Analysis of Big-Mart Dataset

dataMunglers

15 December 2017

Contents

Team Members 1
Our Objective 1
Dataset 1
About BigMart 2
1. Exploring The BigMart Dataset 2
1.1. Factor Columns 2
1.2. Numerical Columns 5
2. Data Manipulation 8
3. Data Visualizing 9
3.1. Visualizing Manipulated Data 9
3.2. Visualizing Relationship between Item__Outlet__Sales and Other Categorical Columns 12
3.3. Visualizing Relationship between Item__Outlet__Sales and Other Numerical Columns 24
4. Modeling 28
4.1. Discover Models 28
4.2. Summary of Modelling 37
References 41
Dataset 41
Analysis & Graphs 41
Modeling 42

Team Members

- ***Ahmet Yetkin Eser***
- ***Berkay Soyer***
- ***Feray Ece Topcu***

Our Objective

- The data scientists at BigMart have collected 2013 sales data for 1559 products across 10 stores in different cities. The main objective is to understand whether specific properties of products and/or stores play a significant role in terms of increasing or decreasing sales volume. To achieve this goal, we will build a predictive model and find out the sales of each product at a particular store. This will help BigMart to boost their sales by learning optimised product organization inside stores.

Dataset

- We take data from analyticsvidhya.com, Our data is Big Mart Sales Practise Problem Data, It is open competition now and its final date is 31 Dec 2017. We downloaded training and test dataset as a csv file.

About BigMart

- Big Mart is One Stop Shopping center and Free Marketplace. Buy, sell and advertise without fee or at low cost. Find more information about BigMart Here.

1. Exploring The BigMart Dataset

- Call Libraries and Read Data from File.

```r
library(dplyr)
library(tidyr)
library(stringr)
library(ggplot2)
library(corrplot)
library(dummies)
setwd("C:/Users/ecetp/Downloads/MEF/BDA 503 - R/Project-BigMart Sales")
#setwd("/Users/yetkineser/Desktop/mef R/data")
train <- read.csv('bigMartTrain.csv')
test  <- read.csv('bigMartTest.csv')
```

- View the structure of train with ***glimpse*** function.

```r
glimpse(train)
```

```
## Observations: 8,523
## Variables: 12
## $ Item_Identifier           <fctr> FDA15, DRC01, FDN15, FDX07, NCD19, ...
## $ Item_Weight               <dbl> 9.300, 5.920, 17.500, 19.200, 8.930,...
## $ Item_Fat_Content          <fctr> Low Fat, Regular, Low Fat, Regular,...
## $ Item_Visibility           <dbl> 0.016047301, 0.019278216, 0.01676007...
## $ Item_Type                 <fctr> Dairy, Soft Drinks, Meat, Fruits an...
## $ Item_MRP                  <dbl> 249.8092, 48.2692, 141.6180, 182.095...
## $ Outlet_Identifier         <fctr> OUT049, OUT018, OUT049, OUT010, OUT...
## $ Outlet_Establishment_Year <int> 1999, 2009, 1999, 1998, 1987, 2009, ...
## $ Outlet_Size               <fctr> Medium, Medium, Medium, , High, Med...
## $ Outlet_Location_Type      <fctr> Tier 1, Tier 3, Tier 1, Tier 3, Tie...
## $ Outlet_Type               <fctr> Supermarket Type1, Supermarket Type...
## $ Item_Outlet_Sales         <dbl> 3735.1380, 443.4228, 2097.2700, 732....
```

- As a conclusion of glimpse, we can see that our train dataset has 12 columns and 8523 rows. We should observe all columns one by one to explore data.

1.1. Factor Columns

- ***Item_Identifier*** : Unique Product ID.

```r
train %>%
  summarise(n_distinct(Item_Identifier))
```

```
##   n_distinct(Item_Identifier)
## 1                        1559
```

Item_Identifier column has 1559 unique value. So, we can say we will examine 1559 unique item properties due to the stores.

- ***Item_Fat_Content*** : Whether the product is low fat or not.

```
train %>%
  group_by(Item_Fat_Content) %>%
  summarise(Count = n(),Perc=round(n()/nrow(.)*100,2)) %>%
  arrange(desc(Count))
```

```
## # A tibble: 5 x 3
##   Item_Fat_Content Count  Perc
##             <fctr> <int> <dbl>
## 1          Low Fat  5089 59.71
## 2          Regular  2889 33.90
## 3               LF   316  3.71
## 4              reg   117  1.37
## 5          low fat   112  1.31
```

Item_Fat_Content distribution can be seen as belove summary. Low fat items have highest rate, so we can say low fat items reside on stores generally. There is corruption about string values, such as "Low fat"/"low fat"/"LF" or "Regular" / "reg" , we know these string values represent same value and so, we should clean this column values on cleaning part.

- ***Item_Type*** : The category to which the product belongs.

```
train%>%
  group_by(Item_Type) %>%
  summarise(Count = n(), Perc = round(n() / nrow(.) * 100, 2)) %>%
  arrange(desc(Count))
```

```
## # A tibble: 16 x 3
##                Item_Type Count  Perc
##                   <fctr> <int> <dbl>
##  1 Fruits and Vegetables  1232 14.46
##  2           Snack Foods  1200 14.08
##  3             Household   910 10.68
##  4          Frozen Foods   856 10.04
##  5                 Dairy   682  8.00
##  6                Canned   649  7.61
##  7          Baking Goods   648  7.60
##  8    Health and Hygiene   520  6.10
##  9           Soft Drinks   445  5.22
## 10                  Meat   425  4.99
## 11                Breads   251  2.94
## 12           Hard Drinks   214  2.51
## 13                Others   169  1.98
## 14         Starchy Foods   148  1.74
## 15             Breakfast   110  1.29
## 16               Seafood    64  0.75
```

Train dataset has 16 different item_type value and "Fruits and Vegetables" items reside on stores popularly.

- ***Outlet_Identifier*** : Unique Store ID

```
train %>%
  group_by(Outlet_Identifier) %>%
  summarise(Count = n(), Perc = round(n() / nrow(.) * 100, 2)) %>%
  arrange(desc(Count))
```

```
## # A tibble: 10 x 3
##    Outlet_Identifier Count  Perc
##                <fctr> <int> <dbl>
##  1             OUT027   935 10.97
##  2             OUT013   932 10.94
##  3             OUT035   930 10.91
##  4             OUT046   930 10.91
##  5             OUT049   930 10.91
##  6             OUT045   929 10.90
##  7             OUT018   928 10.89
##  8             OUT017   926 10.86
##  9             OUT010   555  6.51
## 10             OUT019   528  6.20
```

Outlet_Identifier data show that train dataset includes item information of 10 different stores.

- ***Outlet_Size*** : The size of the store in terms of ground area covered.

```
train%>%
  group_by(Outlet_Size) %>%
  summarise(Count = n(),Perc = round(n() / nrow(.) * 100, 2)) %>%
  arrange(desc(Count))
```

```
## # A tibble: 4 x 3
##   Outlet_Size Count  Perc
##        <fctr> <int> <dbl>
## 1      Medium  2793 32.77
## 2              2410 28.28
## 3       Small  2388 28.02
## 4        High   932 10.94
```

As we can see, outlet_size have some null values. So, on cleaning part we should consider about this column,also.

- ***Outlet_Location_Type*** : The type of city in which the store is located

```
train%>%
  group_by(Outlet_Location_Type) %>%
  summarise(Count = n(),Perc = round(n()/nrow(.) * 100, 2)) %>%
  arrange(desc(Count))
```

```
## # A tibble: 3 x 3
##   Outlet_Location_Type Count  Perc
##                 <fctr> <int> <dbl>
## 1               Tier 3  3350 39.31
## 2               Tier 2  2785 32.68
## 3               Tier 1  2388 28.02
```

Data of Outlet_Location_Type column distribution is fair for each type.

- ***Outlet_Type*** : Whether the outlet is just a grocery store or some sort of supermarket

```
train%>%
  group_by(Outlet_Type)%>%
  summarise(Count=n(),Perc=round(n()/nrow(.)*100,2))%>%
  arrange(desc(Count))
```

```
## # A tibble: 4 x 3
##         Outlet_Type Count  Perc
##              <fctr> <int> <dbl>
## 1 Supermarket Type1  5577 65.43
## 2     Grocery Store  1083 12.71
## 3 Supermarket Type3   935 10.97
## 4 Supermarket Type2   928 10.89
```

This result show that we have item information on "Supermarket Type1" store with the highest rate (65.43). It means, our data includes "Supermarket Type1" store mostly.

1.2. Numerical Columns

- ***Item_Weight*** : Weight of product.

```
train%>%
    summarise(is_NULL=sum(is.na(Item_Weight)==1),
              is_NOT_NULL=sum(!is.na(Item_Weight)==1)
              )
```

```
##   is_NULL is_NOT_NULL
## 1    1463        7060
```

```
train%>%
  filter(!is.na(Item_Weight))%>%
  summarise(
    Max=max(Item_Weight),
    Min=min(Item_Weight),
    Mean=mean(Item_Weight),
    Median=median(Item_Weight),
    QUA1=quantile(Item_Weight,1/4),
    QUA3=quantile(Item_Weight,3/4),
    IQR=IQR(Item_Weight)
  )
```

```
##     Max   Min     Mean Median    QUA1  QUA3     IQR
## 1 21.35 4.555 12.85765   12.6 8.77375 16.85 8.07625
```

The result show that Item_Weight column has 1463 null value, so cleaning part we should consider this column also. When we exclude the null values, the heaviest item is 21.35 gr and the lightest ite is 4.555 gr.

- ***Item_Visibility*** : The % of total display area of all products in a store allocated to the particular product.

```
train%>%
    summarise(is_NULL=sum(is.na(Item_Visibility)==1),
              is_NOT_NULL=sum(!is.na(Item_Visibility)==1)
              )
```

```
##   is_NULL is_NOT_NULL
## 1       0        8523
```

```
train%>%
  filter(!is.na(Item_Visibility))%>%
  summarise(
    Max=max(Item_Visibility),
    Min=min(Item_Visibility),
    Mean=mean(Item_Visibility),
    Median=median(Item_Visibility),
    QUA1=quantile(Item_Visibility,1/4),
    QUA3=quantile(Item_Visibility,3/4),
    IQR=IQR(Item_Visibility)
  )
```

```
##         Max Min       Mean     Median       QUA1       QUA3        IQR
## 1 0.3283909   0 0.06613203 0.05393093 0.02698948 0.09458529 0.06759582
```

There is no null value on Item_Visibility column but from minumum value we can say there is "0" value on column.

- ***Item_MRP*** : Maximum Retail Price (list price) of the product.

```
train%>%
    summarise(is_NULL=sum(is.na(Item_MRP)==1),
              is_NOT_NULL=sum(!is.na(Item_MRP)==1)
              )
```

```
##   is_NULL is_NOT_NULL
## 1       0        8523
```

```
train%>%
  filter(!is.na(Item_MRP))%>%
  summarise(
    Max=max(Item_MRP),
    Min=min(Item_MRP),
    Mean=mean(Item_MRP),
    Median=median(Item_MRP),
    QUA1=quantile(Item_MRP,1/4),
    QUA3=quantile(Item_MRP,3/4),
    IQR=IQR(Item_MRP)
  )
```

```
##        Max   Min     Mean   Median    QUA1     QUA3     IQR
## 1 266.8884 31.29 140.9928 143.0128 93.8265 185.6437 91.8172
```

Good news, there is no null or "0" value on ITEM_MRP column. MAx price is 266.8884 and min price is 31.29.

- ***Outlet_Establishment_Year*** : The year in which store was established.

```
train%>%
    summarise(is_NULL=sum(is.na(Outlet_Establishment_Year)==1),
              is_NOT_NULL=sum(!is.na(Outlet_Establishment_Year)==1)
              )
```

```
##   is_NULL is_NOT_NULL
## 1       0        8523
```

```
train%>%
  filter(!is.na(Outlet_Establishment_Year))%>%
  summarise(
    Max=max(Outlet_Establishment_Year),
    Min=min(Outlet_Establishment_Year),
    Mean=mean(Outlet_Establishment_Year),
    Median=median(Outlet_Establishment_Year),
    QUA1=quantile(Outlet_Establishment_Year,1/4),
    QUA3=quantile(Outlet_Establishment_Year,3/4),
    IQR=IQR(Outlet_Establishment_Year)
  )
```

```
##    Max  Min     Mean Median QUA1 QUA3 IQR
## 1 2009 1985 1997.832   1999 1987 2004  17
```

The oldest store has opened in 1985 and the newest one has opened in 2009. when we look at mean and median,w e can say stores in dataset are generally old stores. (More than 17-18 years old.)

- ***Item_Outlet_Sales*** : Sales of the product in the particulat store. **This is the outcome variable to be predicted.**

```
train%>%
    summarise(is_NULL=sum(is.na(Item_Outlet_Sales)==1),
              is_NOT_NULL=sum(!is.na(Item_Outlet_Sales)==1)
              )
```

```
##   is_NULL is_NOT_NULL
## 1       0        8523
```

```
train%>%
  filter(!is.na(Item_Outlet_Sales))%>%
  summarise(
    Max=max(Item_Outlet_Sales),
    Min=min(Item_Outlet_Sales),
    Mean=mean(Item_Outlet_Sales),
    Median=median(Item_Outlet_Sales),
    QUA1=quantile(Item_Outlet_Sales,1/4),
    QUA3=quantile(Item_Outlet_Sales,3/4),
    IQR=IQR(Item_Outlet_Sales)
  )
```

```
##        Max   Min     Mean   Median     QUA1     QUA3      IQR
## 1 13086.96 33.29 2181.289 1794.331 834.2474 3101.296 2267.049
```

Item_Outlet_Sales column has no null values.

- If we want to see good summary of numeric columns, we should use **summary** function as below.

```
summary(train)
```

```
##   Item_Identifier  Item_Weight     Item_Fat_Content Item_Visibility
##   FDG33  :  10    Min.   : 4.555   LF      : 316     Min.   :0.00000
##   FDW13  :  10    1st Qu.: 8.774   low fat: 112     1st Qu.:0.02699
##   DRE49  :   9    Median :12.600   Low Fat:5089     Median :0.05393
##   DRN47  :   9    Mean   :12.858   reg    : 117     Mean   :0.06613
##   FDD38  :   9    3rd Qu.:16.850   Regular:2889     3rd Qu.:0.09459
##   FDF52  :   9    Max.   :21.350                    Max.   :0.32839
##   (Other):8467    NA's   :1463
##                   Item_Type       Item_MRP      Outlet_Identifier
##   Fruits and Vegetables:1232   Min.   : 31.29   OUT027 : 935
##   Snack Foods          :1200   1st Qu.: 93.83   OUT013 : 932
##   Household            : 910   Median :143.01   OUT035 : 930
##   Frozen Foods         : 856   Mean   :140.99   OUT046 : 930
##   Dairy                : 682   3rd Qu.:185.64   OUT049 : 930
##   Canned               : 649   Max.   :266.89   OUT045 : 929
##   (Other)              :2994                    (Other):2937
##   Outlet_Establishment_Year Outlet_Size   Outlet_Location_Type
##   Min.   :1985                    :2410   Tier 1:2388
##   1st Qu.:1987              High  : 932   Tier 2:2785
##   Median :1999              Medium:2793   Tier 3:3350
##  Mean   :1998              Small :2388
##  3rd Qu.:2004
##  Max.   :2009
##
##              Outlet_Type   Item_Outlet_Sales
##  Grocery Store    :1083   Min.   :   33.29
##  Supermarket Type1:5577   1st Qu.:  834.25
##  Supermarket Type2: 928   Median : 1794.33
##  Supermarket Type3: 935   Mean   : 2181.29
##                           3rd Qu.: 3101.30
##                           Max.   :13086.97
##
```

2. Data Manipulation

- We discover some columns need to be corrected for a good analysis. So, we should manipulate some part of data. Let's first combine the data sets. This will save our time as we don't need to write separate codes for train and test data sets. To combine the two data frames, we must make sure that they have equal columns, which is not the case. Test data set has one less column (response variable). Let's first add the column. We can give this column any value. An intuitive approach would be to extract the mean value of sales from train data set and use it as placeholder for test variable Item_Outlet_Sales. Anyways, let's make it simple for now. I've taken a value -999. Now, we'll combine the data sets.

```
test$Item_Outlet_Sales <- -999
combi <- rbind(train, test)
```

- We saw on exploring part, Item_Weight column has null values which can affect the result of anaylsis. Impute missing value by median. We are using median because it is known to be highly robust to outliers. Moreover, for this problem, our evaluation metric is RMSE which is also highly affected by outliers. Hence, median is better in this case.

```
combi$Item_Weight[is.na(combi$Item_Weight)] <- median(combi$Item_Weight, na.rm = TRUE)
table(is.na(combi$Item_Weight))
```

```
##
## FALSE
## 14204
```

- Let's take up Item_Visibility. On exploration part above, we saw item visibility has zero value also, which is practically not feasible. Hence, we'll consider it as a missing value and once again make the imputation using median.

```
combi$Item_Visibility <- ifelse(combi$Item_Visibility == 0,
                                median(combi$Item_Visibility), combi$Item_Visibility)
```

