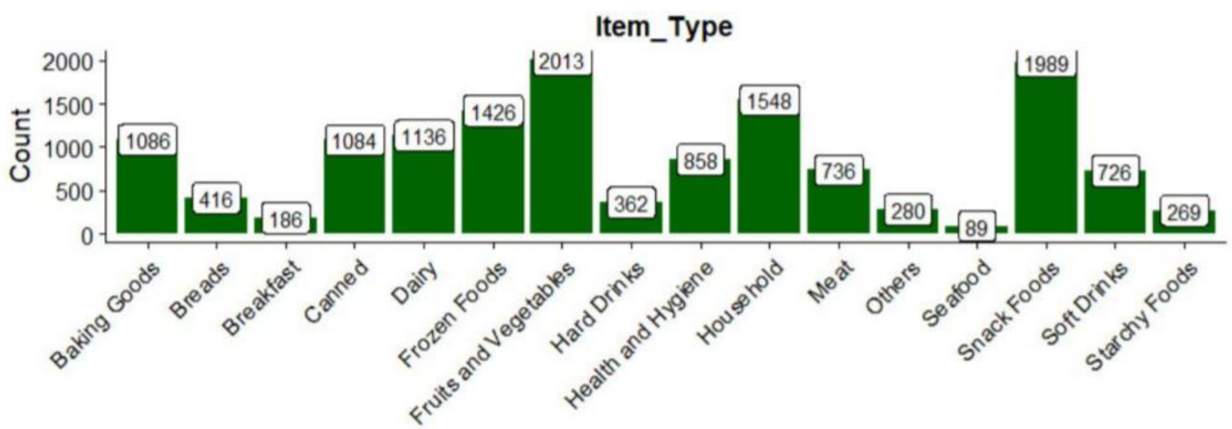
```
> ggplot(combi %>% group_by(Item_Fat_Content) %>% summarise(Count = n())) +
  geom_bar(aes(Item_Fat_Content, Count), stat = "identity", fill = "coral1")
```



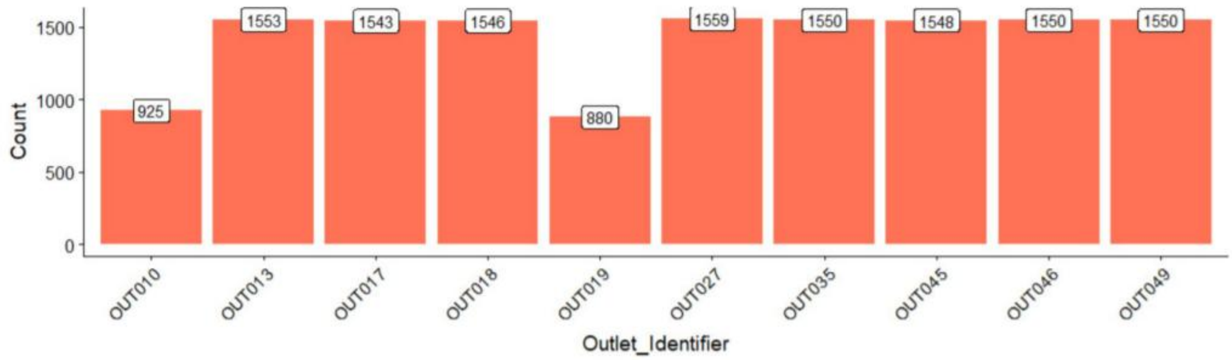

```
> combi$Item_Fat_Content[combi$Item_Fat_Content == "LF"] = "Low Fat"
> combi$Item_Fat_Content[combi$Item_Fat_Content == "low fat"] = "Low Fat"
> combi$Item_Fat_Content[combi$Item_Fat_Content == "reg"] = "Regular"
> ggplot(combi %>% group_by(Item_Fat_Content) %>% summarise(Count = n())) +
  geom_bar(aes(Item_Fat_Content, Count), stat = "identity", fill = "coral1")
```



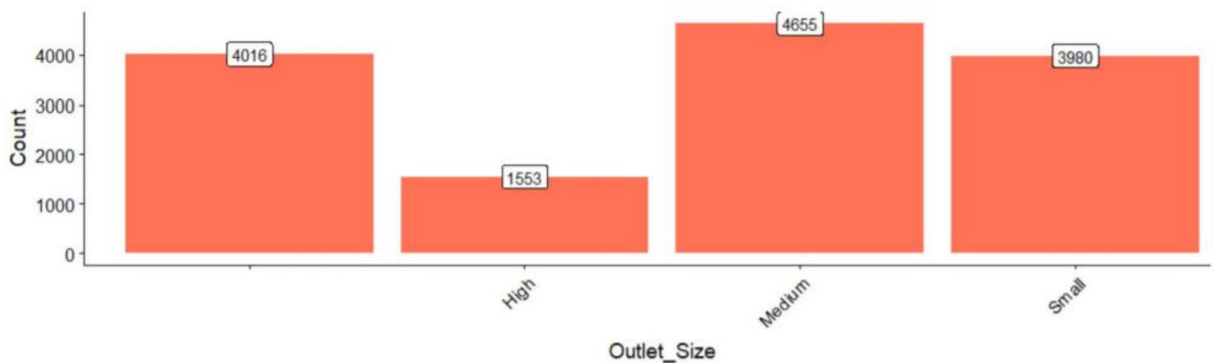

```
> p4 = ggplot(combi %>% group_by(Item_Type) %>% summarise(Count = n())) +
  geom_bar(aes(Item_Type, Count), stat = "identity", fill = "darkgreen") +
  xlab("") +
  geom_label(aes(Item_Type, Count, label = Count), vjust = 0.5) +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))+
  ggtitle("Item_Type")
```

