Predicting Profitability

Marc Soler

15/2/2019

My github page to download the code

The aim of this Machine Learning project is to predict the volume sales of new products never sold before.

The approach to solve the problem is as follows: An exploratory analysis of the dataset, identify relevant product features, cleaning data and visualizing distributions. Building and training Machine Learning Models with different algorithms (RF, GBM,SVM, KNN) Datasets and the complete code can be downloaded at the appendix.

Executive Summary

- The CTO Danielle Sherman requested the Technology department to predict the sales in four different product types: PC, Laptops, Netbooks and Smartphones.
- Balckwell provided the Technology department with an historical dataset of products previously in sale on the website
- To achieve our goal, we had to select the attributes that had the strongest relationship with sales, then train several models on the aforesaid attributes choosing the one with the best metrics.

Once selected the model, we predicted the sales of each products. The following table summarizes our findings:

ProductType	PredictedSalesVolume
Netbook	1577.38573
Smartphone	581.80453
PC	445.55240
Smartphone	349.80493
Laptop	273.27480
PC	212.91320
Smartphone	133.75187
Netbook	116.54840
Smartphone	74.54840
Netbook	39.30853
Laptop	29.58147
Laptop	16.95280
Netbook	16.04707

Cause of Action

- Data Preparation:
  - Data Exploration by looking at the distribution of the attributes with histograms
  - Found out that the outliers of the Sale Volumes belong to the product type “Accessories” and “Game Console”
  - Removed such outliers in order to have a sounder model
  - Identification and removal of missing data.

  - Use of Correlation matrix and of Random Forest on the attributes to identify the ones that have the strongest relationship with the volume of the sales.
  - The attributes 4 Stars Review and Positive Review were selected after this process.
- Technical Approach
  - Thanks to a Nested Loop we ran four different models (i.e. Random forest, Gradient Boosting Machine, Support Vector Machine and K nearest neighbor) against two possible combination of attributes, the first one being Volume against 4 Star Reviews and Positive Service Review, and the second one being Volume against all the attributes of the dataset.
  - The model that we believe has the highest probability of predicting the right sales is the random forest trained on all the variables of our dataset

Insights and Recommendations

- Small dataset:
  - The main issues with this task is the little amount of historical data.
  - With more observation we could have broken down by Product Type and we could have done a personalized analysis for each one of them.
  - Because of the small size, we had to be too cautious with removing observations even though not deemed relevant, since otherwise the dataset would have been too small, and any inference would have little relevance in the real world
  - Thus, we would recommend increasing the resources dedicated to the collection of data, either by acquiring it from third parties or by increasing the data mining forces inside of our company
- Distribution of Products:
  - As we can see from the Appendix, the Products most sold are the one belonging to the category "Accessories", followed by Printer and Extended Warranty.
  - On the other hand, Game Console which was sold the least, followed by Notebook
  - Looking at the distribution of Sales Volumes, we can see that for some Products it is more uniform than for others. However, because of the little data, we cannot infer much on this matter.

Investigating the dataset

```
#import relevant libraries
library(readxl)
library(caret)
library(corrplot)
library(ggplot2)
library(RColorBrewer)
library(kernlab)
library(randomForest)
library(gbm)
library(plyr)
library(dplyr)
library(reshape2)
library(party)
```

```r
#import data
existing_atributes <- read.csv('existingproductattributes20172.csv')
```

```r
#This will show the structure of the dataset, to give us an overview of what is ahead
glimpse(existing_atributes)
```

```
## Observations: 80
## Variables: 18
## $ ProductType           <fct> PC, PC, PC, Laptop, Laptop, Accessories,...
## $ ProductNum            <int> 101, 102, 103, 104, 105, 106, 107, 108, ...
## $ Price                 <dbl> 949.00, 2249.99, 399.00, 409.99, 1079.99...
## $ x5StarReviews         <int> 3, 2, 3, 49, 58, 83, 11, 33, 16, 10, 21,...
## $ x4StarReviews         <int> 3, 1, 0, 19, 31, 30, 3, 19, 9, 1, 2, 25,...
## $ x3StarReviews         <int> 2, 0, 0, 8, 11, 10, 0, 12, 2, 1, 2, 6, 5...
## $ x2StarReviews         <int> 0, 0, 0, 3, 7, 9, 0, 5, 0, 0, 4, 3, 0, 8...
## $ x1StarReviews         <int> 0, 0, 0, 9, 36, 40, 1, 9, 2, 0, 15, 3, 1...
## $ PositiveServiceReview <int> 2, 1, 1, 7, 7, 12, 3, 5, 2, 2, 2, 9, 2, ...
## $ NegativeServiceReview <int> 0, 0, 0, 8, 20, 5, 0, 3, 1, 0, 1, 2, 0, ...
## $ Recommendproduct      <dbl> 0.9, 0.9, 0.9, 0.8, 0.7, 0.3, 0.9, 0.7, ...
## $ BestSellersRank       <int> 1967, 4806, 12076, 109, 268, 64, NA, 2, ...
## $ ShippingWeight        <dbl> 25.80, 50.00, 17.40, 5.70, 7.00, 1.60, 7...
## $ ProductDepth          <dbl> 23.94, 35.00, 10.50, 15.00, 12.90, 5.80,...
## $ ProductWidth          <dbl> 6.62, 31.75, 8.30, 9.90, 0.30, 4.00, 10....
## $ ProductHeight         <dbl> 16.89, 19.00, 10.20, 1.30, 8.90, 1.00, 1...
## $ ProfitMargin          <dbl> 0.15, 0.25, 0.08, 0.08, 0.09, 0.05, 0.05...
## $ Volume                <int> 12, 8, 12, 196, 232, 332, 44, 132, 64, 4...
```