- Let's proceed to categorical variables now. During exploration, we saw there are mis-matched levels in variables which needs to be corrected. Item fat content should be corrected.

```
combi$Item_Fat_Content <- str_replace(
           str_replace(
             str_replace(combi$Item_Fat_Content,"LF","Low Fat")
             ,"reg","Regular"),"low fat","Low Fat")

table(combi$Item_Fat_Content)
```

```
##
## Low Fat Regular
##    9185    5019
```

- We need to mutate new columns for more meaningful data. First, we evaluate Item_Identifier column because We discovered Item_Identifier column has special codes to recognize the type of item when we tried to understand data. So, letters on Item_Identifier column means Food, Drinks and etc. and numbers can be meaningful. We observed first three letters and two letters to control which of them is more meaningful. In any case, we hold both of them as two seperate column to use them later but now, we decided to use first two letters. (DR=Drink, FD=Food,NC=Non-Consumable) Secondly, we generate new Outlet_Age column from Outlet_Establishment_Year.

```
##first two and three letter.
combi <-
  combi %>%
  mutate(Item_Identifier_Str3 = substr(Item_Identifier,1,3),  #First three letter of Item_Identifier.
         Item_Identifier_Str2 = substr(Item_Identifier,1,2),  #First second letter of Item_Identifier.
         Item_Identifier_Num=as.numeric(substr(Item_Identifier,4,6)), # Number part of Item_Identifier
         Outlet_Age=2013-Outlet_Establishment_Year, #Outlet Age
         PK=row_number())

#table(combi$Item_Identifier_Str3)

table(combi$Item_Identifier_Str2)
```

```
##
##    DR    FD    NC
##  1317 10201  2686
```

```
#combi %>% summarise(n_distinct(Item_Identifier_Str3))
```

3. Data Visualizing

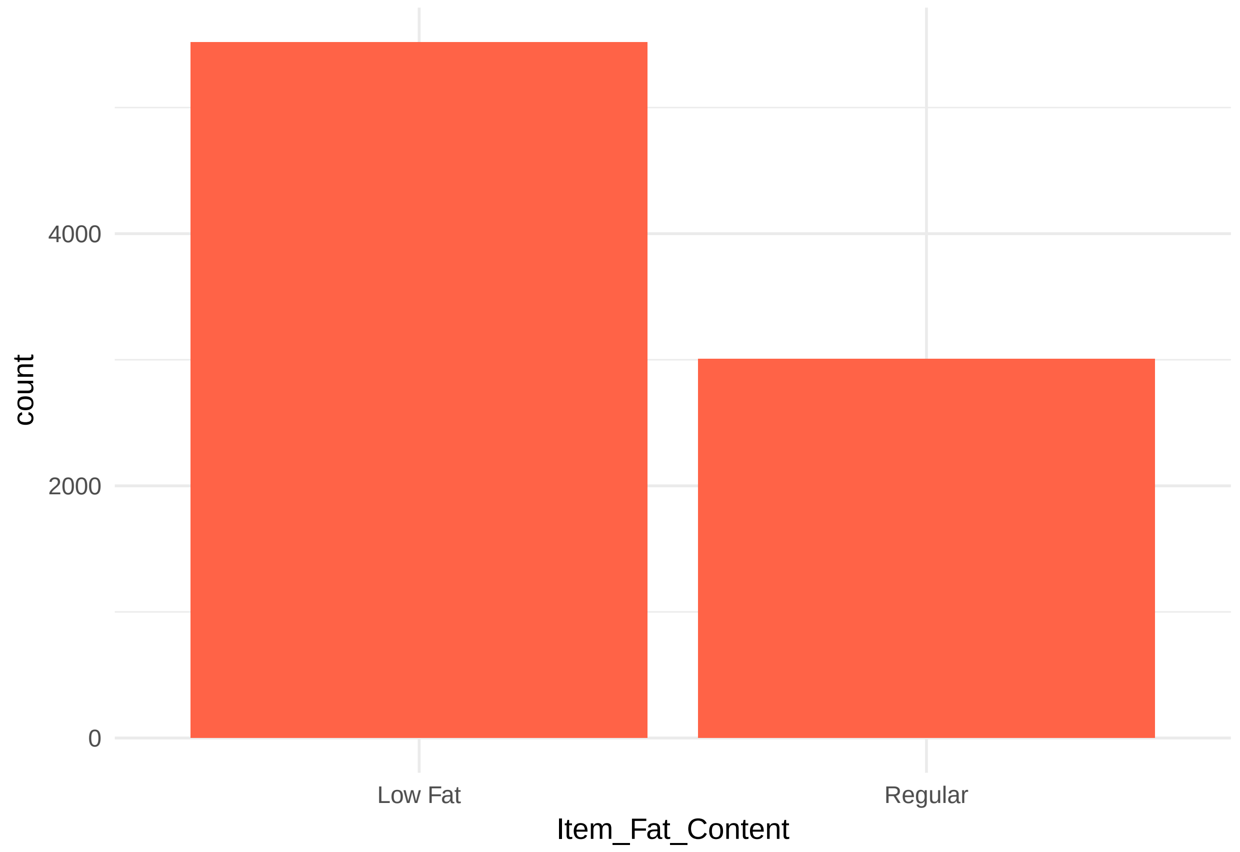
- We should use train data for visualizing because test data has no Item_Outlet_Sales column properly.

```
#Split combined data into train and test data again to examine clearly.
new_train <- combi %>%
  filter(Item_Outlet_Sales != -999)

new_test <- combi %>%
  filter(Item_Outlet_Sales == -999)
#dim(new_train)
#dim(new_test)
```

3.1. Visualizing Manipulated Data

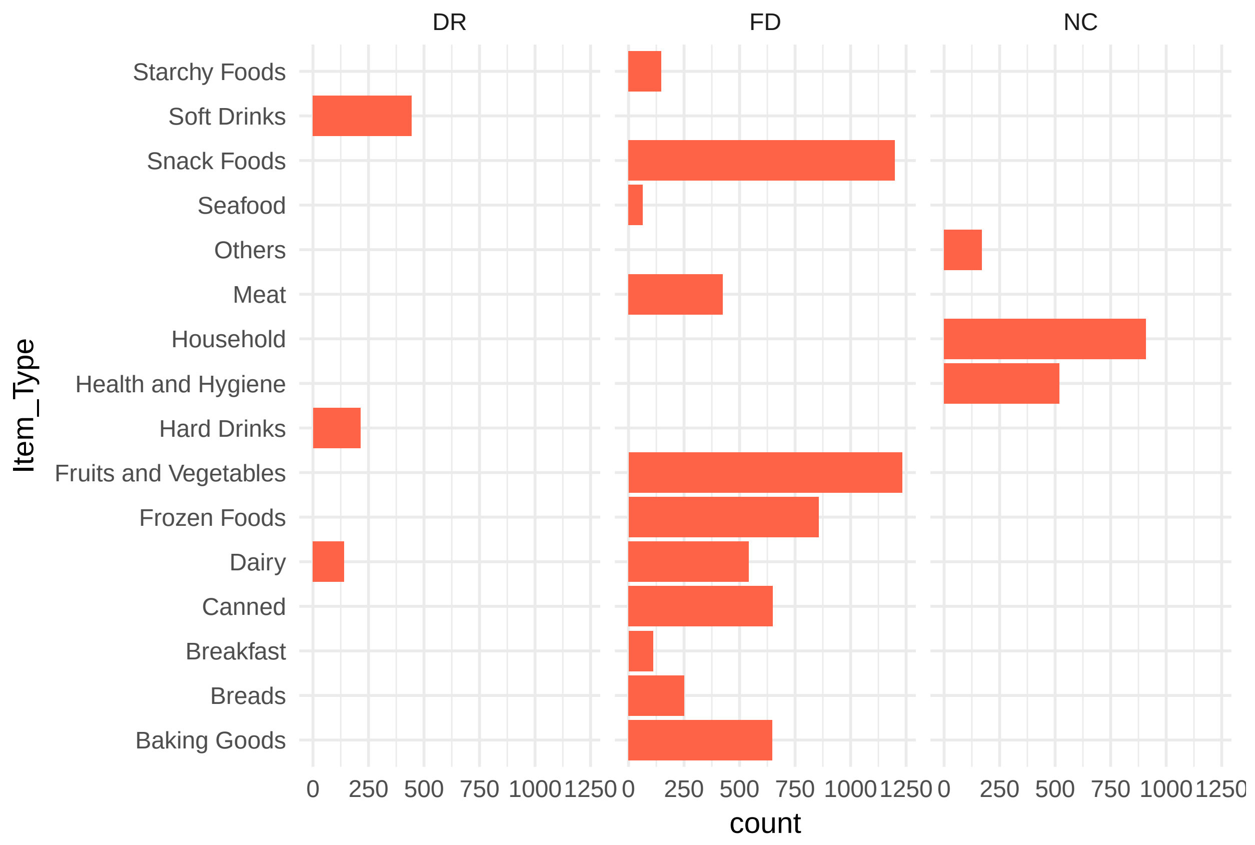
- We manipulated Item_Fat_Content column; so graph it to see new version.

```
# Looking at new Item_Fat_Content column.
qplot(x=Item_Fat_Content,data=new_train) +
  geom_bar(fill="tomato") +
  theme_minimal()
```



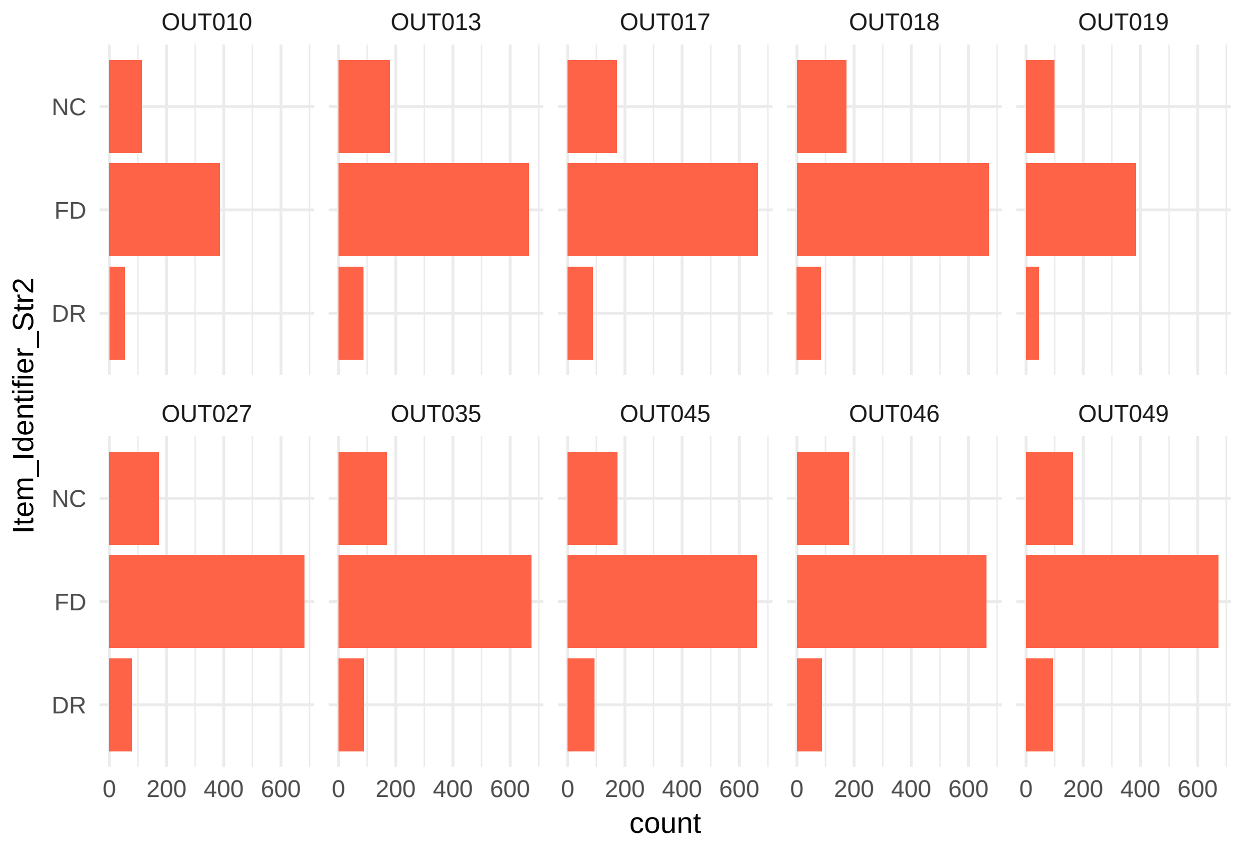
- Looking at Item type and Item Identifier because we observed there is a correlation between these two columns and we manipulated Item_Identifier_Content column as Item_Identifier_Str2 according to Item_Type data. (For example; if Item_Type = “Soft Drinks”, then Item_Identifier_Content starts with ‘DR’ .) As you can see on graph; manipulated Item_Identifier_Str2 column is correct and clear now. So output graph shows; DR (Drink) Idenfier has 3 types of item contents while FD(Food) Identifier has 11 types and NC (Non-Consumable) Identifier has 3 types of item contents. Additionally, we can see “Dairy” type is in both DR and FD. Furthermore; “Soft Drinks” consists the most number of items in “DR” Identifier, “Fruit and Vegetables” in “FD” Indetifier and “Household” in “NC” Identifier.

```r
# Looking at Item Type with facet wrap according to Item_Identifier_Str
qplot(x=Item_Type,data=new_train)+
  geom_bar(fill="tomato")+
  theme(axis.text = element_text(color="tomato"))+
  coord_flip() +
  theme_minimal() +
  facet_wrap(~Item_Identifier_Str2,nrow=1)
```



- Observing the relationship between Item Type/Item Identifier and Outlet_Identifier column to check Item distribution is similar for each Outlet. As we can see, the distribution of items on outlets is very similar.

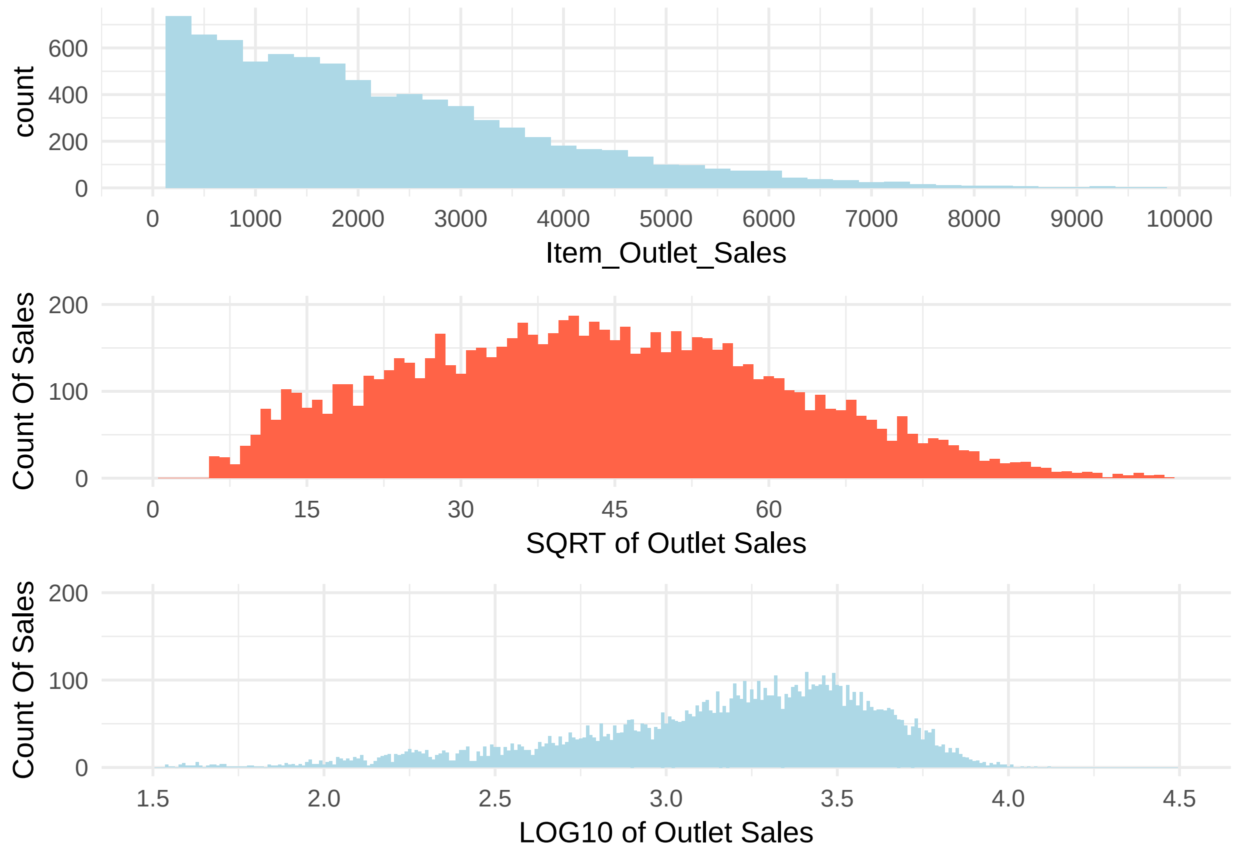
```r
# Looking at Item Type with facet wrap according to Outlet Identifier.

qplot(x = Item_Identifier_Str2, data = new_train) +
  geom_bar(fill = "tomato") +
  theme(axis.text = element_text(angle = 0)) +
  coord_flip() +
  theme_minimal() +
  facet_wrap(~Outlet_Identifier, nrow = 2)
```



3.2. Visualizing Relationship between Item__Outlet__Sales and Other Categorical Columns

- Observing the histogram of Item Outlet Sales for looking of sales distribution. Also, the histogram of Item Outlet Sales with sqrt and log function are drawn, so we can decide which one is more close to normal distribution. As conclusion, we can say SQRT of Item Outlet Sales is much more normal than the others.