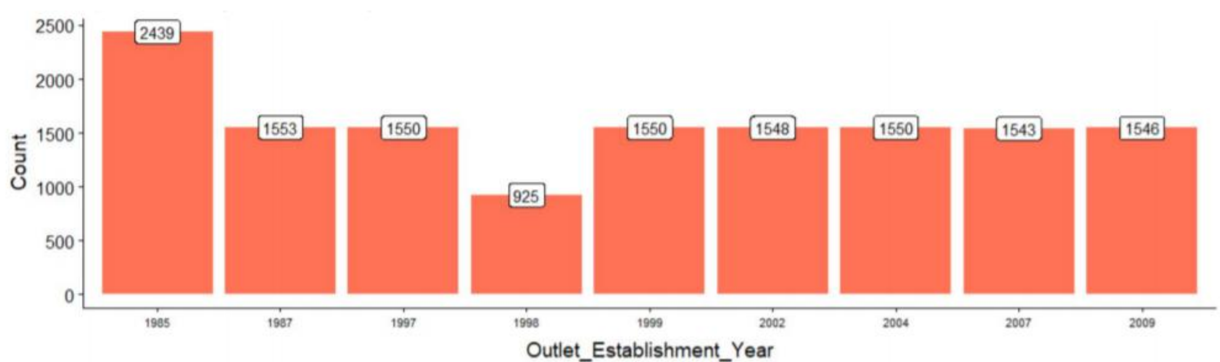


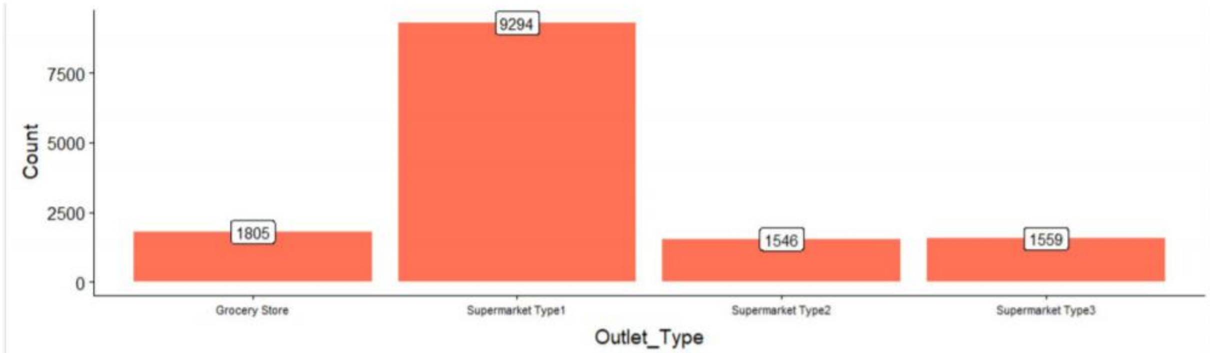

```
> p5 = ggplot(combi %>% group_by(Outlet_Identifier) %>% summarise(Count = n())) + geom_bar(aes(Outlet_Identifier, Count), stat = "identity", fill = "coral1") + geom_label(aes(Outlet_Identifier, Count, label = Count), vjust = 0.5) + theme(axis.text.x = element_text(angle = 45, hjust = 1))
```




```
> p6 = ggplot(combi %>% group_by(Outlet_Size) %>% summarise(Count = n())) +
geom_bar(aes(Outlet_Size, Count), stat = "identity", fill = "coral1") +
geom_label(aes(Outlet_Size, Count, label = Count), vjust = 0.5) +
theme(axis.text.x = element_text(angle = 45, hjust = 1))
```







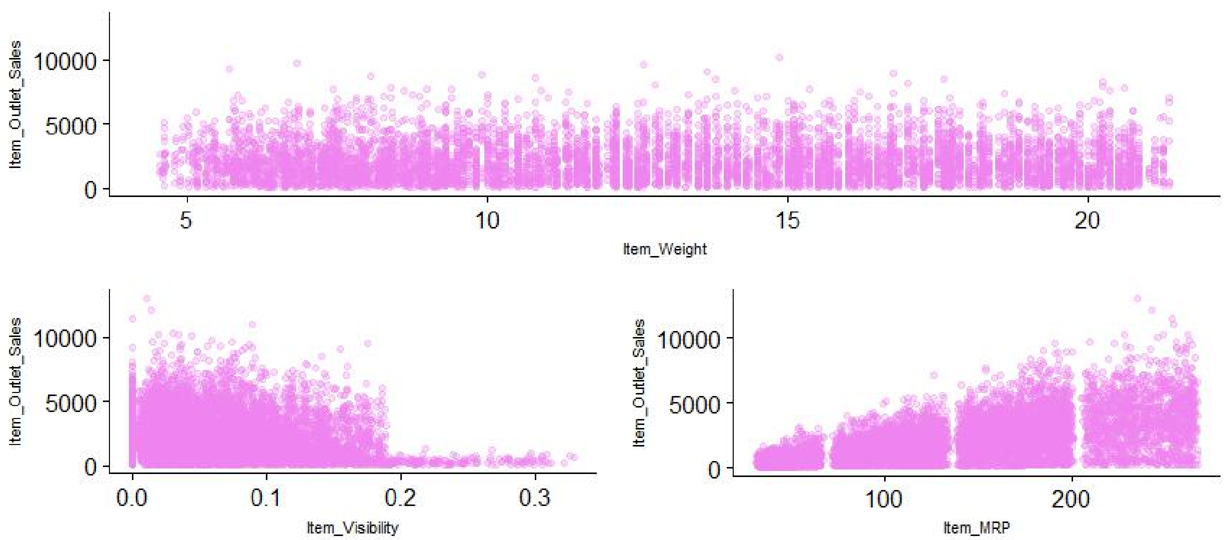
Observation:

• 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.

(ii) Bivariate Analysis

```
> p9 = ggplot(train) + geom_point(aes(Item_Weight, Item_Outlet_Sales), colour
= "violet", alpha = 0.3) + theme(axis.title = element_text(size = 8.5))
> p10 = ggplot(train) + geom_point(aes(Item_Visibility, Item_Outlet_Sales),
colour = "violet", alpha = 0.3) + theme(axis.title = element_text(size = 8.5))
> p11 = ggplot(train) + geom_point(aes(Item_MRP, Item_Outlet_Sales), colour =
"violet", alpha = 0.3) + theme(axis.title = element_text(size = 8.5))
```

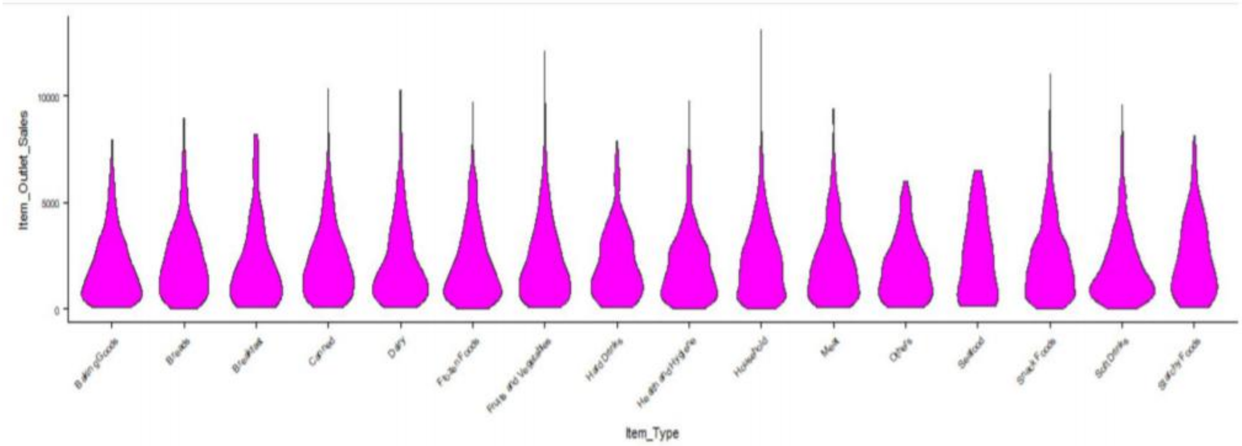


Observations:

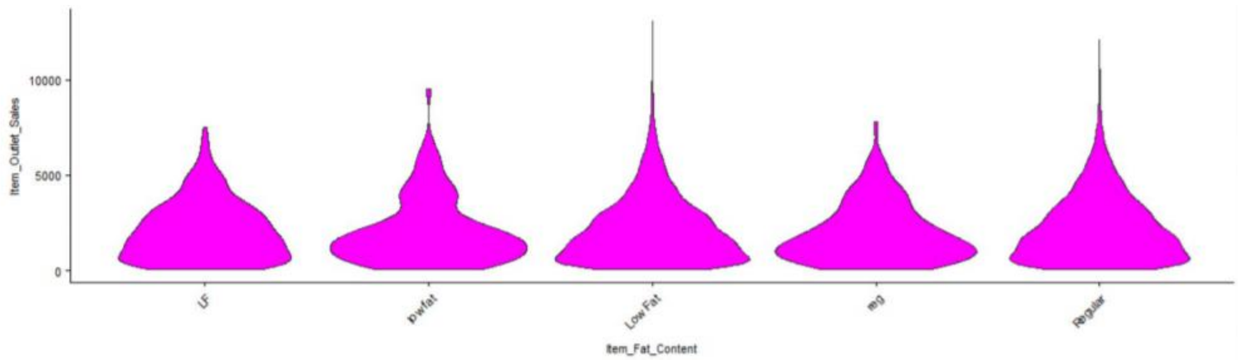
- Item_Outlet_Sales is spread well across the entire range of the Item_Weight without any obvious pattern.
- In Item_Visibility vs Item_Outlet_Sales, there is a string of points at Item_Visibility = 0.0 which seems strange as item visibility cannot be completely zero. We will take note of this issue and deal with it in the later stages.
- In the third plot of Item_MRP vs Item_Outlet_Sales, we can clearly see 4 segments of prices that can be used in feature engineering to create a new variable.

Target Variable vs Independent Categorical Variables

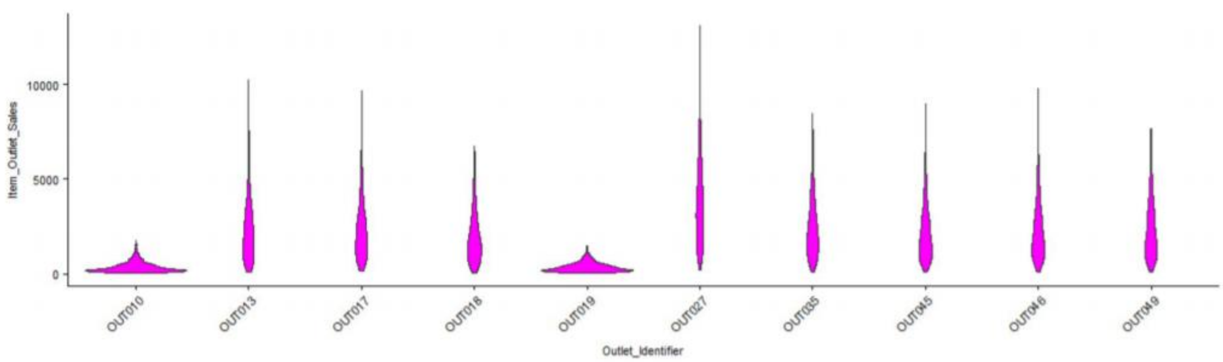
```
> p12 = ggplot(train) + geom_violin(aes(Item_Type, Item_Outlet_Sales), fill = "magenta") + theme(axis.text.x = element_text(angle = 45, hjust = 1), axis.text = element_text(size = 6), axis.title = element_text(size = 8.5))
```




```
> p13 = ggplot(train) + geom_violin(aes(Item_Fat_Content, Item_Outlet_Sales), fill = "magenta") + theme(axis.text.x = element_text(angle = 45, hjust = 1), axis.text = element_text(size = 8), axis.title = element_text(size = 8.5))
```



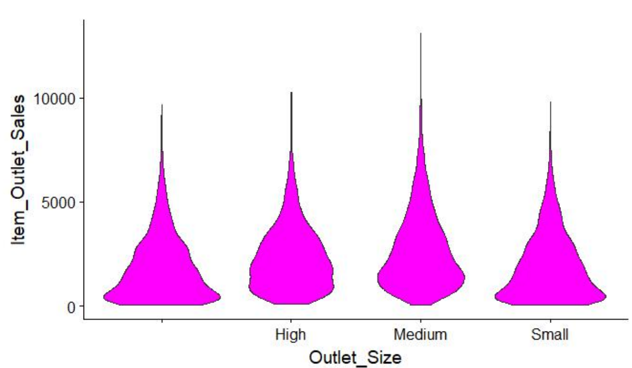

```
> p14 = ggplot(train) + geom_violin(aes(Outlet_Identifier, Item_Outlet_Sales),
fill = "magenta") + theme(axis.text.x = element_text(angle = 45, hjust = 1),
axis.text = element_text(size = 8), axis.title = element_text(size = 8.5))
```



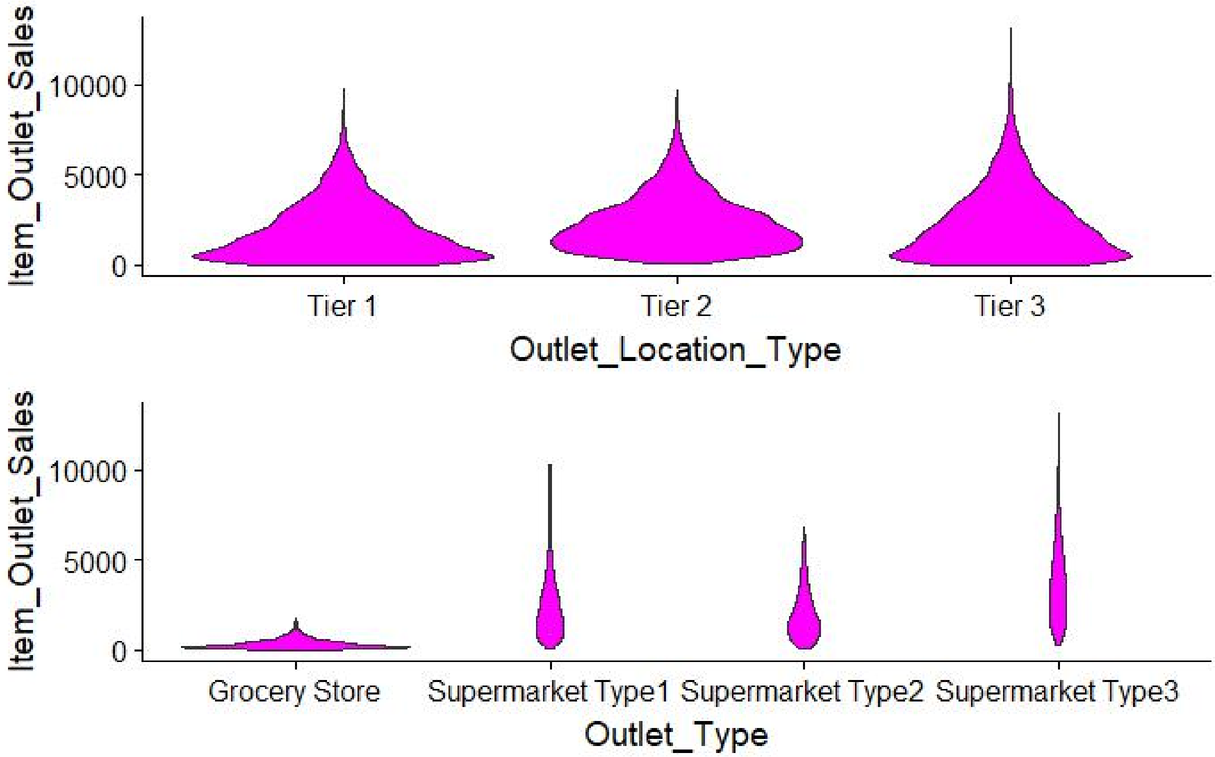
Observations :

• Distribution of Item_Outlet_Sales across the categories of Item_Type is not very distinct and same is the case with Item_Fat_Content.