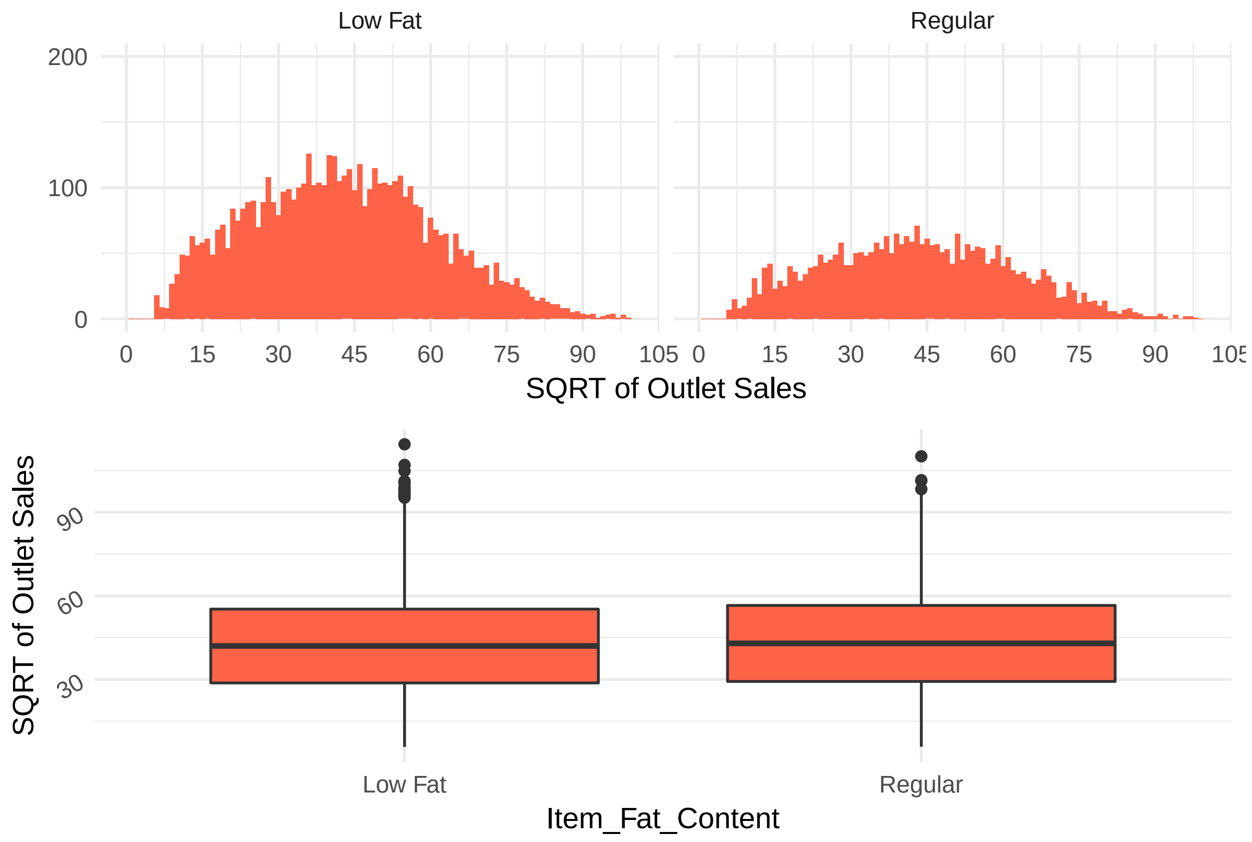
```r
library(gridExtra)

# Looking at Item_Outlet_Sales
p0 <- qplot(x = Item_Outlet_Sales, data = new_train, binwidth = 250,fill=I("lightblue")) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
  theme_minimal() +
  scale_x_continuous(limits = c(0, 10000), breaks = seq(0, 10000, 1000))

# It is better now like normal distibution with sqrt and log10
p1 <- qplot(x = sqrt(Item_Outlet_Sales), data = new_train, binwidth = 1,
      ylab = "Count Of Sales",
      xlab = "SQRT of Outlet Sales",fill=I("tomato")) +
  theme(axis.text.x = element_text(angle = 90),
        axis.text.y = element_text(angle = 30)) +
  theme_minimal() +
  scale_x_continuous(limits = c(0,100), breaks = seq(0,70,15)) +
  scale_y_continuous(limits = c(0,200), breaks = seq(0,200,100))

p2 <- qplot(x = log10(Item_Outlet_Sales+1), data = new_train, binwidth = 0.01,
      ylab = "Count Of Sales",
      xlab = "LOG10 of Outlet Sales",fill=I("lightblue")) +
  theme(axis.text.x = element_text(angle = 90),
        axis.text.y = element_text(angle = 30)) +
  theme_minimal() +
  scale_x_continuous(limits = c(1.5,4.5), breaks = seq(1.5, 4.5, 0.5)) +
  scale_y_continuous(limits = c(0,200), breaks = seq(0, 200, 100))

grid.arrange(p0, p1, p2, ncol=1)
```



- We can say SQRT of Item Outlet Sales has normal distribution according to above graphs.
- Here we can see the comparison of distributions according to low and regular fat. The distribution of sqrt of Outlet Sales for regular and low fat close to a normal distribution. However, low fat's distribution is slightly skewed to right. Also, low fat's sales volume is considerably higher than regular items.

```
# "Sales Distribution trough the Item Fat Content"
p0 <- qplot(x = sqrt(Item_Outlet_Sales), data = new_train, binwidth = 1,
      ylab = "",
      xlab = "SQRT of Outlet Sales",
      fill=I("tomato")) +
  theme(axis.text.x = element_text(angle = 90),
        axis.text.y = element_text(angle = 30)) +
  scale_x_continuous(limits = c(0,100), breaks = seq(0,120,15)) +
  scale_y_continuous(limits = c(0,200), breaks = seq(0,200,100)) +
  theme_minimal() +
  facet_wrap(~Item_Fat_Content)
```

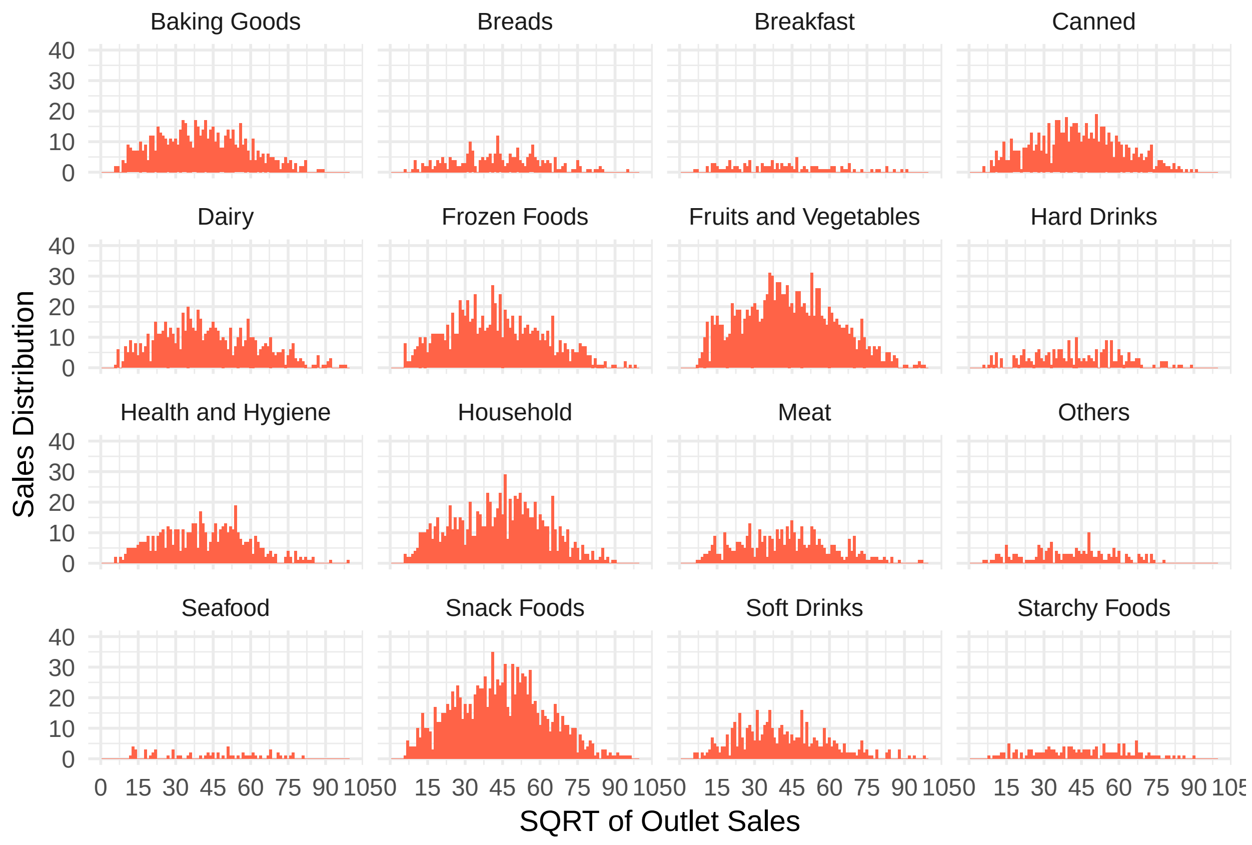
```
#boxplot
p1 <- qplot(x = Item_Fat_Content, y = sqrt(Item_Outlet_Sales),
      ylab = "SQRT of Outlet Sales",
      data = new_train,
      geom = "boxplot",
      fill=I("tomato")) +
      theme_minimal() +
      theme(axis.text.x = element_text(angle = 0),
            axis.text.y = element_text(angle = 30))

grid.arrange(p0, p1, ncol=1)
```



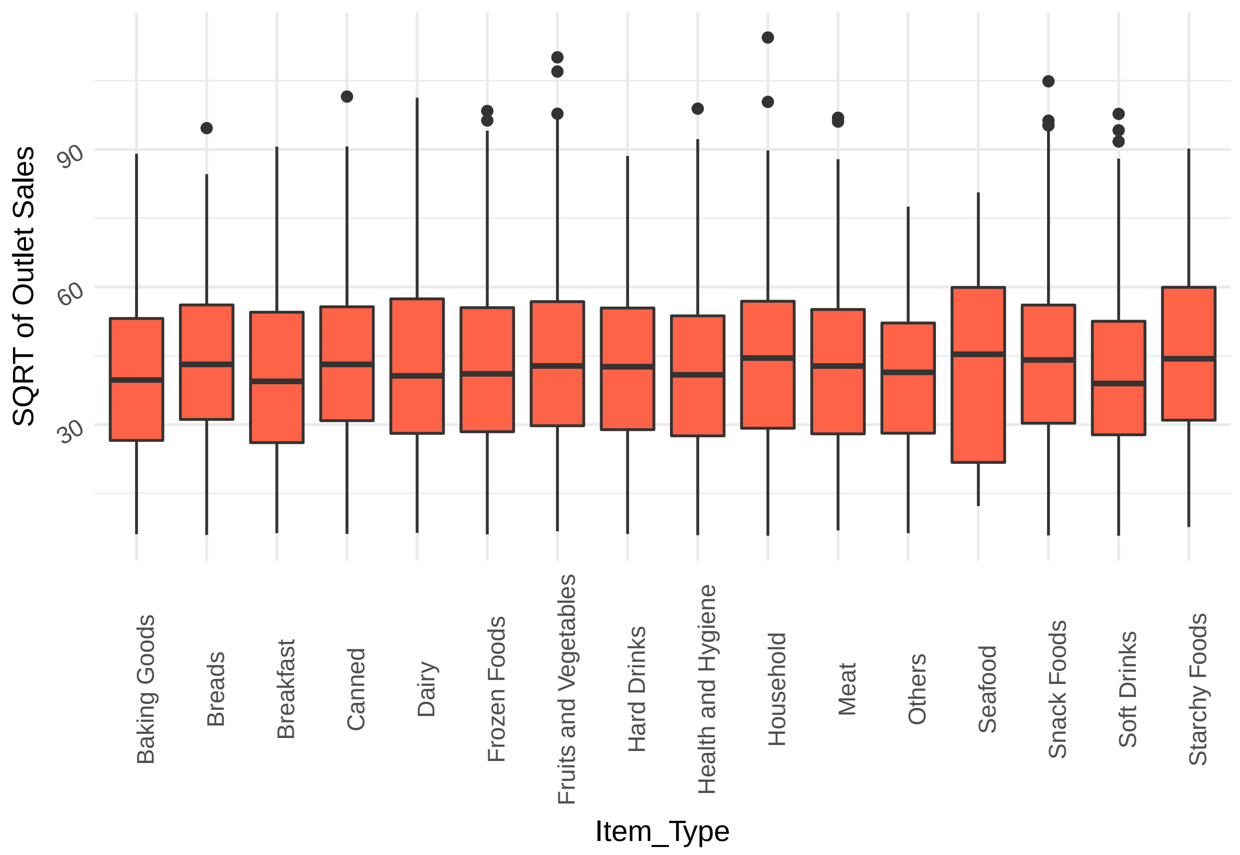
- We take the sqrt of item outles sales created box plots for each item type to show the relationship. The variation of Snack Foods sales is significantly higher than other types. We assume that since Seafood has more types of items varying from luxury to cost-effective compared to other item types.

```
# It is better now like normal distibution with sqrt and log10
p0 <- qplot(x = sqrt(Item_Outlet_Sales), data = new_train, binwidth = 1,
      ylab = "Sales Distribution",
      xlab = "SQRT of Outlet Sales",
      fill=I("tomato")) +
  theme(axis.text.x = element_text(angle = 90),
        axis.text.y = element_text(angle = 30)) +
  theme_minimal() +
  scale_x_continuous(limits = c(0,100), breaks = seq(0,120,15)) +
  scale_y_continuous(limits = c(0,40), breaks = seq(0,40,10)) +
  facet_wrap(~Item_Type)
```

```r
p1 <- qplot(x = Item_Type, y = sqrt(Item_Outlet_Sales),
      ylab = "SQRT of Outlet Sales",
      data = new_train,
      geom = "boxplot",
      fill=I("tomato")) +
      theme_minimal() +
      theme(axis.text.x = element_text(angle = 90),
            axis.text.y = element_text(angle = 30))

p0
```