• The distribution for OUT010 and OUT019 categories of Outlet_Identifier are quite similar and very much different from the rest of the categories of Outlet_Identifier.



```
>p15=ggplot(train) + geom_violin(aes(Outlet_Location_Type, Item_Outlet_Sales),
fill = "magenta") p16 = ggplot(train) + geom_violin(aes(Outlet_Type,
Item_Outlet_Sales), fill = "magenta") plot_grid(p15, p16, ncol = 1)
```



Observations :

- Tier 1 and Tier 3 locations of Outlet_Location_Type look similar.
- In the Outlet_Type plot, Grocery Store has most of its data points around the lower sales values as compared to the other categories.

5.) Missing Value Treatment

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. The downside of this method is the loss of information and drop in prediction power of model.

2. **) Mean/Median/Mode Imputation:** In case of continuous variable, missing values can be replaced with 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.

3.) **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 dataset - 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.

```
> sum(is.na(combi$Item_Weight))
[1] 2439
>
```

Imputing Missing Value

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 the mean weight based on the Item Identifier variable.

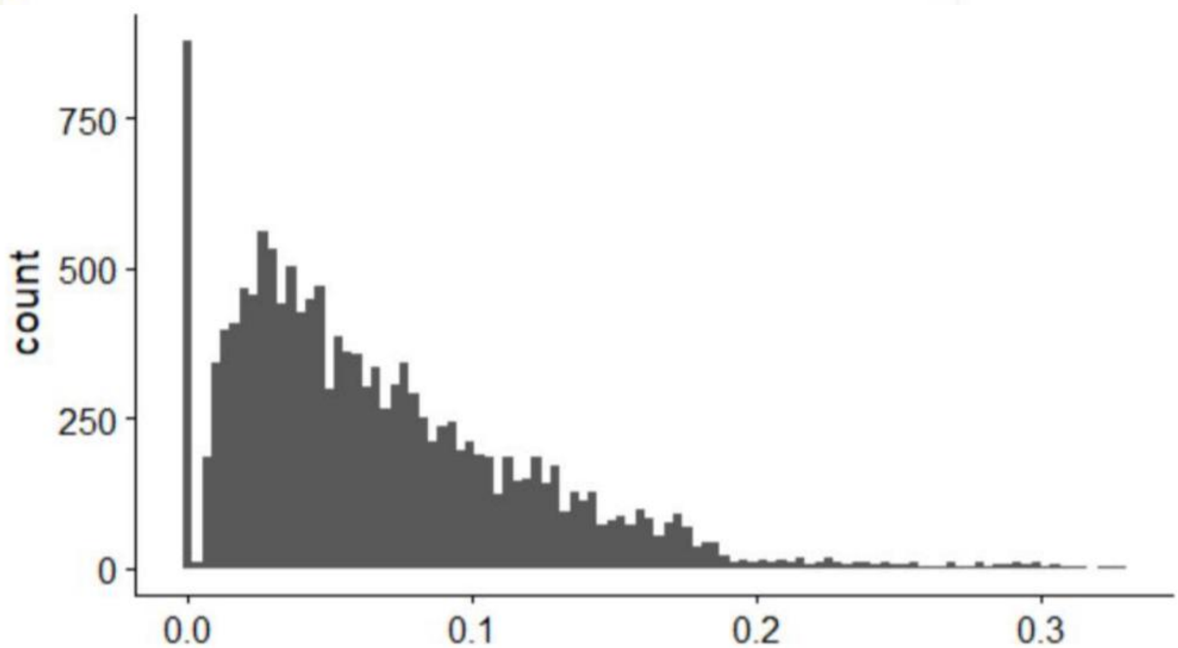
```
> missing_index = which(is.na(combi$Item_Weight))
> for(i in missing_index){

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

}
```

let's see if there is still any missing data in Item Weight

```
> sum (is.na (combisItem_Weight))
[1] 0
```

Replacing Os in Item_Visibility variable

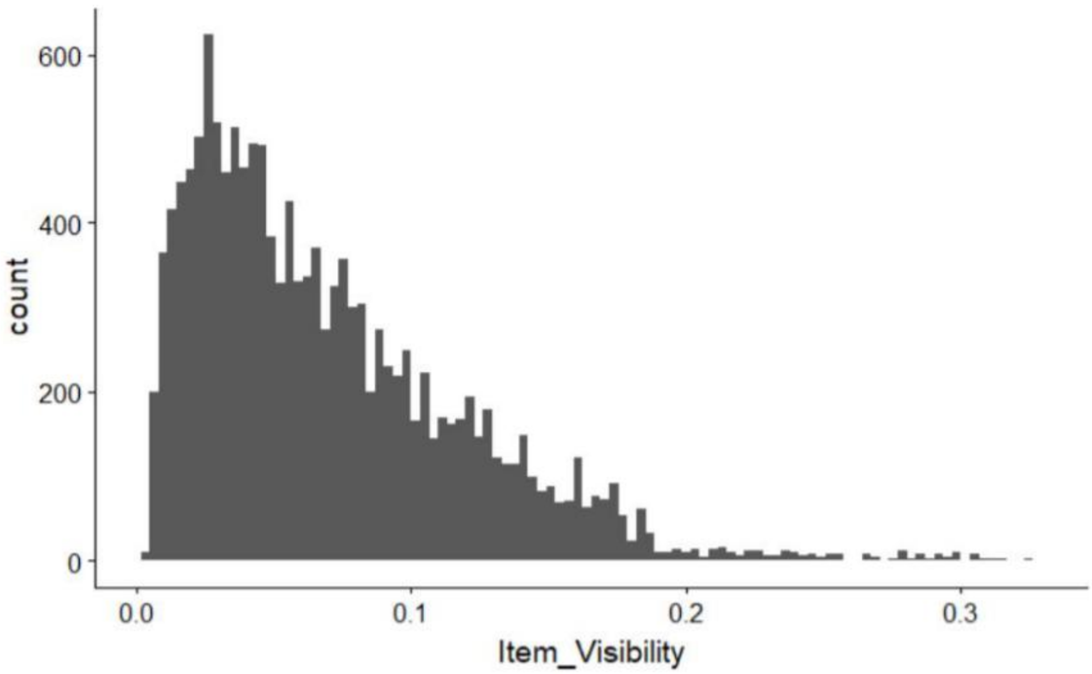
```
> ggplot(combi) + geom_histogram(aes(Item_Visibility), bins = 100)
>
```



```
> 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)
}
```

After the replacement of zeroes, We'll plot the histogram of Item_Visibility again. In the histogram, we can see that the issue of zero item visibility has been resolved.

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



Data Preprocessing:-

Data pre-processing refers to the transformations applied to your data before feeding it to the algorithm. It invloves further cleaning of data, data transformation, data scaling and many more things.

For our data, we will deal with the skewness and scale the numerical variables

Removing Skewness

Skewness in variables is undesirable for predictive modeling. Some machine learning methods assume normally distributed data and a skewed variable can be transformed by taking its log, square root, or cube root so as to make its distribution as close to normal distribution as possible. In our data, variables Item_Visibility and price_per_unit_wt are highly skewed. So, we will treat their skewness with the help of log transformation.

```
> combi[,Item_Visibility:=log(Item_Visibility+1)]
> combi[,price_per_unit_wt := log(price_per_unit_wt + 1)]
```

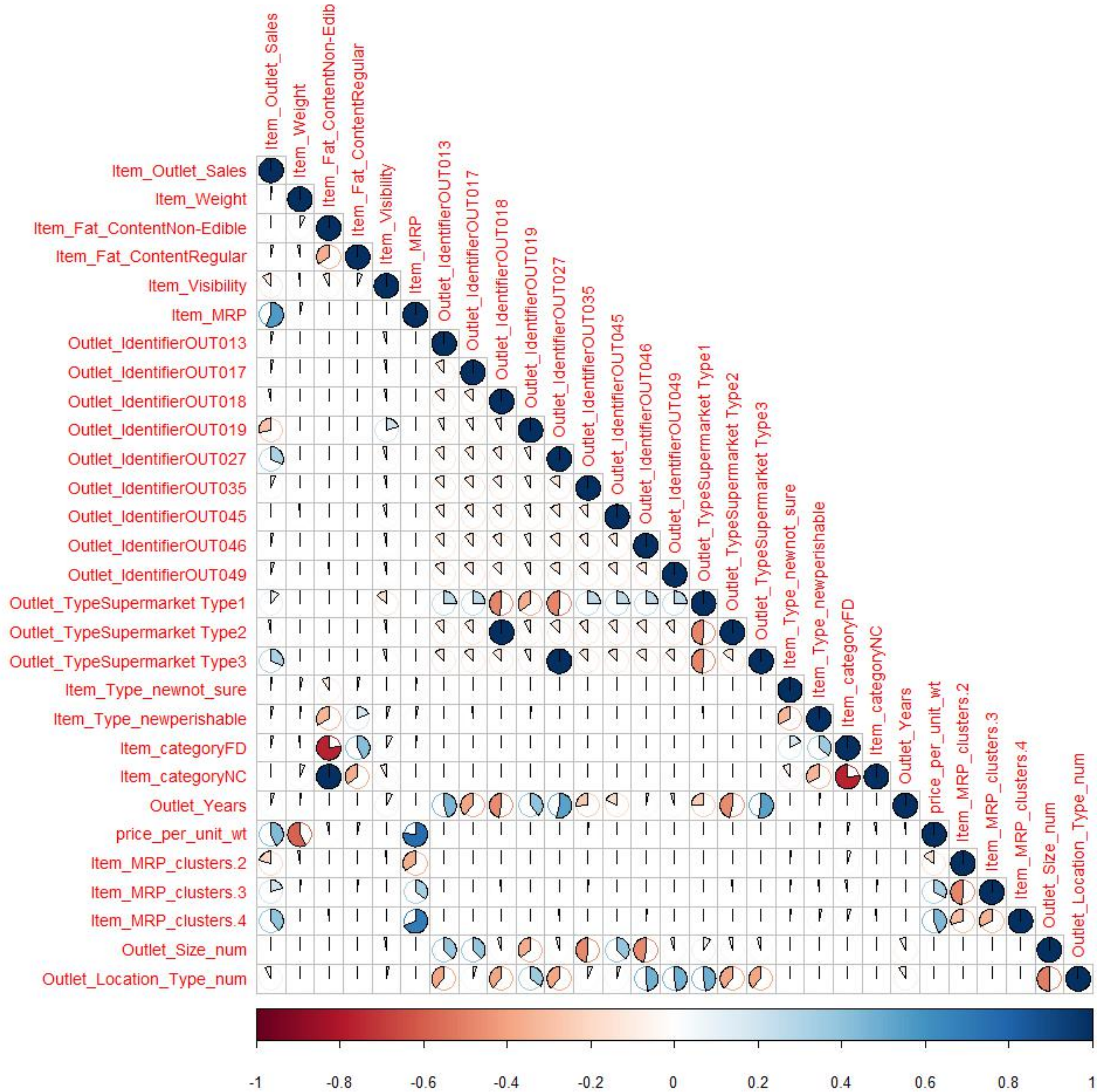
Correlated Variables

Let's examine the correlated features of train dataset. Correlation varies from -1 to 1.

1. negative correlation: < 0 and >= -1
2. positive correlation: > 0 and <= 1
3. no correlation: 0

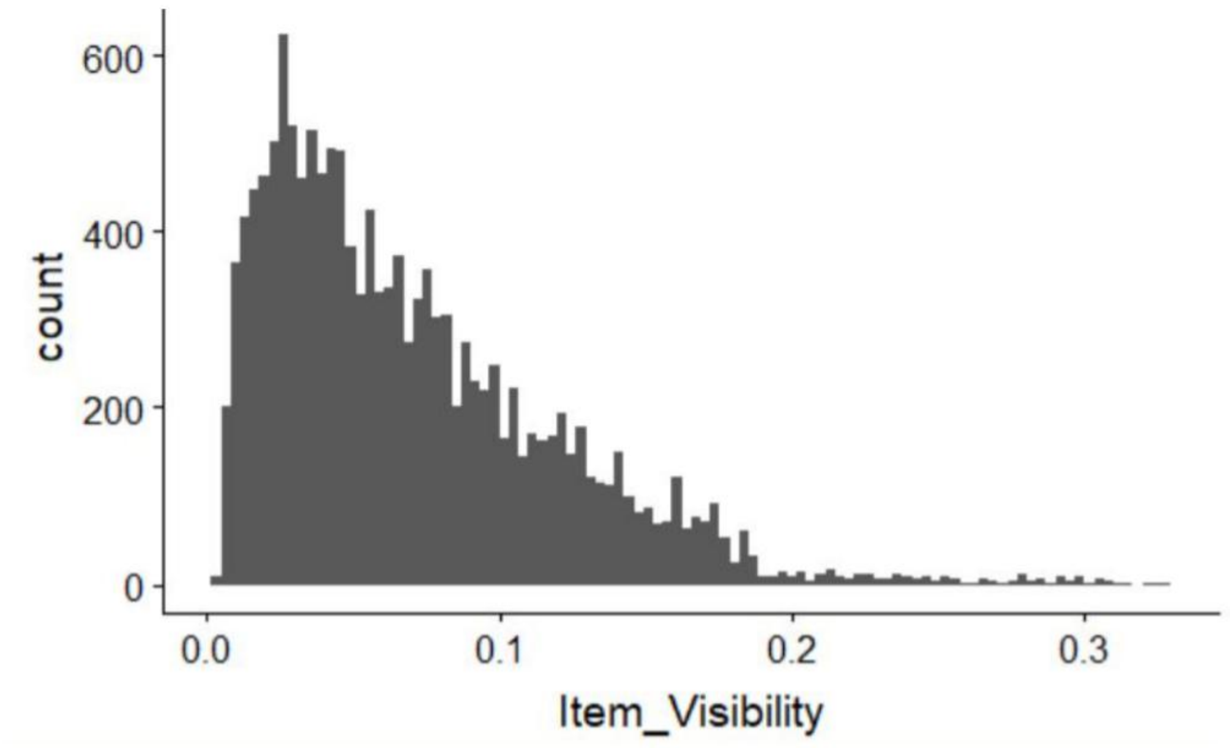
It is not desirable to have correlated features if we are using linear regressions.

```
> cor_train = cor(train[,-c("Item_Identifier")])
> corrplot(cor_train, method = "pie", type = "lower", tl.cex = 0.9)
```