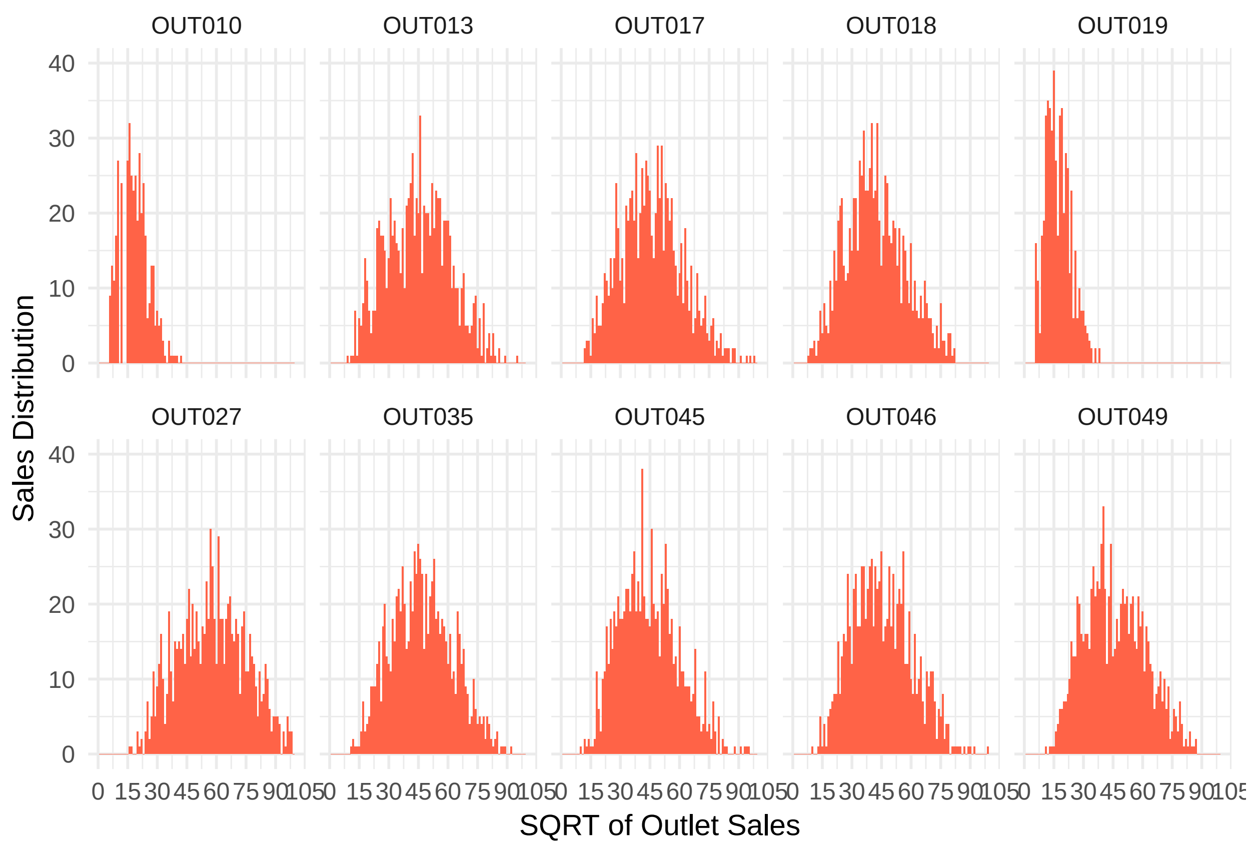


```r
p1
```

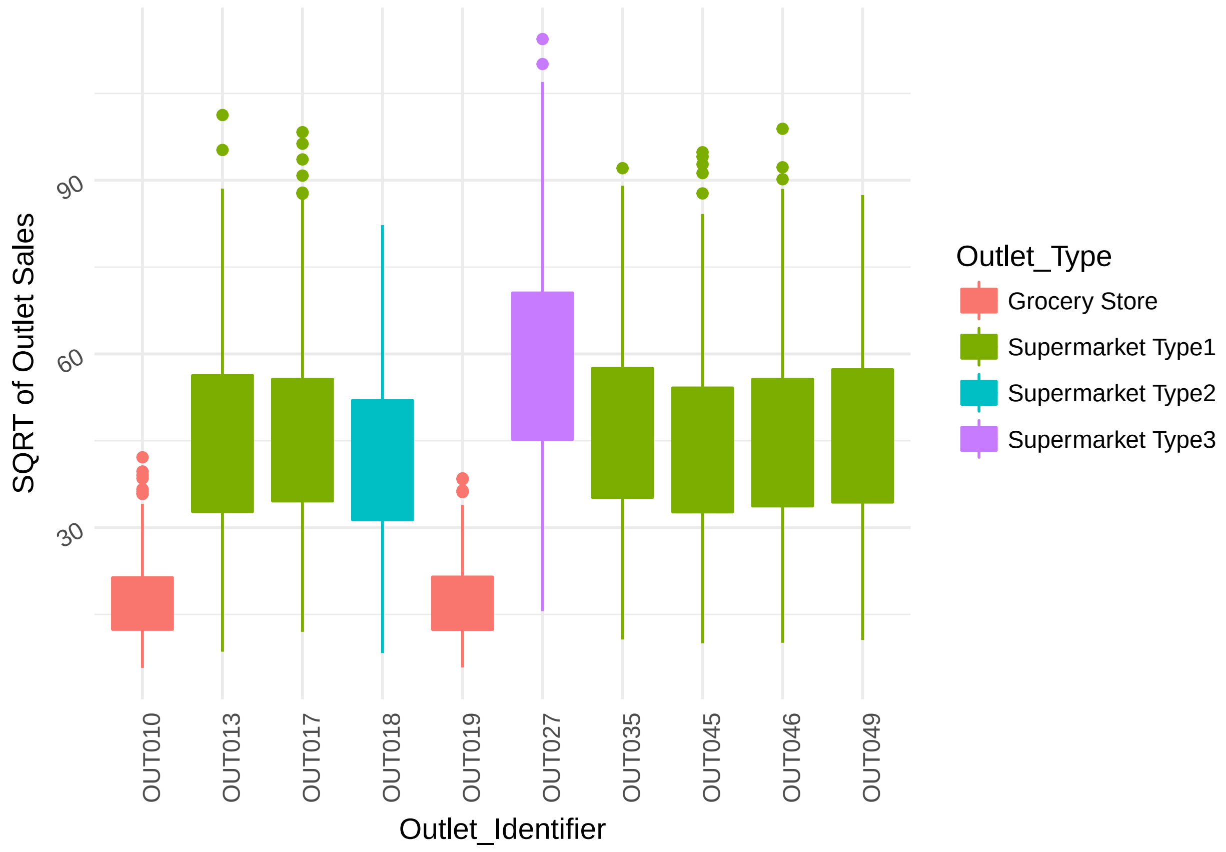


- We investigated the relationship between Outlet Sales and Outlet Identifier by plotting a box plot. It is clearly noticeable that the distributions are grouped according to Outlet Types. The means of outlet types lying in the same group are similar.

```r
# It is better now like normal distibution with sqrt and log10
p0 <- qplot(x = sqrt(Item_Outlet_Sales), data = new_train, binwidth = 1,
      ylab = "Sales Distribution",
      xlab = "SQRT of Outlet Sales",
      fill=I("tomato")) +
  theme(axis.text.x = element_text(angle = 90),
        axis.text.y = element_text(angle = 30)) +
  theme_minimal() +
  scale_x_continuous(limits = c(0,100), breaks = seq(0,120,15)) +
  scale_y_continuous(limits = c(0,40), breaks = seq(0,40,10)) +
  facet_wrap(~Outlet_Identifier,2)

p1 <- qplot(x = Outlet_Identifier, y = sqrt(Item_Outlet_Sales),
      color = Outlet_Type,
      ylab = "SQRT of Outlet Sales",
      data = new_train,
      geom = "boxplot",
      fill=Outlet_Type) +
      #geom_vline(aes(xintercept=quantile(new_train$Item_Outlet_Sales, c(.01))),  color="red", linetype
      theme_minimal() +
      theme(axis.text.x = element_text(angle = 90),
            axis.text.y = element_text(angle = 30))

#library(cowplot)
#cowplot::plot_grid(p0, p1 ,labels = c("A", "B") ,ncol = 2)
p0
```

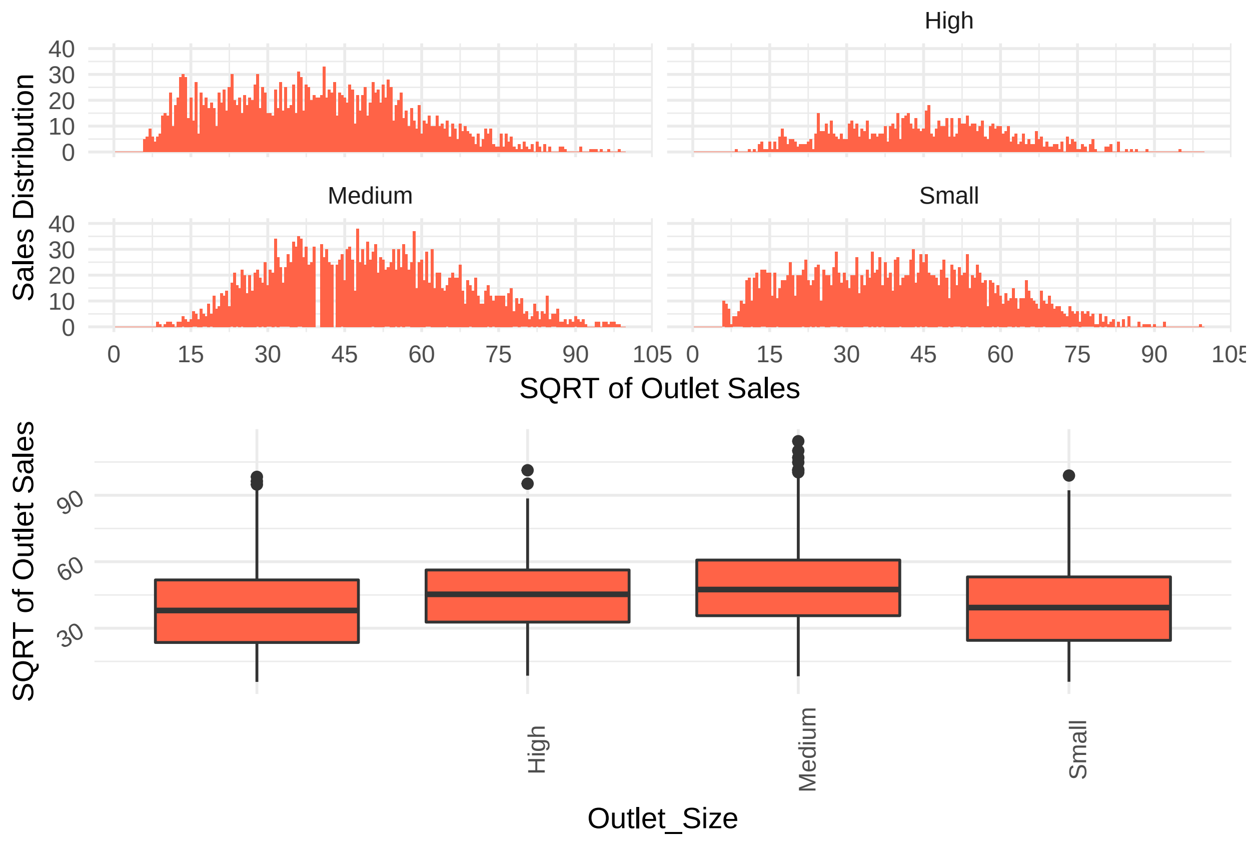


```
p1
```

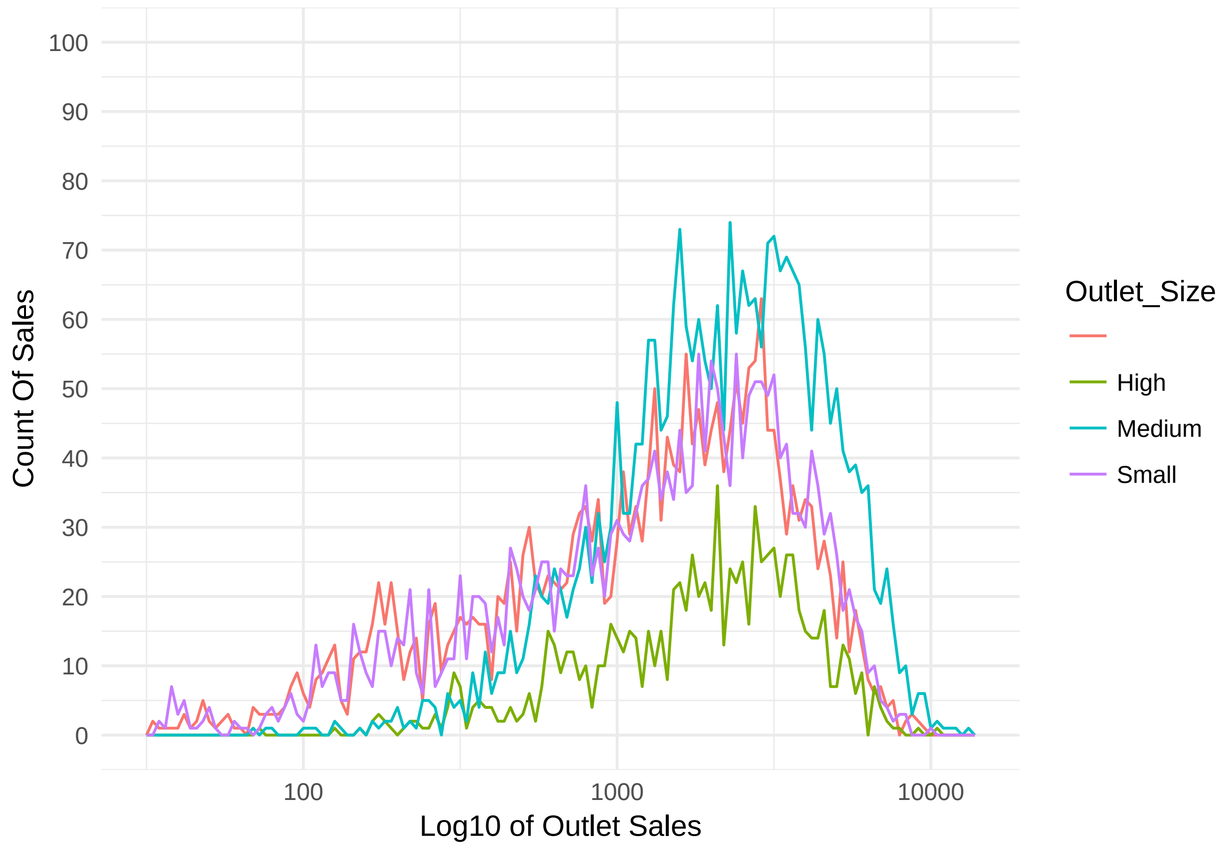


- This graphs show the releationship between Item Outlet Sales and Outlet Size. The items are more frequently bought as outlet size grows.

```
# It is better now like normal distibution with sqrt and log10
p0 <- qplot(x = sqrt(Item_Outlet_Sales), data = new_train, binwidth = 0.5,
      ylab = "Sales Distribution",
      xlab = "SQRT of Outlet Sales",
      fill=I("tomato")) +
  theme(axis.text.x = element_text(angle = 90),
        axis.text.y = element_text(angle = 30)) +
  theme_minimal() +
  scale_x_continuous(limits = c(0,100), breaks = seq(0,120,15)) +
  scale_y_continuous(limits = c(0,40), breaks = seq(0,40,10)) +
  facet_wrap(~Outlet_Size)

p1 <- qplot(x = Outlet_Size, y = sqrt(Item_Outlet_Sales),
      ylab = "SQRT of Outlet Sales",
      data = new_train,
      geom = "boxplot",
      fill=I("tomato")) +
      theme_minimal() +
      theme(axis.text.x = element_text(angle = 90),
            axis.text.y = element_text(angle = 30))

p2 <- qplot(x = Item_Outlet_Sales, data = new_train, binwidth = 0.02,
      ylab = "Count Of Sales",
      xlab = "Log10 of Outlet Sales",
      geom = "freqpoly",
      color = Outlet_Size) +
  theme(axis.text.x = element_text(angle = 90, color = "tomato"),
        axis.text.y = element_text(angle = 30, color = "tomato")) +
        theme_minimal() +
  scale_x_continuous(limits = c(1.5,4.5), breaks = seq(1.5,4.5,0.5)) +
  scale_y_continuous(limits = c(0,100), breaks = seq(0,100,10)) +
  scale_x_log10()


grid.arrange(p0, p1, ncol=1)
```

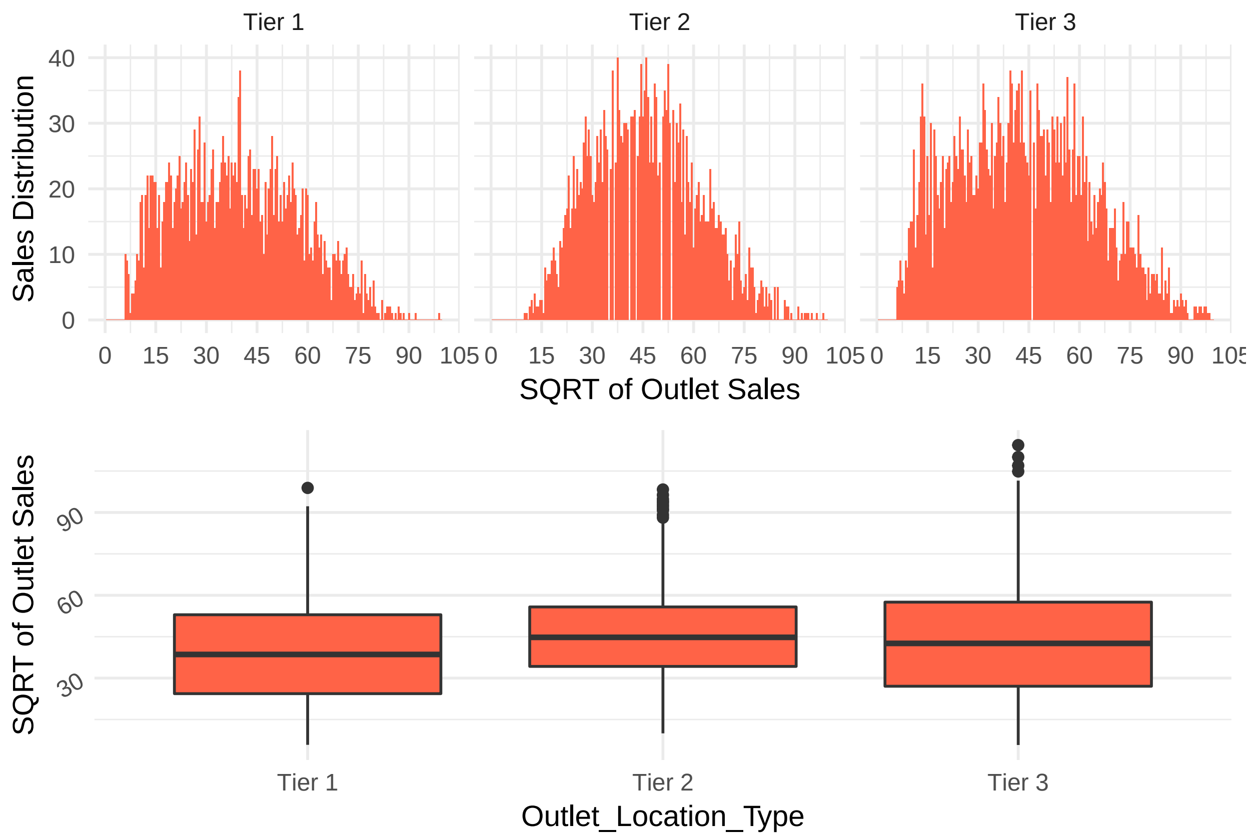


```
p2
```

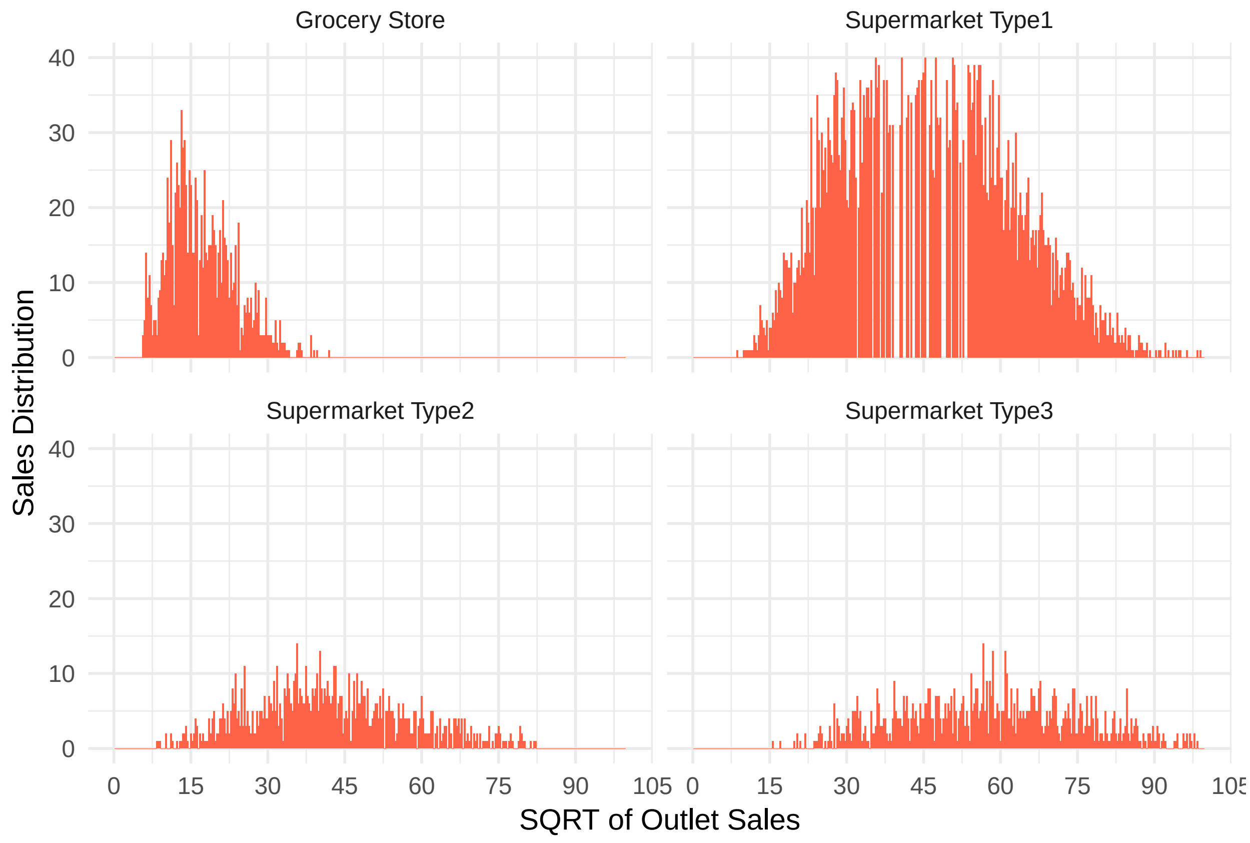


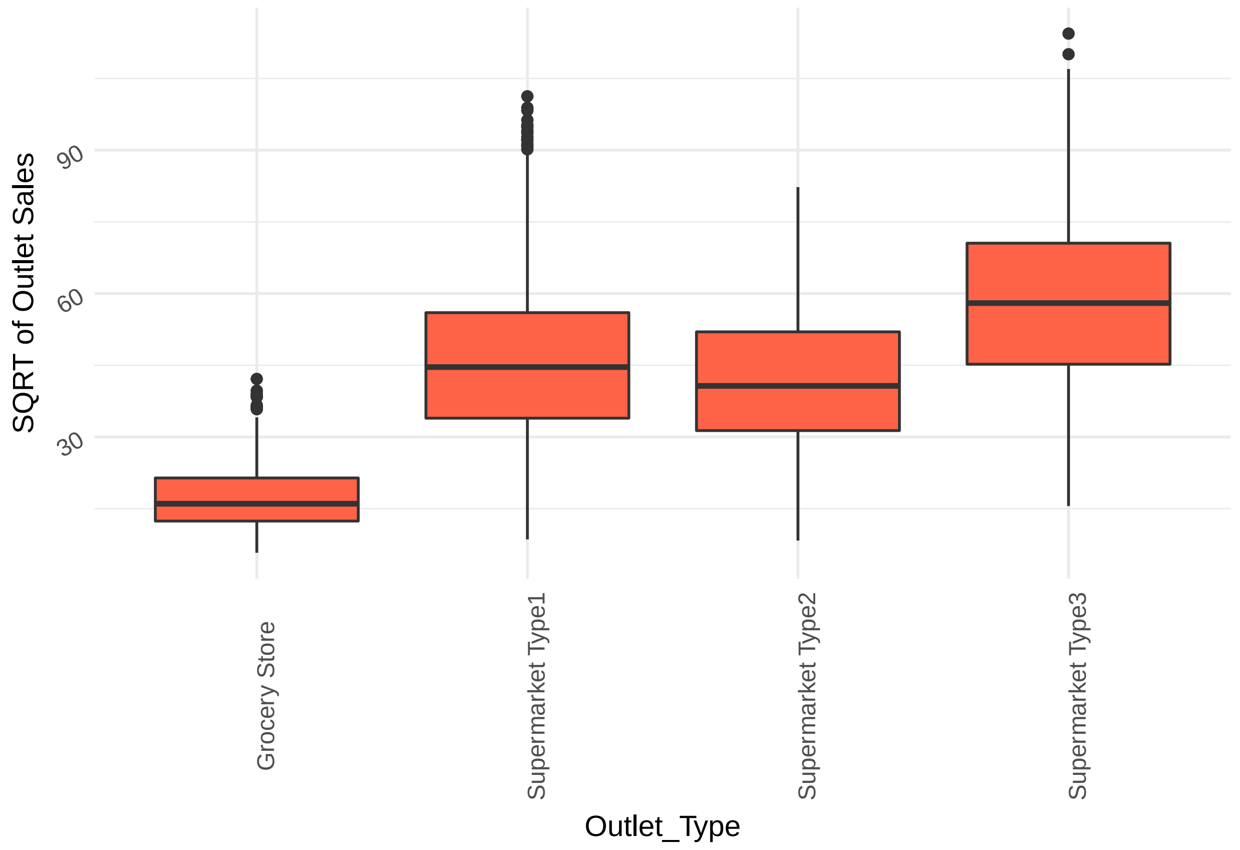
* In order to investigate the relationship between Outlet's locations and Sales, we plotted histograms and box plots for each Tier. Tier 2's sales is distinguishedly higher than the other two. So, the location of the outlet might be an important factor on affecting sales.

```
# It is better now like normal distibution with sqrt and log10
p0 <- qplot(x = sqrt(Item_Outlet_Sales), data = new_train, binwidth = 0.5,
      ylab = "Sales Distribution",
      xlab = "SQRT of Outlet Sales",
      fill=I("tomato")) +
  theme(axis.text.x = element_text(angle = 90),
        axis.text.y = element_text(angle = 30)) +
  theme_minimal() +
  scale_x_continuous(limits = c(0,100), breaks = seq(0,120,15)) +
  scale_y_continuous(limits = c(0,40), breaks = seq(0,40,10)) +
  facet_wrap(~Outlet_Location_Type)

p1 <- qplot(x = Outlet_Location_Type, y = sqrt(Item_Outlet_Sales),
      ylab = "SQRT of Outlet Sales",
      data = new_train,
      geom = "boxplot",
      fill=I("tomato")) +
      theme_minimal() +
      theme(axis.text.x = element_text(angle = 0),
            axis.text.y = element_text(angle = 30))

grid.arrange(p0, p1, ncol=1)
```



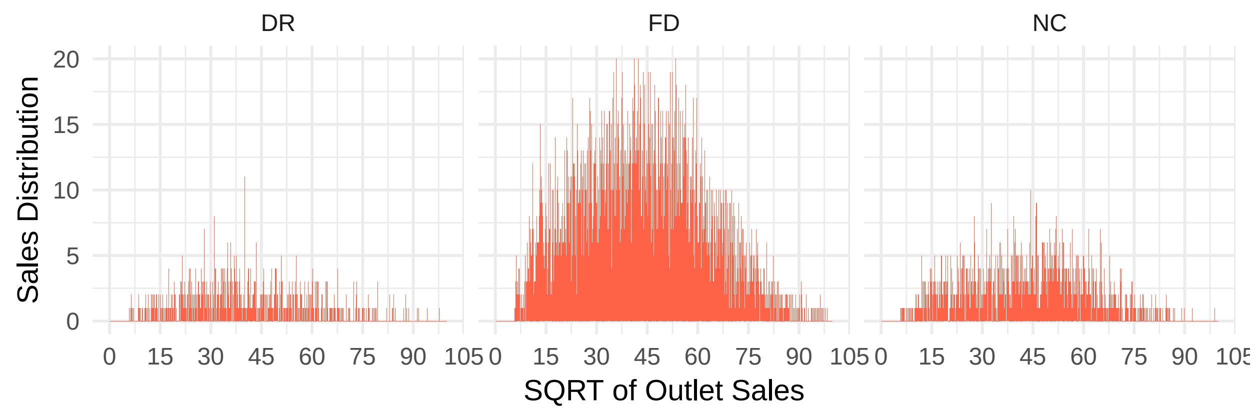
Tier 1
Tier 2
Tier 3
Sales Distribution
0
10
20
30
40
0
15
30
45
60
75
90
105
SQRT of Outlet Sales
SQRT of Outlet Sales
30
60
90
Tier 1
Tier 2
Tier 3
Outlet_Location_Type

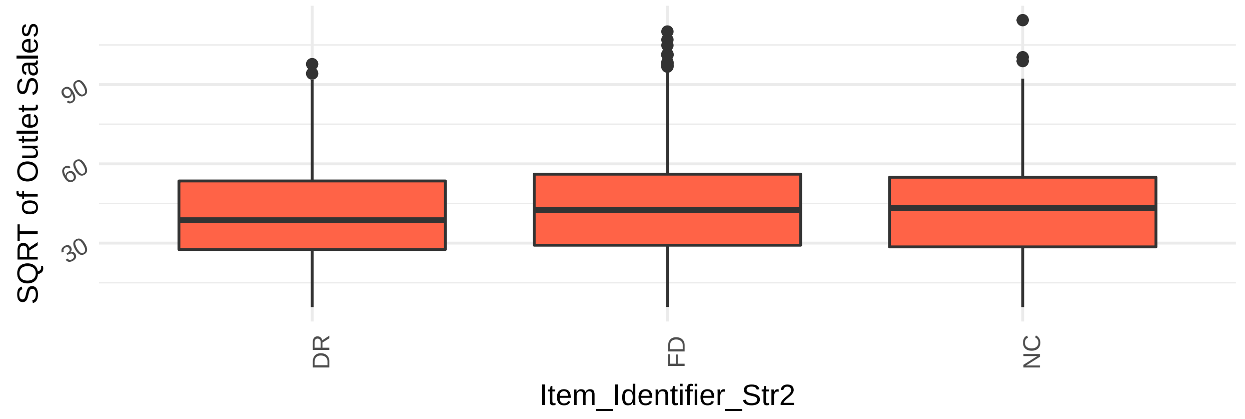

Grocery Store
Supermarket Type1
Supermarket Type2
Supermarket Type3
Sales Distribution
SQRT of Outlet Sales
0
10
20
30
40
0
15
30
45
60
75
90
105



- We compared the sales distributions of each unique identifier. DR stands for Drinkables, FD stands for Food and NC stands for Non Consumable. The frequency of sales for FD is significantly higher than DR and NC. However, the means and percentiles of the all identifier are similar to each other. The reason behind this might be the amount of items of FD is much higher than DR and NC.

```
# It is better now like normal distibution with sqrt and log10
p0 <- qplot(x = sqrt(Item_Outlet_Sales), data = new_train, binwidth = 0.1,
      ylab = "Sales Distribution",
      xlab = "SQRT of Outlet Sales",
      fill=I("tomato")) +
  theme(axis.text.x = element_text(angle = 90),
        axis.text.y = element_text(angle = 30)) +
  theme_minimal() +
  scale_x_continuous(limits = c(0,100), breaks = seq(0,120,15)) +
  scale_y_continuous(limits = c(0,20), breaks = seq(0,20,5)) +
  facet_wrap(~Item_Identifier_Str2)

p1 <- qplot(x = Item_Identifier_Str2, y = sqrt(Item_Outlet_Sales),
      ylab = "SQRT of Outlet Sales",
      data = new_train,
      geom = "boxplot",
      fill=I("tomato")) +
      theme_minimal() +
      theme(axis.text.x = element_text(angle = 90),
            axis.text.y = element_text(angle = 30))

grid.arrange(p0, p1, ncol=1)
```





3.3. Visualizing Relationship between Item__Outlet__Sales and Other Numerical Columns

- Firstly, all categorical columns convert into numerical columns because looking at the correlation between numeric columns and Item Outlet Sales.

Bulk data is like that:

```
new_combi <- rbind(new_train, new_test)

glimpse(new_combi)
```

```
## Observations: 14,204
## Variables: 17
## $ Item_Identifier           <fctr> FDA15, DRC01, FDN15, FDX07, NCD19, ...
## $ Item_Weight               <dbl> 9.300, 5.920, 17.500, 19.200, 8.930,...
## $ Item_Fat_Content          <chr> "Low Fat", "Regular", "Low Fat", "Re...
## $ Item_Visibility           <dbl> 0.016047301, 0.019278216, 0.01676007...
## $ Item_Type                 <fctr> Dairy, Soft Drinks, Meat, Fruits an...
## $ Item_MRP                  <dbl> 249.8092, 48.2692, 141.6180, 182.095...
## $ Outlet_Identifier         <fctr> OUT049, OUT018, OUT049, OUT010, OUT...
## $ Outlet_Establishment_Year <int> 1999, 2009, 1999, 1998, 1987, 2009, ...
## $ Outlet_Size               <fctr> Medium, Medium, Medium, , High, Med...
## $ Outlet_Location_Type      <fctr> Tier 1, Tier 3, Tier 1, Tier 3, Tie...
## $ Outlet_Type               <fctr> Supermarket Type1, Supermarket Type...
## $ Item_Outlet_Sales         <dbl> 3735.1380, 443.4228, 2097.2700, 732....
## $ Item_Identifier_Str3      <chr> "FDA", "DRC", "FDN", "FDX", "NCD", "...
## $ Item_Identifier_Str2      <chr> "FD", "DR", "FD", "FD", "NC", "FD", ...
## $ Item_Identifier_Num       <dbl> 15, 1, 15, 7, 19, 36, 10, 10, 17, 28...
## $ Outlet_Age                <dbl> 14, 4, 14, 15, 26, 4, 26, 28, 11, 6,...
## $ PK                        <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 1...
```

- First we change our categorical data to numerical(1,0) data by spread columns.

```
new_combi <- rbind(new_train, new_test)

new_combi$Item_Fat_Content <- ifelse(combi$Item_Fat_Content == "Regular",1,0)

library(dummies)

new_combi <- dummy.data.frame(new_combi, names = c('Outlet_Size','Outlet_Location_Type'
                                        ,'Outlet_Type','Item_Identifier_Str2'),sep = '_')

glimpse(new_combi)
```

```
## Observations: 14,204
## Variables: 27
## $ Item_Identifier                   <fctr> FDA15, DRC01, FDN15, FDX07, N...
## $ Item_Weight                       <dbl> 9.300, 5.920, 17.500, 19.200, ...
## $ Item_Fat_Content                  <dbl> 0, 1, 0, 1, 0, 1, 1, 0, 1, 1, ...
## $ Item_Visibility                   <dbl> 0.016047301, 0.019278216, 0.01...
## $ Item_Type                         <fctr> Dairy, Soft Drinks, Meat, Fru...
## $ Item_MRP                          <dbl> 249.8092, 48.2692, 141.6180, 1...
## $ Outlet_Identifier                 <fctr> OUT049, OUT018, OUT049, OUT01...
## $ Outlet_Establishment_Year         <int> 1999, 2009, 1999, 1998, 1987, ...
## $ Outlet_Size_                      <int> 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, ...
## $ Outlet_Size_High                  <int> 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, ...
## $ Outlet_Size_Medium                <int> 1, 1, 1, 0, 0, 1, 0, 1, 0, 0, ...
## $ Outlet_Size_Small                 <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
## $ `Outlet_Location_Type_Tier 1`     <int> 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, ...
## $ `Outlet_Location_Type_Tier 2`     <int> 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, ...
## $ `Outlet_Location_Type_Tier 3`     <int> 0, 1, 0, 1, 1, 1, 1, 1, 0, 0, ...
## $ `Outlet_Type_Grocery Store`       <int> 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, ...
## $ `Outlet_Type_Supermarket Type1`   <int> 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, ...
## $ `Outlet_Type_Supermarket Type2`   <int> 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, ...
## $ `Outlet_Type_Supermarket Type3`   <int> 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, ...
## $ Item_Outlet_Sales                 <dbl> 3735.1380, 443.4228, 2097.2700...
## $ Item_Identifier_Str3              <chr> "FDA", "DRC", "FDN", "FDX", "N...
## $ Item_Identifier_Str2_DR           <int> 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, ...
## $ Item_Identifier_Str2_FD           <int> 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, ...
## $ Item_Identifier_Str2_NC           <int> 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, ...
## $ Item_Identifier_Num               <dbl> 15, 1, 15, 7, 19, 36, 10, 10, ...
## $ Outlet_Age                        <dbl> 14, 4, 14, 15, 26, 4, 26, 28, ...
## $ PK                                <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10,...
```