Item_Outlet_Sales
Item_Weight
Item_Fat_ContentNon-Edible
Item_Fat_ContentRegular
Item_Visibility
Item_MRP
Outlet_IdentifierOUT013
Outlet_IdentifierOUT017
Outlet_IdentifierOUT018
Outlet_IdentifierOUT019
Outlet_IdentifierOUT027
Outlet_IdentifierOUT035
Outlet_IdentifierOUT045
Outlet_IdentifierOUT046
Outlet_IdentifierOUT049
Outlet_TypeSupermarket Type1
Outlet_TypeSupermarket Type2
Outlet_TypeSupermarket Type3
Item_Type_newnot_sure
Item_Type_newperishable
Item_categoryFD
Item_categoryNC
Outlet_Years
price_per_unit_wt
Item_MRP_clusters.2
Item_MRP_clusters.3
Item_MRP_clusters.4
Outlet_Size_num
Outlet_Location_Type_num
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1

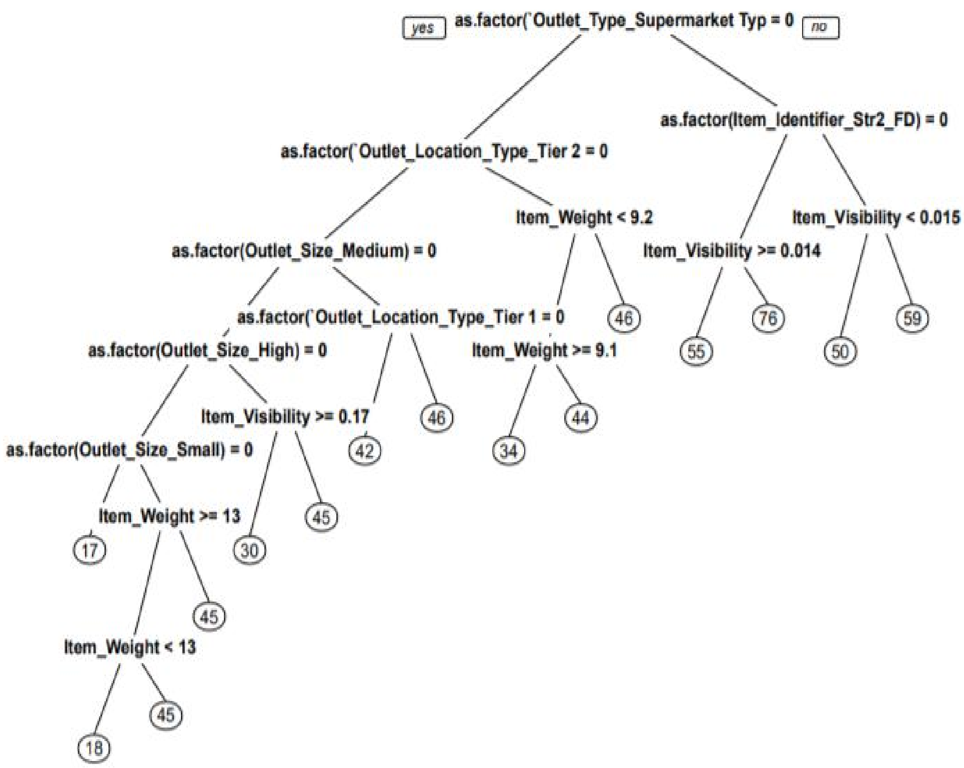
Observation:

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



Decision Tree

```
library(dummies)
new_combi=dummy.data.frame(new_combi,names=c('Outlet_Size','Outlet_location_T
ype','Outlet_Type','Item_Identifier_Str2'),sep = '_')
glimpse(new_combi)

(formula_sqrt_tree=as.formula(sqrt(Item_Outlet_Sales) ~ Item_Weight +
                                   Item_Visibility +
                                   as.factor(Outlet_Size_High) +
                                   as.factor(Outlet_Size_Medium) +
                                   as.factor(Outlet_Size_Small) +
                                   as.factor(`Outlet_Location_Type_Tier 1`) +
                                   as.factor(`Outlet_Location_Type_Tier 2`) +
                                   as.factor(`Outlet_Location_Type_Tier 3`) +
                                   as.factor(`Outlet_Type_Supermarket  Type3`)
+
                                   as.factor(Item_Identifier_Str2_DR) +
                                   as.factor(Item_Identifier_Str2_FD) +
                                   as.factor(Item_Identifier_Str2_NC)))
pred_train=new_combi %>%
  filter(Item_Outlet_Sales != -999)
pred_test=new_combi %>%
  filter(Item_Outlet_Sales == -999)
set.seed(1)
n=nrow(pred_train)
shuffled=pred_train[sample(n),]

train_indices <- 1:round(0.7*n)
test_indices <- (round(0.7*n)+1):n
splitted_train <- shuffled[train_indices,]
splitted_test <- shuffled[test_indices,]

library(rpart)
library(e1071)
library(rpart.plot)
library(caret)
main_tree=rpart(formula_sqrt_tree,   data   =   splitted_train,   control   =
rpart.control(cp=0.001))
prp(main_tree)
```


yes
as.factor(`Outlet_Type_Supermarket Typ = 0
no
as.factor(`Outlet_Location_Type_Tier 2 = 0
as.factor(Item_Identifier_Str2_FD) = 0
Item_Weight < 9.2
Item_Visibility < 0.015
as.factor(Outlet_Size_Medium) = 0
Item_Visibility >= 0.014
as.factor(`Outlet_Location_Type_Tier 1 = 0
46
76
59
as.factor(Outlet_Size_High) = 0
Item_Weight >= 9.1
55
50
Item_Visibility >= 0.17
46
44
as.factor(Outlet_Size_Small) = 0
42
34
Item_Weight >= 13
45
17
30
45
Item_Weight < 13
45
18

Prediction Part using Python: -

Linear Regression Algorithm

Linear regression is the simplest and most widely used statistical technique for predictive modeling. Given below is the linear regression equation:

where X1, X2,…,Xn are the independent variables, Y is the target variable and all thetas are the coefficients. Magnitude of a coefficient wrt to the other coefficients determines the importance of the corresponding independent variable.