- We delete our character columns before correlation.

```
new_combi <- select(new_combi, -c(Item_Identifier, Outlet_Identifier, Item_Fat_Content,Outlet_Establishm

str(new_combi)
```

```
## 'data.frame':    14204 obs. of  20 variables:
```

```
##  $ Item_Weight                   : num  9.3 5.92 17.5 19.2 8.93 ...
##  $ Item_Visibility               : num  0.016 0.0193 0.0168 0.054 0.054 ...
##  $ Item_MRP                      : num  249.8 48.3 141.6 182.1 53.9 ...
##  $ Outlet_Size_                  : int  0 0 0 1 0 0 0 0 1 1 ...
##  $ Outlet_Size_High              : int  0 0 0 0 1 0 1 0 0 0 ...
##  $ Outlet_Size_Medium            : int  1 1 1 0 0 1 0 1 0 0 ...
##  $ Outlet_Size_Small             : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Outlet_Location_Type_Tier 1   : int  1 0 1 0 0 0 0 0 0 0 ...
##  $ Outlet_Location_Type_Tier 2   : int  0 0 0 0 0 0 0 0 1 1 ...
##  $ Outlet_Location_Type_Tier 3   : int  0 1 0 1 1 1 1 1 0 0 ...
##  $ Outlet_Type_Grocery Store     : int  0 0 0 1 0 0 0 0 0 0 ...
##  $ Outlet_Type_Supermarket Type1: int  1 0 1 0 1 0 1 0 1 1 ...
##  $ Outlet_Type_Supermarket Type2: int  0 1 0 0 0 1 0 0 0 0 ...
##  $ Outlet_Type_Supermarket Type3: int  0 0 0 0 0 0 0 1 0 0 ...
##  $ Item_Outlet_Sales             : num  3735 443 2097 732 995 ...
##  $ Item_Identifier_Str2_DR       : int  0 1 0 0 0 0 0 0 0 0 ...
##  $ Item_Identifier_Str2_FD       : int  1 0 1 1 0 1 1 1 1 1 ...
##  $ Item_Identifier_Str2_NC       : int  0 0 0 0 1 0 0 0 0 0 ...
##  $ Item_Identifier_Num           : num  15 1 15 7 19 36 10 10 17 28 ...
##  $ Outlet_Age                    : num  14 4 14 15 26 4 26 28 11 6 ...
##  - attr(*, "dummies")=List of 4
##   ..$ Outlet_Size         : int  9 10 11 12
##   ..$ Outlet_Location_Type: int  13 14 15
##   ..$ Outlet_Type         : int  16 17 18 19
##   ..$ Item_Identifier_Str2: int  22 23 24
```

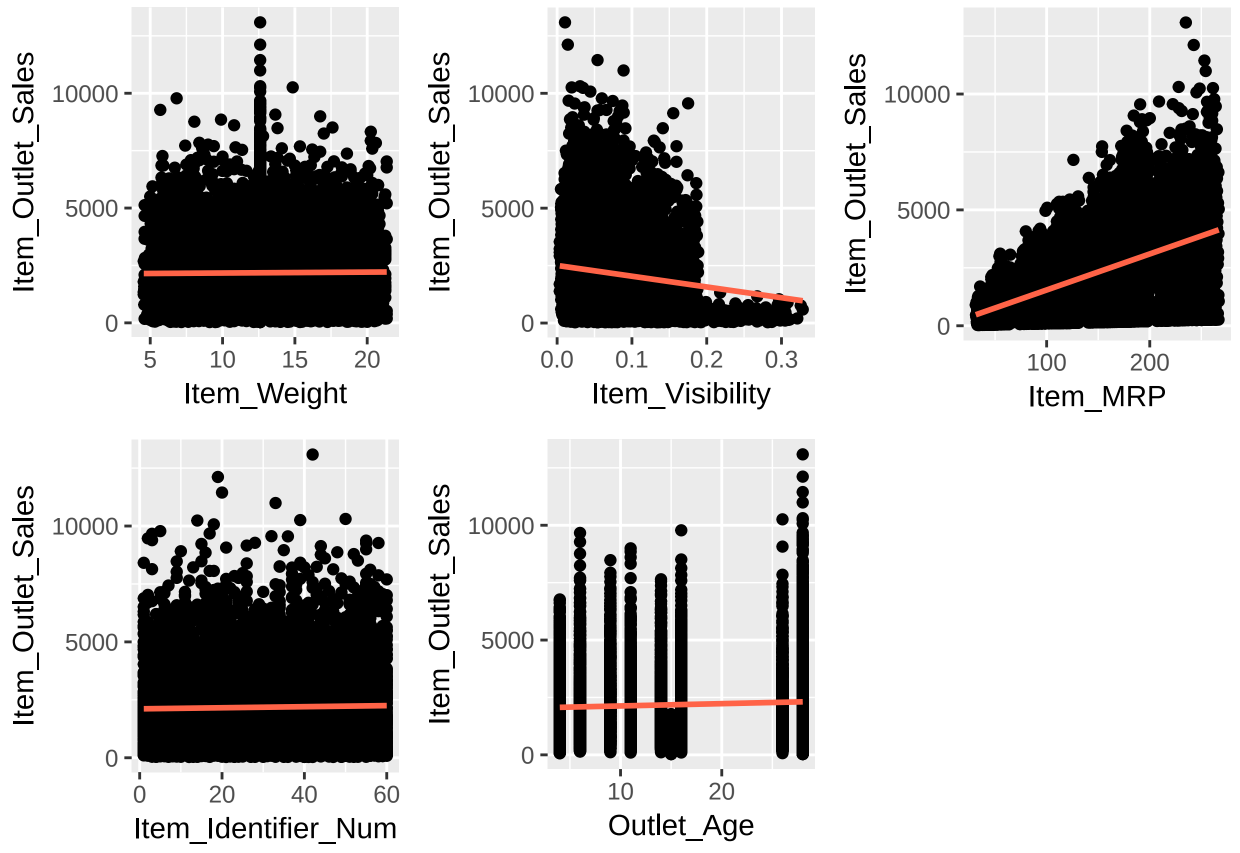
- We divide our test and train data for looking correlation.

```
pred_train <- new_combi %>%
  filter(Item_Outlet_Sales != -999)

pred_test <- new_combi %>%
  filter(Item_Outlet_Sales == -999)
#dim(pred_train)
#dim(pred_test)
```

- Looking correlation between numerical data columns and Item_Outlet_Sales.

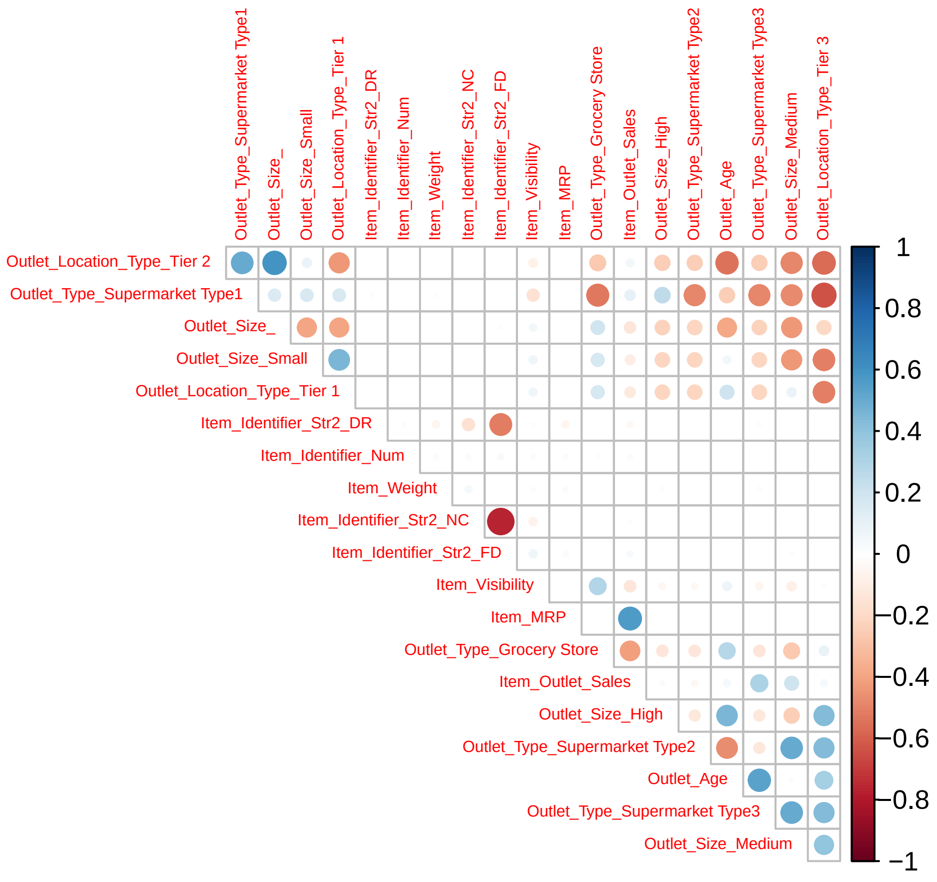
```
p1 <- ggplot(pred_train, aes(x = Item_Weight, y = Item_Outlet_Sales)) +
  geom_point() +
  geom_smooth(method = 'lm', se = FALSE, color='tomato')

p2 <- ggplot(pred_train, aes(x = Item_Visibility, y = Item_Outlet_Sales)) +
  geom_point() +
  geom_smooth(method = 'lm', se = FALSE, color='tomato')

p3 <- ggplot(pred_train, aes(x = Item_MRP, y = Item_Outlet_Sales)) +
  geom_point() +
  geom_smooth(method = 'lm', se = FALSE, color='tomato')

p4 <- ggplot(pred_train, aes(x = Item_Identifier_Num, y = Item_Outlet_Sales)) +
  geom_point() +
  geom_smooth(method = 'lm', se = FALSE, color='tomato')

p5 <- ggplot(pred_train, aes(x = Outlet_Age, y = Item_Outlet_Sales)) +
  geom_point() +
```

```r
  geom_smooth(method = 'lm', se = FALSE, color='tomato')

grid.arrange(p1, p2, p3, p4, p5, ncol=3)
```



- Look at correlation matrix. (https://cran.r-project.org).

```r
library("corrplot")
library(RColorBrewer)

M<-cor(pred_train)


corrplot(M, diag = FALSE, order = "FPC",
         tl.pos = "td", tl.cex = 0.5, method = "circle",type="upper")
```



4. Modeling

4.1. Discover Models

Multiple Regression is used when response variable is continuous in nature and predictors are many. Had it been categorical, we would have used Logistic Regression.

- Model1: Linear Regression for all dataset which is bulk.

```
new_train <- select(new_train, -c(Item_Identifier, Item_Identifier_Str3, Outlet_Establishment_Year, PK)

set.seed(1)
n <- nrow(new_train)
shuffled <- new_train[sample(n),]
#glimpse(new_train)

#split train data again:
train_indices <- 1:round(0.7*n)
test_indices <-  (round(0.7*n)+1):n

splitted_train_simple <- shuffled[train_indices,]
splitted_test_simple <- shuffled[test_indices,]


# build model with train data: (70% of actual data)
linear_model_simple <- lm(Item_Outlet_Sales ~ ., data = splitted_train_simple)
```

```r
#summary(linear_model_simple)
linear_model_log1<- lm(log10(Item_Outlet_Sales) ~ ., data = splitted_train_simple)
#summary(linear_model_log)
linear_model_sqrt <- lm(sqrt(Item_Outlet_Sales) ~ ., data = splitted_train_simple)
#summary(linear_model_sqrt)

#make prediction with test data (%30 of actual train data)
pred_simple_test <- predict(linear_model_simple, splitted_test_simple)
pred_log_test <-10 ^ predict(linear_model_log1, splitted_test_simple)
pred_sqrt_test <-predict(linear_model_sqrt, splitted_test_simple) ^ 2
pred_simple_train <- predict(linear_model_simple, splitted_train_simple)
pred_log_train <-10 ^ predict(linear_model_log1, splitted_train_simple)
pred_sqrt_train <-predict(linear_model_sqrt, splitted_train_simple) ^ 2

#
# MAE Function for test
MAE <- function(actual, predicted){mean(abs(actual - predicted))}
# MAE of Simple for test
model1_mae_simple <- MAE(splitted_test_simple$Item_Outlet_Sales, pred_simple_test)
# MAE of Log for test
model1_mae_log <- (MAE(splitted_test_simple$Item_Outlet_Sales, pred_log_test))
# MAE of Sqrt for test
model1_mae_sqrt <- (MAE(splitted_test_simple$Item_Outlet_Sales, pred_sqrt_test))

# RMSE Function
RMSE <- function(actual, predicted)  {sqrt(mean((actual - predicted)^2))}
# RMSE of Simple for test dataset
model1_rmse_simple_test <-(RMSE(splitted_test_simple$Item_Outlet_Sales, pred_simple_test))
# RMSE of Simple for train dataset
model1_rmse_simple_train <-(RMSE(splitted_train_simple$Item_Outlet_Sales, pred_simple_train))
# RMSE of Simple for train/test
model1_rmse_ratio <- (RMSE(splitted_train_simple$Item_Outlet_Sales, pred_simple_train))/(RMSE(splitted_
# RMSE of Log for test dataset
model1_rmse_log_test <- (RMSE(splitted_test_simple$Item_Outlet_Sales, pred_log_test))
# RMSE of Log for train dataset
model1_rmse_log_train <- (RMSE(splitted_train_simple$Item_Outlet_Sales, pred_log_train))
# RMSE of Log for train/test
model1_rmse_log_ratio <- (RMSE(splitted_train_simple$Item_Outlet_Sales, pred_log_train))/(RMSE(splitted
# RMSE of Sqrt for test dataset
model1_rmse_sqrt_test <- (RMSE(splitted_test_simple$Item_Outlet_Sales, pred_sqrt_test))
# RMSE of Sqrt for train dataset
model1_rmse_sqrt_train <- (RMSE(splitted_train_simple$Item_Outlet_Sales, pred_sqrt_train))
# RMSE of Sqrt for train/test
model1_rmse_sqrt_ratio <-(RMSE(splitted_train_simple$Item_Outlet_Sales, pred_sqrt_train))/(RMSE(splitte
```

- Model2: Linear Regression for all dataset which generated dummy library.

```r
set.seed(1)
n <- nrow(pred_train)
shuffled <- pred_train[sample(n),]
#glimpse(pred_train)



#split train data again:
```

```r
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,]


# build model with train data: (70% of actual data)
linear_model_simple <- lm(Item_Outlet_Sales ~ ., data = splitted_train)
#summary(linear_model_simple)
 linear_model_log2 <- lm(log10(Item_Outlet_Sales) ~ ., data = splitted_train)
#summary(linear_model_log)
 linear_model_sqrt <- lm(sqrt(Item_Outlet_Sales) ~ ., data = splitted_train)
#summary(linear_model_sqrt)

#make prediction with test data (%30 of actual train data)
pred_simple_test <- predict(linear_model_simple, splitted_test)
pred_log_test <-10 ^ predict(linear_model_log2, splitted_test)
pred_sqrt_test <-predict(linear_model_sqrt, splitted_test) ^ 2
pred_simple_train <- predict(linear_model_simple, splitted_train)
pred_log_train <-10 ^ predict(linear_model_log2, splitted_train)
pred_sqrt_train <-predict(linear_model_sqrt, splitted_train) ^ 2

#
# MAE Function for test
(MAE <- function(actual, predicted){mean(abs(actual - predicted))})
```

```
## function(actual, predicted){mean(abs(actual - predicted))}
```

```r
# MAE of Simple for test
MAE(splitted_test$Item_Outlet_Sales, pred_simple_test)
```

```
## [1] 840.0915
```

```r
# MAE of Log for test
(MAE(splitted_test$Item_Outlet_Sales, pred_log_test))
```

```
## [1] 798.8246
```

```r
# MAE of Sqrt for test
(MAE(splitted_test$Item_Outlet_Sales, pred_sqrt_test))
```

```
## [1] 780.975
```

```r
# RMSE Function
(RMSE <- function(actual, predicted)  {sqrt(mean((actual - predicted)^2))})
```

```
## function(actual, predicted)  {sqrt(mean((actual - predicted)^2))}
```

```r
# RMSE of Simple for test dataset
(RMSE(splitted_test$Item_Outlet_Sales, pred_simple_test))
```

```
## [1] 1138.833
```

```r
# RMSE of Simple for train dataset
(RMSE(splitted_train$Item_Outlet_Sales, pred_simple_train))
```

```
## [1] 1121.622
```

```r
# RMSE of Simple for train/test
(RMSE(splitted_train$Item_Outlet_Sales, pred_simple_train))/(RMSE(splitted_test$Item_Outlet_Sales, pred_
```

```
## [1] 0.9848873
```

```r
# RMSE of Log for test dataset
(RMSE(splitted_test$Item_Outlet_Sales, pred_log_test))
```

```
## [1] 1164.727
```

```r
# RMSE of Log for train dataset
(RMSE(splitted_train$Item_Outlet_Sales, pred_log_train))
```

```
## [1] 1127.386
```

```r
# RMSE of Log for train/test
(RMSE(splitted_train$Item_Outlet_Sales, pred_log_train))/(RMSE(splitted_test$Item_Outlet_Sales, pred_log
```

```
## [1] 0.9679399
```

```r
# RMSE of Sqrt for test dataset
(RMSE(splitted_test$Item_Outlet_Sales, pred_sqrt_test))
```

```
## [1] 1116.636
```

```r
# RMSE of Sqrt for train dataset
(RMSE(splitted_train$Item_Outlet_Sales, pred_sqrt_train))
```

```
## [1] 1089.782
```

```r
# RMSE of Sqrt for train/test
(RMSE(splitted_train$Item_Outlet_Sales, pred_sqrt_train))/(RMSE(splitted_test$Item_Outlet_Sales, pred_s
```

```
## [1] 0.9759507
```

```r
summary(linear_model_simple)$coefficients[,4] < 0.05
```

```
##                      (Intercept)                      Item_Weight
##                             TRUE                            FALSE
##                  Item_Visibility                         Item_MRP
##                            FALSE                             TRUE
##                     Outlet_Size_                 Outlet_Size_High
##                             TRUE                            FALSE
##               Outlet_Size_Medium   `Outlet_Location_Type_Tier 1`
##                            FALSE                            FALSE
##   `Outlet_Location_Type_Tier 2`     `Outlet_Type_Grocery Store`
##                            FALSE                             TRUE
## `Outlet_Type_Supermarket Type1` `Outlet_Type_Supermarket Type2`
##                             TRUE                             TRUE
##         Item_Identifier_Str2_DR         Item_Identifier_Str2_FD
##                            FALSE                            FALSE
##              Item_Identifier_Num                       Outlet_Age
##                             TRUE                             TRUE
```

```r
('')
```

```
## [1] ""
```

```r
summary(linear_model_log2)$coefficients[,4] < 0.05
```

```
##                      (Intercept)                      Item_Weight
##                             TRUE                            FALSE
##                  Item_Visibility                         Item_MRP
##                            FALSE                             TRUE
##                     Outlet_Size_                 Outlet_Size_High
##                             TRUE                             TRUE
##               Outlet_Size_Medium   `Outlet_Location_Type_Tier 1`
##                            FALSE                             TRUE
##   `Outlet_Location_Type_Tier 2`     `Outlet_Type_Grocery Store`
##                             TRUE                             TRUE
## `Outlet_Type_Supermarket Type1` `Outlet_Type_Supermarket Type2`
##                             TRUE                             TRUE
##          Item_Identifier_Str2_DR          Item_Identifier_Str2_FD
##                            FALSE                            FALSE
##              Item_Identifier_Num                       Outlet_Age
##                             TRUE                             TRUE
```

```
('')
```

```
## [1] ""
```

```
summary(linear_model_sqrt)$coefficients[,4] < 0.05
```

```
##                      (Intercept)                      Item_Weight
##                             TRUE                            FALSE
##                  Item_Visibility                         Item_MRP
##                            FALSE                             TRUE
##                     Outlet_Size_                 Outlet_Size_High
##                             TRUE                            FALSE
##               Outlet_Size_Medium   `Outlet_Location_Type_Tier 1`
##                            FALSE                            FALSE
##   `Outlet_Location_Type_Tier 2`     `Outlet_Type_Grocery Store`
##                            FALSE                             TRUE
## `Outlet_Type_Supermarket Type1` `Outlet_Type_Supermarket Type2`
##                             TRUE                             TRUE
##          Item_Identifier_Str2_DR          Item_Identifier_Str2_FD
##                            FALSE                            FALSE
##              Item_Identifier_Num                       Outlet_Age
##                             TRUE                             TRUE
```

- Model3: Linear Regression based on summary of regressions(p values<0.05).

```
set.seed(1)

(formula_simp <- as.formula(Item_Outlet_Sales ~ Item_MRP + Outlet_Size_ + `Outlet_Type_Grocery Store` +
```

```
## Item_Outlet_Sales ~ Item_MRP + Outlet_Size_ + `Outlet_Type_Grocery Store` +
##     `Outlet_Type_Supermarket Type1` + `Outlet_Type_Supermarket Type2` +
##     Item_Identifier_Num + Outlet_Age
```

```
(formula_log <- as.formula(log10(Item_Outlet_Sales) ~ Item_MRP + Outlet_Size_ + Outlet_Size_High + `Outl
```

```
## log10(Item_Outlet_Sales) ~ Item_MRP + Outlet_Size_ + Outlet_Size_High +
##     `Outlet_Location_Type_Tier 1` + `Outlet_Location_Type_Tier 2` +
##     `Outlet_Type_Grocery Store` + `Outlet_Type_Supermarket Type1` +
##     `Outlet_Type_Supermarket Type2` + Item_Identifier_Num + Outlet_Age
```

```
(formula_sqrt <- as.formula(sqrt(Item_Outlet_Sales) ~ Item_MRP + Outlet_Size_ + `Outlet_Type_Grocery Sto
## sqrt(Item_Outlet_Sales) ~ Item_MRP + Outlet_Size_ + `Outlet_Type_Grocery Store` +
##     `Outlet_Type_Supermarket Type1` + `Outlet_Type_Supermarket Type2` +
##     Item_Identifier_Num + Outlet_Age
```

```
# build model with train data: (70% of actual data)
linear_model_simple <- lm(formula_simp, data = splitted_train)
#summary(linear_model_simple)
 linear_model_log3 <- lm(formula_log, data = splitted_train)
#summary(linear_model_log)
 linear_model_sqrt <- lm(formula_sqrt, data = splitted_train)
#summary(linear_model_sqrt)

# make prediction with test data (%30 of actual train data)
pred_simple_test <- predict(linear_model_simple, splitted_test)
pred_log_test <-10 ^ predict(linear_model_log3, splitted_test)
pred_sqrt_test <-predict(linear_model_sqrt, splitted_test) ^ 2
pred_simple_train <- predict(linear_model_simple, splitted_train)
pred_log_train <-10 ^ predict(linear_model_log3, splitted_train)
pred_sqrt_train <-predict(linear_model_sqrt, splitted_train) ^ 2


# MAE Function
MAE <- function(actual, predicted){mean(abs(actual - predicted))}
# MAE of Simple for test
MAE(splitted_test$Item_Outlet_Sales, pred_simple_test)
```

```
## [1] 839.5721
```

```
# MAE of Log for test
(MAE(splitted_test$Item_Outlet_Sales, pred_log_test))
```

```
## [1] 799.2971
```

```
# MAE of Sqrt for Test
(MAE(splitted_test$Item_Outlet_Sales, pred_sqrt_test))
```

```
## [1] 781.7254
```

```
# RMSE Function
RMSE <- function(actual, predicted)  {sqrt(mean((actual - predicted)^2))}
# RMSE of Simple for test dataset
(RMSE(splitted_test$Item_Outlet_Sales, pred_simple_test))
```

```
## [1] 1139.64
```

```
# RMSE of Simple for train dataset
(RMSE(splitted_train$Item_Outlet_Sales, pred_simple_train))
```

```
## [1] 1122.395
```

```
# RMSE of Simple for train/test
(RMSE(splitted_train$Item_Outlet_Sales, pred_simple_train))/(RMSE(splitted_test$Item_Outlet_Sales, pred_
```

```
## [1] 0.9848678
```

```
# RMSE of Log for test dataset
(RMSE(splitted_test$Item_Outlet_Sales, pred_log_test))
```

```
## [1] 1165.215
```

```r
# RMSE of Log for train dataset
(RMSE(splitted_train$Item_Outlet_Sales, pred_log_train))
```

```
## [1] 1127.359
```

```r
# RMSE of Log for train/test
(RMSE(splitted_train$Item_Outlet_Sales, pred_log_train))/(RMSE(splitted_test$Item_Outlet_Sales, pred_log
```

```
## [1] 0.9675116
```

```r
# RMSE of Sqrt for test dataset
(RMSE(splitted_test$Item_Outlet_Sales, pred_sqrt_test))
```

```
## [1] 1118.607
```

```r
# RMSE of Sqrt for train dataset
(RMSE(splitted_train$Item_Outlet_Sales, pred_sqrt_train))
```

```
## [1] 1090.689
```

```r
# RMSE of Sqrt for train/test
(RMSE(splitted_train$Item_Outlet_Sales, pred_sqrt_train))/(RMSE(splitted_test$Item_Outlet_Sales, pred_s
```

```
## [1] 0.9750425
```

```r
#Item_MRP + Outlet_Size_ + `Outlet_Type_Grocery Store` + `Outlet_Type_Supermarket Type1` + `Outlet_Type_

str(splitted_train)
```

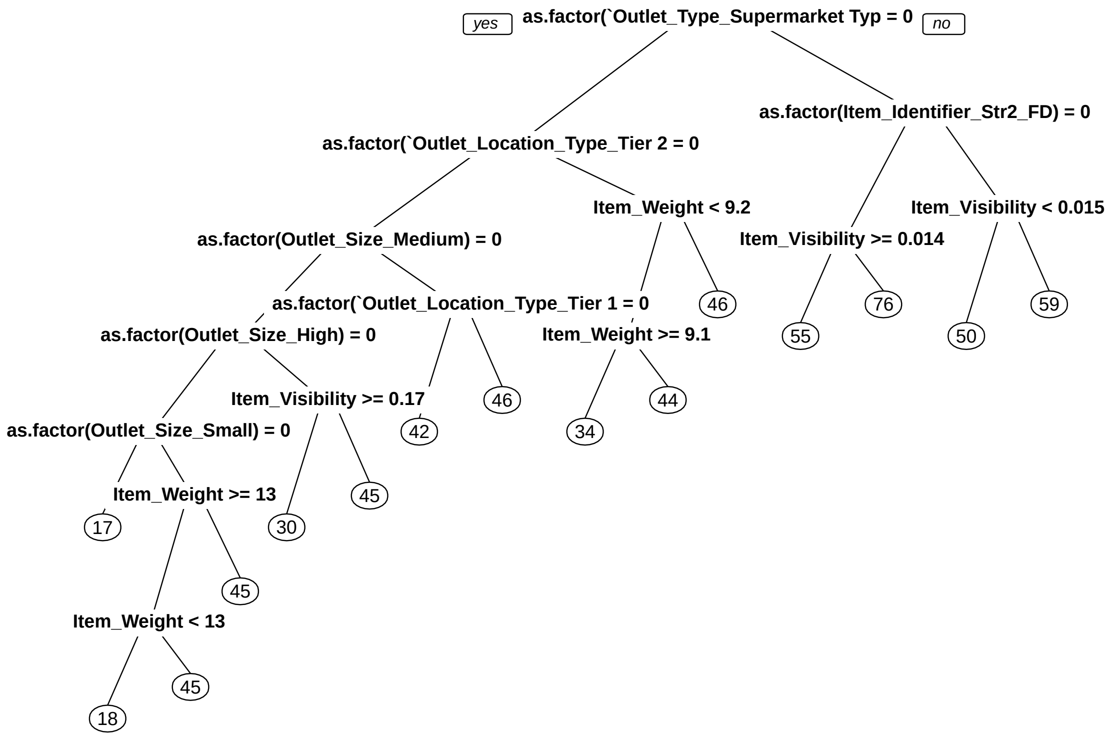
```
## 'data.frame':    5966 obs. of  20 variables:
##  $ Item_Weight                  : num  9 6.71 19.5 12.6 14.65 ...
##  $ Item_Visibility              : num  0.032 0.2035 0.1282 0.0553 0.0993 ...
##  $ Item_MRP                     : num  99.6 41 156.1 222.5 49.9 ...
##  $ Outlet_Size_                 : int  0 1 0 0 0 1 0 0 0 0 ...
##  $ Outlet_Size_High             : int  0 0 0 0 0 0 0 0 1 0 ...
##  $ Outlet_Size_Medium           : int  0 0 0 1 1 0 1 1 0 1 ...
##  $ Outlet_Size_Small            : int  1 0 1 0 0 0 0 0 0 0 ...
##  $ Outlet_Location_Type_Tier 1  : int  0 0 0 0 1 0 1 1 0 0 ...
##  $ Outlet_Location_Type_Tier 2  : int  1 0 1 0 0 1 0 0 0 0 ...
##  $ Outlet_Location_Type_Tier 3  : int  0 1 0 1 0 0 0 0 1 1 ...
##  $ Outlet_Type_Grocery Store    : int  0 1 0 0 0 0 0 0 0 0 ...
##  $ Outlet_Type_Supermarket Type1: int  1 0 1 0 1 1 1 1 1 0 ...
##  $ Outlet_Type_Supermarket Type2: int  0 0 0 0 0 0 0 0 0 1 ...
##  $ Outlet_Type_Supermarket Type3: int  0 0 0 1 0 0 0 0 0 0 ...
##  $ Item_Outlet_Sales            : num  1923 252 2792 7159 564 ...
##  $ Item_Identifier_Str2_DR      : int  0 0 0 0 0 0 0 1 0 0 ...
##  $ Item_Identifier_Str2_FD      : int  0 1 1 1 1 1 1 0 1 0 ...
##  $ Item_Identifier_Str2_NC      : int  1 0 0 0 0 0 0 0 0 1 ...
##  $ Item_Identifier_Num          : num  7 36 35 11 22 9 57 48 57 29 ...
##  $ Outlet_Age                   : num  9 15 9 28 14 11 14 14 26 4 ...
##  - attr(*, "dummies")=List of 4
##   ..$ Outlet_Size         : int  9 10 11 12
##   ..$ Outlet_Location_Type: int  13 14 15
##   ..$ Outlet_Type         : int  16 17 18 19
##   ..$ Item_Identifier_Str2: int  22 23 24
```

- Model4: Linear Regression based on summary of regressions(p values<0.05) and based on decision tree results(We created new variable with decision tree using variables which are not used in lineer regression).

```r
(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)))
```

```
## 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)
```

```r
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

```
splitted_train_new <- mutate(splitted_train,decision_tree_result = predict(main_tree, splitted_train))
splitted_test_new <- mutate(splitted_test,decision_tree_result = predict(main_tree, splitted_test))

(formula_sqrt_tree <- as.formula(sqrt(Item_Outlet_Sales) ~ Item_MRP + Outlet_Size_ + `Outlet_Type_Groce
```

```
## sqrt(Item_Outlet_Sales) ~ Item_MRP + Outlet_Size_ + `Outlet_Type_Grocery Store` +
##     `Outlet_Type_Supermarket Type1` + `Outlet_Type_Supermarket Type2` +
##     Item_Identifier_Num + Outlet_Age + decision_tree_result
```

```
linear_model_sqrt_tree <- lm(formula_sqrt_tree, data = splitted_train_new)
summary(linear_model_sqrt_tree)
```

```
##
## Call:
## lm(formula = formula_sqrt_tree, data = splitted_train_new)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -43.848  -6.727   0.108   6.934  43.795
##
## Coefficients:
##                                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)                     11.581842   4.777798   2.424  0.01538 *
## Item_MRP                         0.165014   0.002230  74.005  < 2e-16 ***
## Outlet_Size_                    -1.119991   0.430668  -2.601  0.00933 **
## `Outlet_Type_Grocery Store`    -23.492769   3.305240  -7.108 1.32e-12 ***
## `Outlet_Type_Supermarket Type1`  -8.252826   1.184357  -6.968 3.56e-12 ***
## `Outlet_Type_Supermarket Type2` -12.045619   1.655579  -7.276 3.89e-13 ***
## Item_Identifier_Num               0.033000   0.008016   4.117 3.89e-05 ***
## Outlet_Age                       -0.091119   0.032182  -2.831  0.00465 **
## decision_tree_result              0.427309   0.079731   5.359 8.66e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 10.71 on 5957 degrees of freedom
## Multiple R-squared:  0.656,  Adjusted R-squared:  0.6555
## F-statistic:  1420 on 8 and 5957 DF,  p-value: < 2.2e-16
```

```
# make prediction with test data (%30 of actual train data)
pred_sqrt_test_tree <-predict(linear_model_sqrt_tree, splitted_test_new) ^ 2
pred_sqrt_train_tree <-predict(linear_model_sqrt_tree, splitted_train_new) ^ 2