One of these assumptions is that of absence of multicollinearity, i.e, the independent variables should be correlated. However, as per the correlation plot above, we have a few highly correlated independent variables in our data. This issue of multicollinearity can be dealt with regularization.

For the time being, let’s build our linear regression model with all the variables.

Code: -

```
import pandas as pd
train=pd.read_csv("train.csv")
test=pd.read_csv("test.csv")
train['source']='train'
test['source']='test'
data=pd.concat([train,test],ignore_index=True)

#Implementing one-hot-Coding method for getting the categorical variables
from sklearn.preprocessing import LabelEncoder
```

```
le = LabelEncoder()
data['Outlet']=le.fit_transform(data['Outlet_Identifier'])
var_mod=['Item_Fat_Content','Outlet_Location_Type','Outlet_Size','Item_Type','O
utlet_ Type']
le = LabelEncoder()
for i in var_mod:
 data[i]=le.fit_transform(data[i])
```

#One Hot Encoding:

```
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 commonly 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)
```

#Define target and ID columns:

```
  target = 'Item_Outlet_Sales'
  IDcol = ['Item_Identifier','Outlet_Identifier']
  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 on testing data:
  dtest[target] = alg.predict(dtest[predictors])
```

#Export submission file:

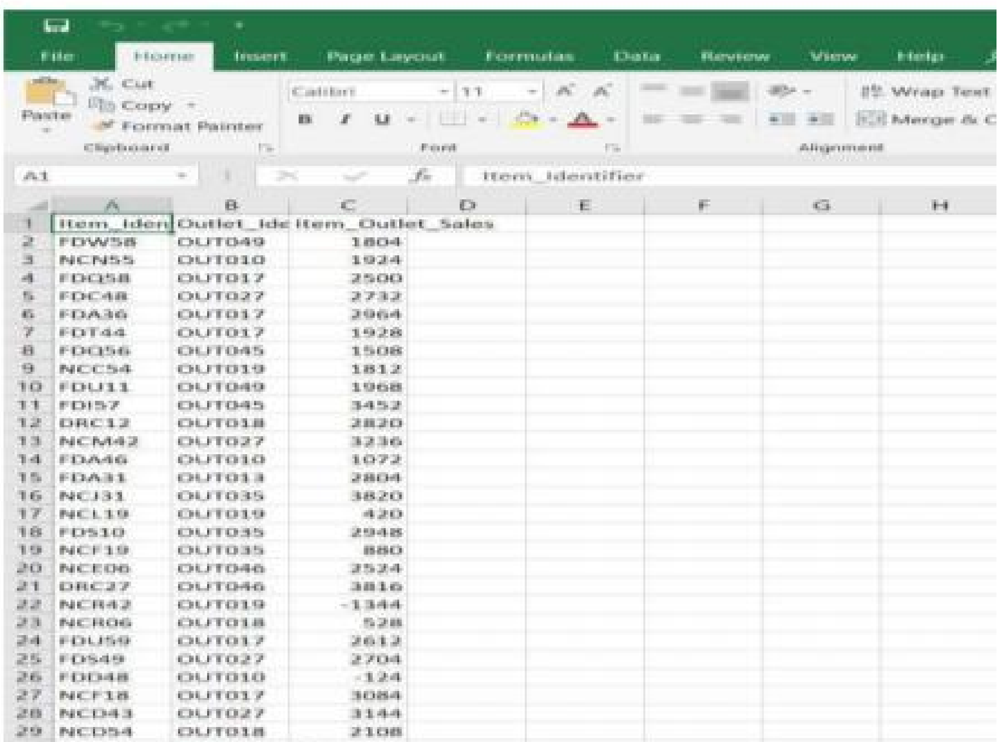
```
IDcol.append(target) submission = pd.DataFrame({ x: dtest[x] for x in IDcol})
submission.to_csv(filename, index=False)
```

#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')
```

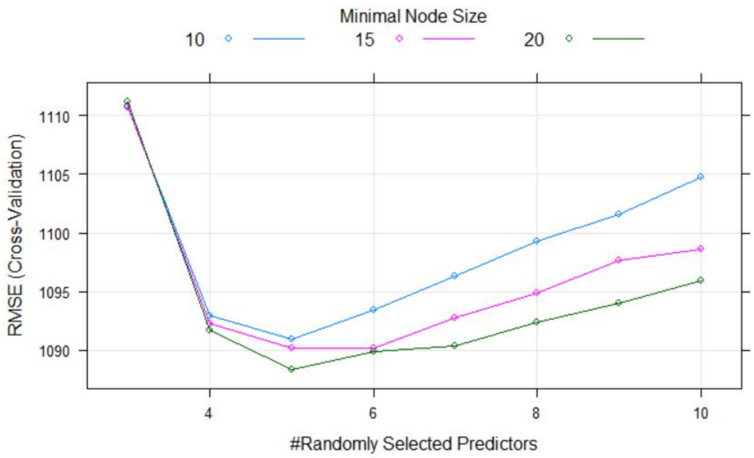


	A	B	C
1	Item_Iden	Outlet_Ide	Item_Outlet_Sales
2	FDW58	OUT049	1804
3	NCN55	OUT010	1924
4	FDQ58	OUT017	2500
5	FDC48	OUT027	2732
6	FDA36	OUT017	2964
7	FDT44	OUT017	1928
8	FDQ56	OUT045	1508
9	NCC54	OUT019	1812
10	FDU11	OUT049	1968
11	FDI57	OUT045	3452
12	DRC12	OUT018	2820
13	NCM42	OUT027	3236
14	FDA46	OUT010	1072
15	FDA31	OUT013	2804
16	NCJ31	OUT035	3820
17	NCL19	OUT019	420
18	FDS10	OUT035	2948
19	NCF19	OUT035	880
20	NCE06	OUT046	2524
21	DRC27	OUT046	3816
22	NCR42	OUT019	-1344
23	NCR06	OUT018	528
24	FDU59	OUT017	2612
25	FDS49	OUT027	2704
26	FDD48	OUT010	-124
27	NCF18	OUT017	3084
28	NCD43	OUT027	3144
29	NCD54	OUT018	2108

Random Forest Algorithm

Random Forest is a tree based bootstrapping algorithm wherein a certain no. of weak learners (decision trees) are combined to make a powerful prediction model. For every individual learner, a random sample of rows and a few randomly chosen variables are used to build a decision tree model. Final prediction can be a function of all the predictions made by the individual learners. In case of regression problem, the final prediction can be mean of all the predictions. We will now build a Random Forest model with 400 trees. The other tuning parameters used here are mtry-no. of predictor variables randomly sampled at each split, and min.node.size — minimum size of terminal nodes (setting this number large causes smaller trees and reduces overfitting).