# MAE of Sqrt for Test
(MAE(splitted_test$Item_Outlet_Sales, pred_sqrt_test_tree))
```

```
## [1] 784.439
```

```
# RMSE of Sqrt for test dataset
(RMSE(splitted_test$Item_Outlet_Sales, pred_sqrt_test_tree))
```

```
## [1] 1121.72
```

```
# RMSE of Sqrt for train dataset
(RMSE(splitted_train$Item_Outlet_Sales, pred_sqrt_train_tree))
```

```
## [1] 1086.913
```

```
# RMSE of Sqrt for train/test
(RMSE(splitted_train$Item_Outlet_Sales, pred_sqrt_train_tree))/(RMSE(splitted_test$Item_Outlet_Sales, p
```

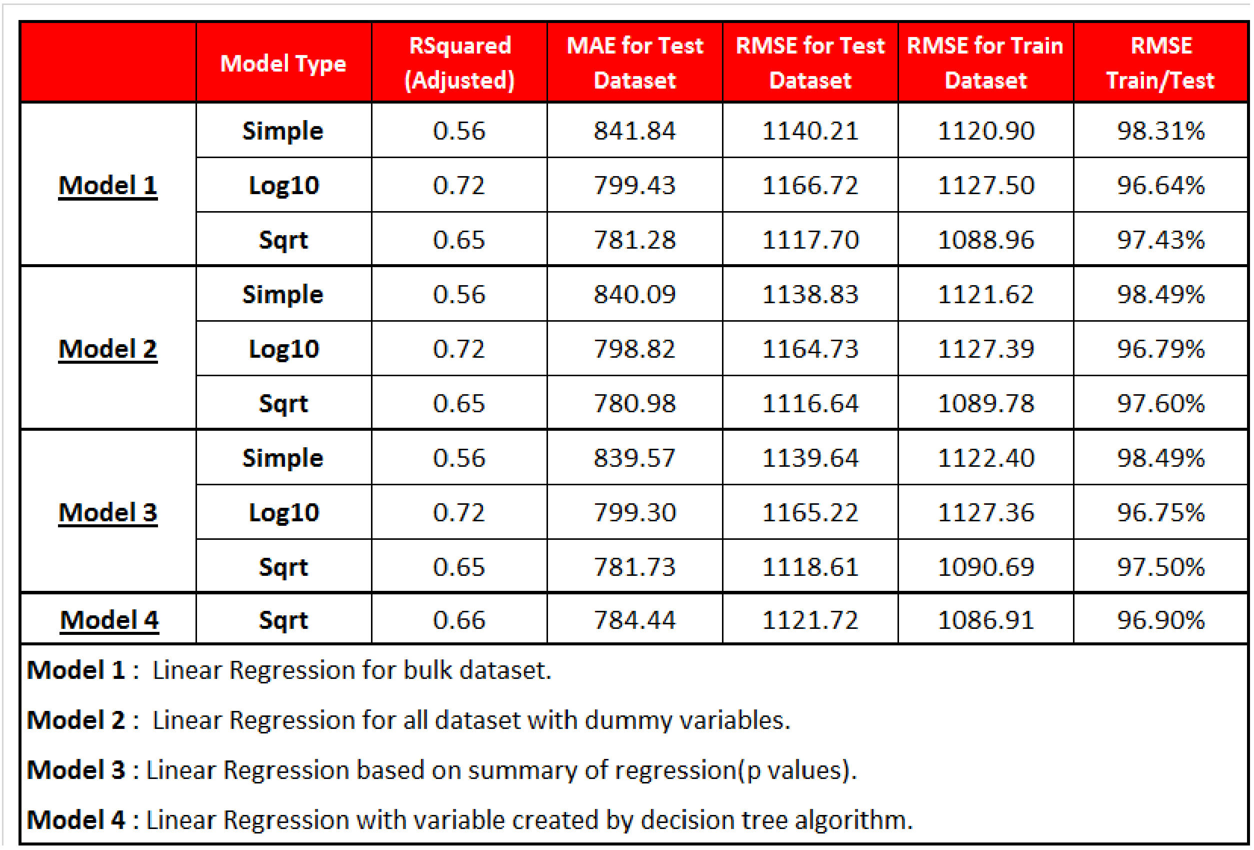
```
## [1] 0.9689697
```

4.2. Summary of Modelling

- Summary of all 4 models can be examined as below. Model 1 is built with bulk cleaned data, Model 2 is generated with dummy variables that created while Model 3 has selected columns of dummy dataset according to regression, and model 4 is building a decision tree. And for each dataset, we try the models for simple Item_Outlet_Sales column, log10 and square-root of it.
- Firstly, when we look at Adjusted R-Squared values, they are almost same for each dataset although we change the selected columns for each. It means that the columns that affect mostly are selected for each model.
- Adjusted R^2 measures the goodness of fit of a regression model. Higher the R^2, better is the model. When we look at the table, we can say that for eachl model, logarithmic of the target column gives us much more better model. This is interesting because we explored on visualizing data part, the squareroot of this column is inclined to be normal distribution. So, logatihmic surprises us !
- So, we can say that according to R-Squared values, linear regression with bulk dataset without new features (but cleaned) is the best when the target column is logarithmic. It does not matter if the dataset has new features because the significant predictors such as Item_MRP,Outlet_Type,Outlet_Size,Outlet_Age are already in dataset according to summary of models.
- Secondly,Error measures in the estimation period: Mean Absolute Error (MAE) and Root mean squared error (RMSE) are two of the most common metrics used to measure accuracy for continuous variables.

MAE is the average over the test sample of the absolute differences between prediction and actual observation where all individual differences have equal weight. RMSE is a quadratic scoring rule that also measures the average magnitude of the error. It's the square root of the average of squared differences between prediction and actual observation. Both MAE and RMSE express average model prediction error in units of the variable of interest.

- So, when we look at the MAE of TEST data for each model, we can say SQRT of target column gives us better model for each model, but there is no big difference between logarithmic. It is expected because squareroot of the target column has normal distribution as we mentioned before. According to MAE values, model 2 with sqrt of the target column is much more better.
- The RMSE is the square root of the variance of the residuals. It indicates the absolute fit of the model to the data–how close the observed data points are to the model's predicted values. Whereas R-squared is a relative measure of fit, RMSE is an absolute measure of fit.
- RMSE ratio is equal to RMSE_Train / RMSE_Test, so it means there is no big difference between our train and test sample.
- Lower values of RMSE indicate better fit. RMSE is a good measure of how accurately the model predicts the response, and is the most important criterion for fit if the main purpose of the model is prediction.



	Model Type	RSquared (Adjusted)	MAE for Test Dataset	RMSE for Test Dataset	RMSE for Train Dataset	RMSE Train/Test
Model 1	**Simple**	0.56	841.84	1140.21	1120.90	98.31%
	Log10	0.72	799.43	1166.72	1127.50	96.64%
	Sqrt	0.65	781.28	1117.70	1088.96	97.43%
Model 2	**Simple**	0.56	840.09	1138.83	1121.62	98.49%
	Log10	0.72	798.82	1164.73	1127.39	96.79%
	Sqrt	0.65	780.98	1116.64	1089.78	97.60%
Model 3	**Simple**	0.56	839.57	1139.64	1122.40	98.49%
	Log10	0.72	799.30	1165.22	1127.36	96.75%
	Sqrt	0.65	781.73	1118.61	1090.69	97.50%
Model 4	**Sqrt**	0.66	784.44	1121.72	1086.91	96.90%

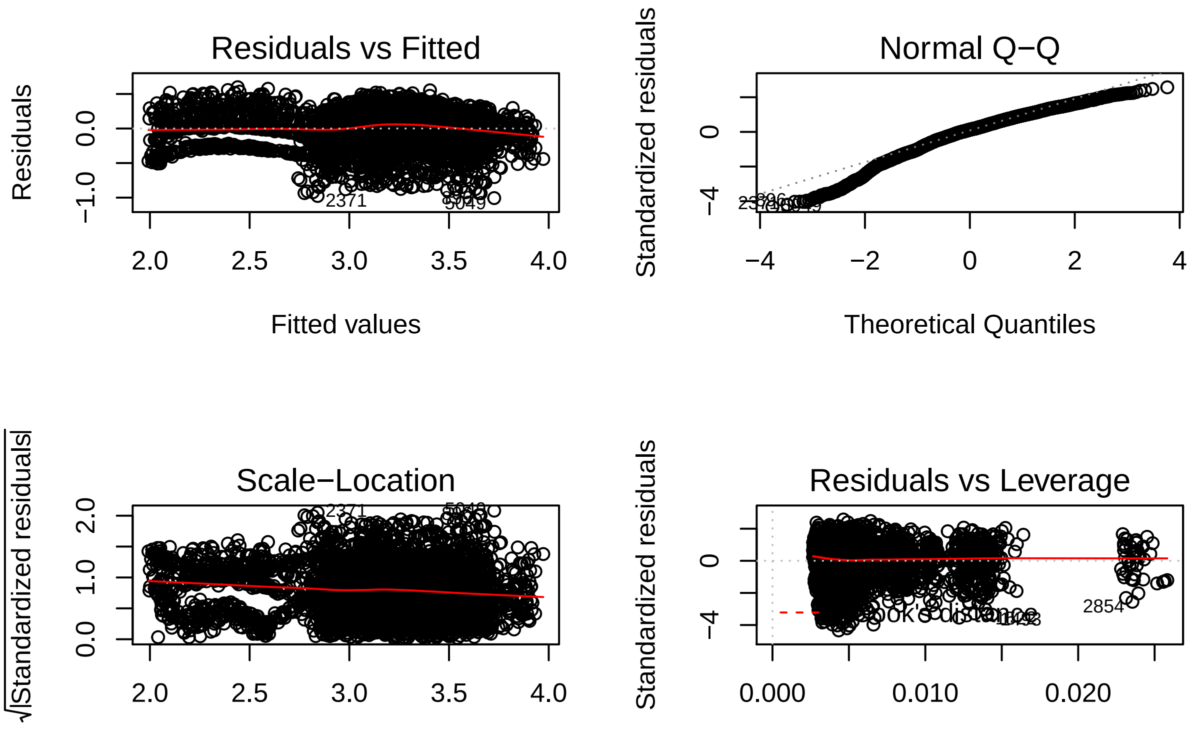
Model 1 : Linear Regression for bulk dataset.

Model 2 : Linear Regression for all dataset with dummy variables.

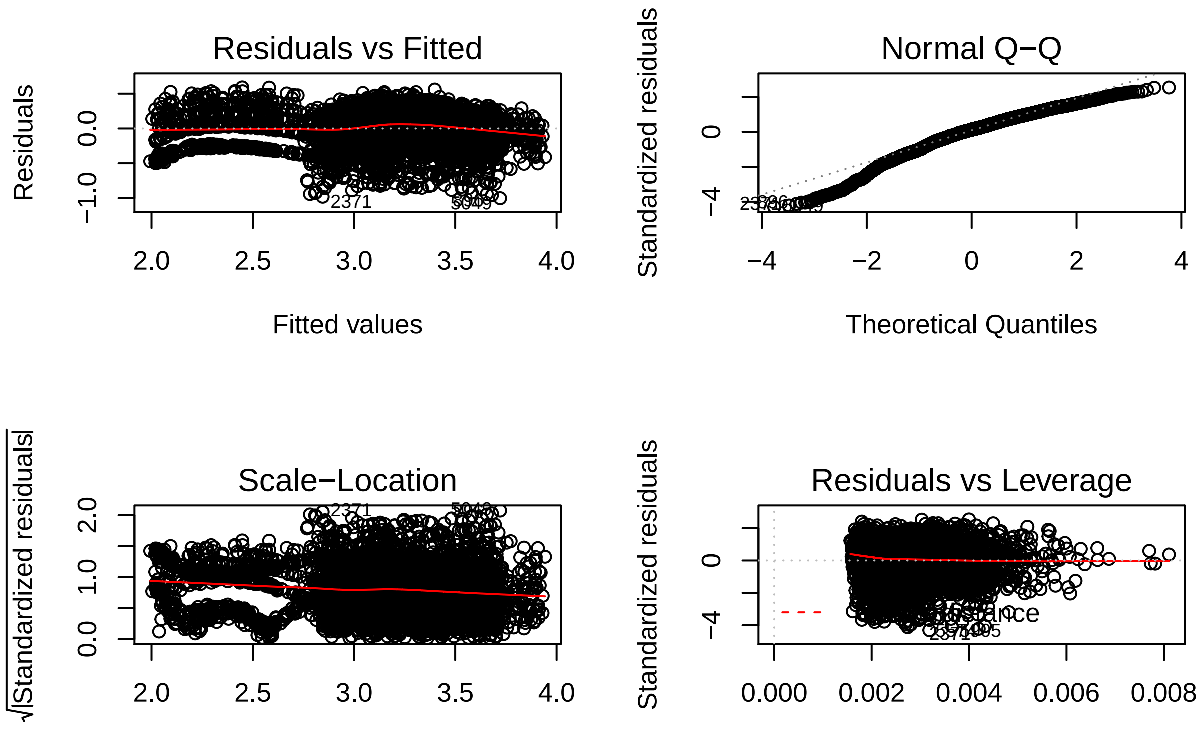
Model 3 : Linear Regression based on summary of regression(p values).

Model 4 : Linear Regression with variable created by decision tree algorithm.

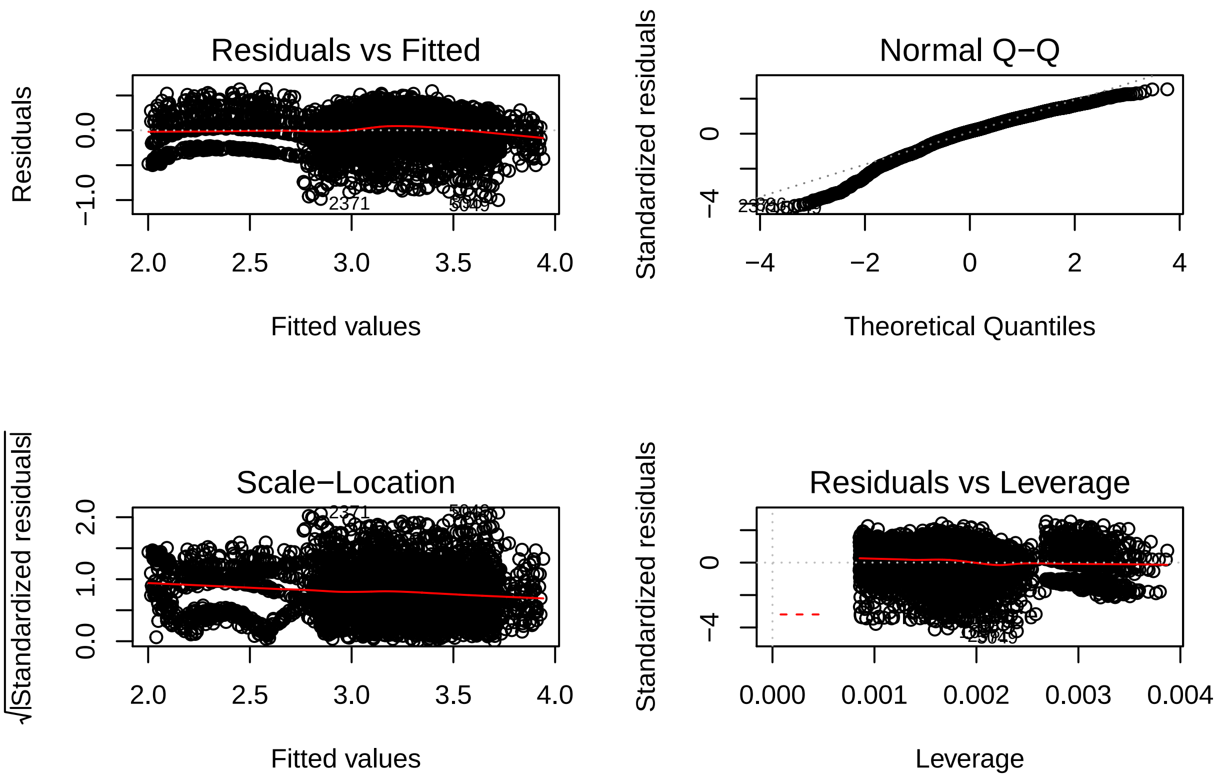
```r
par(mfrow = c(2,2))

#par(mfrow=c(2,2))
plot(linear_model_log1)
```



```r
plot(linear_model_log2)
```



```r
plot(linear_model_log3)
```



- As we can see on Residuals vs Fitted graph of all models with log target column, there is no specific trend on them. So, we can say linear regression model with logarithmic target is the best way for model according to R-Square and residual vs fitted value plot. This model can be further improved by detecting outliers and high leverage points. **In conclusion, we can say that we should use linear regression with cleaned-bulk data and logarithmic target column for this dataset.**

References

Dataset

- Big Mart Sales Practise Problem

Analysis & Graphs

- Dplyr
- https://www.rstudio.com/resources/cheatsheets/
- https://mef-bda503.github.io/files/02_Tidyverse.html#1
- Udacity Data Analysis with R - Lesson 3 and 5
- DataCamp - Cleaning Data with R
- R Project-An Introduction to corrplot Package
- https://plot.ly/ggplot2/
- ggplot2
- gplot2

Modeling

- statistical-modeling-in-r-part-1
- Kernel from Analyticsvidya.com
- Observing the Results