```
> set.seed(1237)
> my_control = trainControl(method="cv", number=5)

> tgrid = expand.grid(
  .mtry = c(3:10),
  .splitrule = "variance",
  .min.node.size = c(10,15,20)
)

> rf_mod = train(x = train[, -c("Item_Identifier", "Item_Outlet_Sales")],
                y = train$Item_Outlet_Sales,
                method='ranger',
                trControl= my_control,
                tuneGrid = tgrid,
                num.trees = 400,
                importance = "permutation")
> plot(rf_mod)
```



XGBoost Model

XGBoost is a fast and efficient algorithm and a boosting algorithm .XGBoost works only with numeric variables and we have already done that. There are many tuning parameters in XGBoost which can be broadly classified into General Parameters, Booster Parameters and Task Parameters.

- General parameters refers to which booster we are using to do boosting. The commonly used are tree or linear model
- Booster parameters depends on which booster you have chosen
- Learning Task parameters that decides on the learning scenario, for example, regression tasks may use different parameters with ranking tasks.

Let’s have a look at the parameters that we are going to use in our model.

1. **eta**: It is also known as the learning rate or the shrinkage factor. It actually shrinks the feature weights to make the boosting process more conservative. The range is 0 to 1. Low eta value means model is more robust to overfitting.
2. **gamma**: The range is 0 to ∞. Larger the gamma more conservative the algorithm is.
3. **max_depth**: We can specify maximum depth of a tree using this parameter.

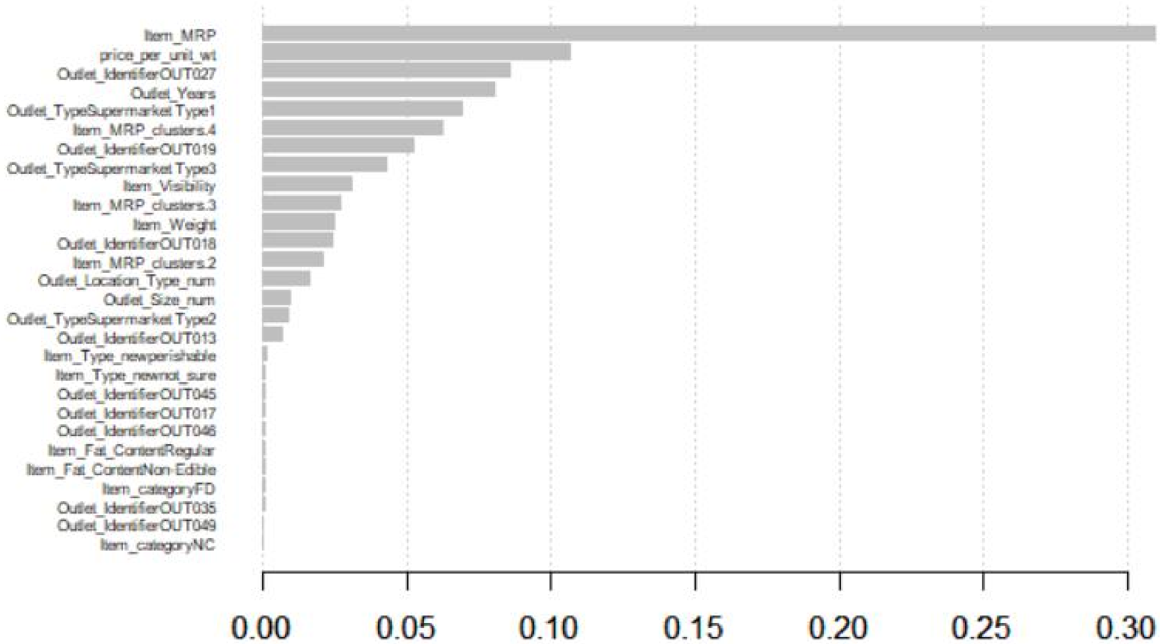
4. **subsample**: It is the proportion of rows that the model will randomly select to grow trees.
5. **colsample_bytree**: It is the ratio of variables randomly chosen for build each tree in the model.

```
> param_list = list(
  objective = "reg:linear",
  eta=0.01,
  gamma = 1,
  max_depth=6,
  subsample=0.8,
  colsample_bytree=0.5
)
> dtrain = xgb.DMatrix(data = as.matrix(train[,-c("Item_Identifier",
"Item_Outlet_Sales")]), label= train$Item_Outlet_Sales)
>dtest = xgb.DMatrix(data = as.matrix(test[,-c("Item_Identifier")]))
>set.seed(112)
xgbcv = xgb.cv(params = param_list,
               data = dtrain,
               nrounds = 1000,
               nfold = 5,
               print_every_n = 10,
               early_stopping_rounds = 30,
               maximize = F)
>xgb_model = xgb.train(data = dtrain, params = param_list, nrounds = 430)
```

CONCLUSION

Variable Importance

```
>var_imp=xgb.importance(feature_names=setdiff(names(train),
c("Item_Identifier", "Item_Outlet_Sales")),model = xgb_model)
>xgb.plot.importance(var_imp)
```



As expected Item_MRP is the most important variable in predicting the target variable. New features created by us, like price_per_unit_wt, Outlet_Years, Item_MRP_Clusters, are also among the top most important variables. This is why feature engineering plays such a crucial role in predictive modeling.

In future work, we can use the output of this project as part of the price optimization problem which can be used by BigMart to optimize their products prices according to sales of the outlets.

<u>**REFERENCES**</u>

[1] Tanu Jain* Dr. A.K Dua Varun Sharma, 'Quantitative Analysis of Apriori and Eclat Algorithm for Association Rule Mining' International Journal Of Engineering And Computer Science ISSN: 2319-7242 Volume 4 Issue 10 Oct 2015, Page No. 14649- 14652.

[2] Wenjie H., Qing Z., Wei X., Hongjiao F., Mingming W., and Xun L. 'A Novel Trigger Model for Sales Prediction with Data Mining Techniques', Data Science Journal, 14: 15, 2015,pp. 1–8.

[3] Eric T.,Manish G.,Praveen K,∗ ,1,' 'The Role of Big Data and Predictive Analytics in Retailing', journal of retailing 93,2017,pp.79-95.

[4] Priyanka P., Kavita S., Oza Ass., K. Kamat., 'Big Data Predictive Analysis:Using R Analytical Tool',International conference on I-SMAC 2017,pp.839-842

[5] A Forecast for Big Mart Sales Based on Random Forests and Multiple Linear Regression 1Heramb Kadam, 2Rahul Shevade, 3 Prof. Deven Ketkar, 4Mr. Sufiyan Rajguru 1 BE IT, FAMT, Ratnagiri 2 BE IT, FAMT, Ratnagiri, 3 Assistant Professor ,IT department, FAMT, 4 BE IT, FAMT, Ratnagiri.