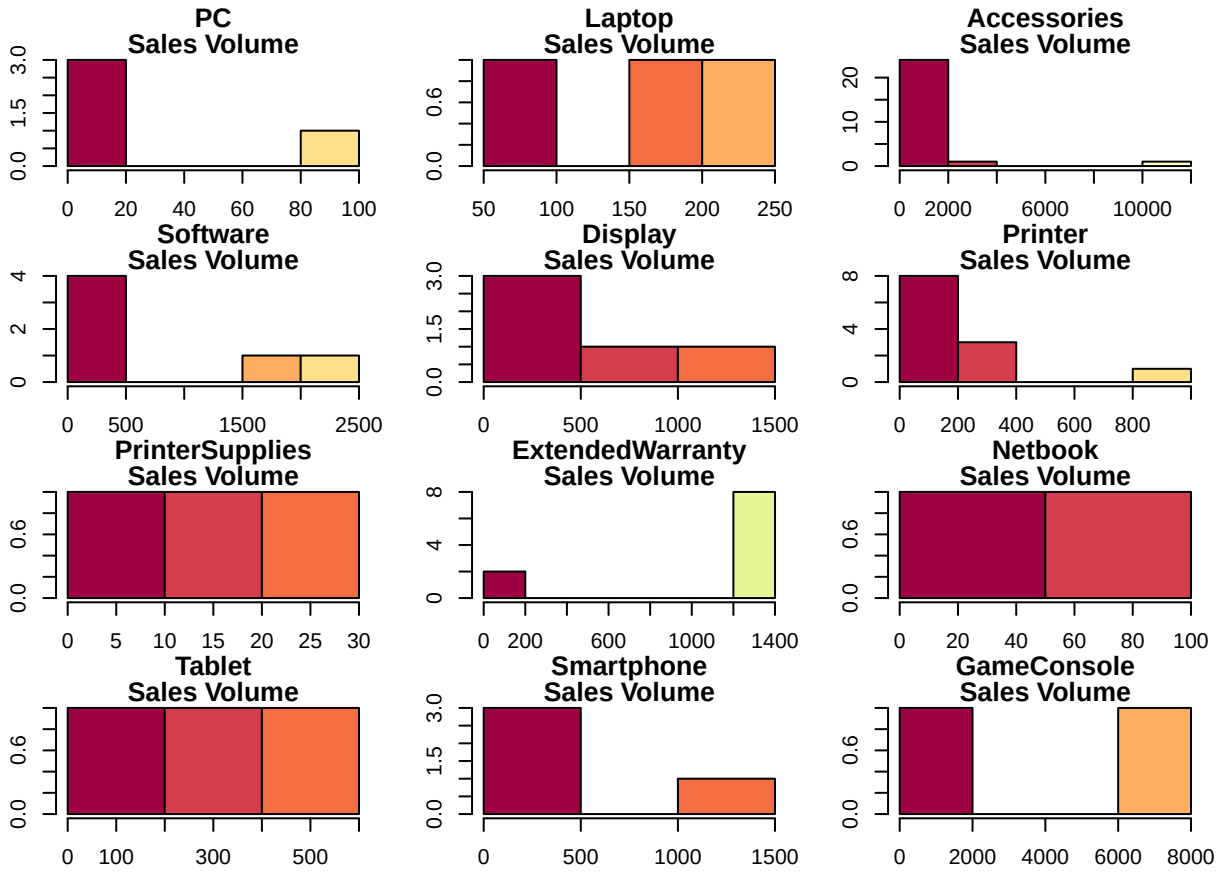
Check for unique product types

```r
if (length(unique(existing_atributes$ProductNum)) == nrow(existing_atributes)) {
  print(paste("There are", length(unique(existing_atributes$ProductNum)), "unique ProductNum values out
  } else { (print("CHECK: There is at least 1 repeated ID value in the 80 observations."))
  }
```

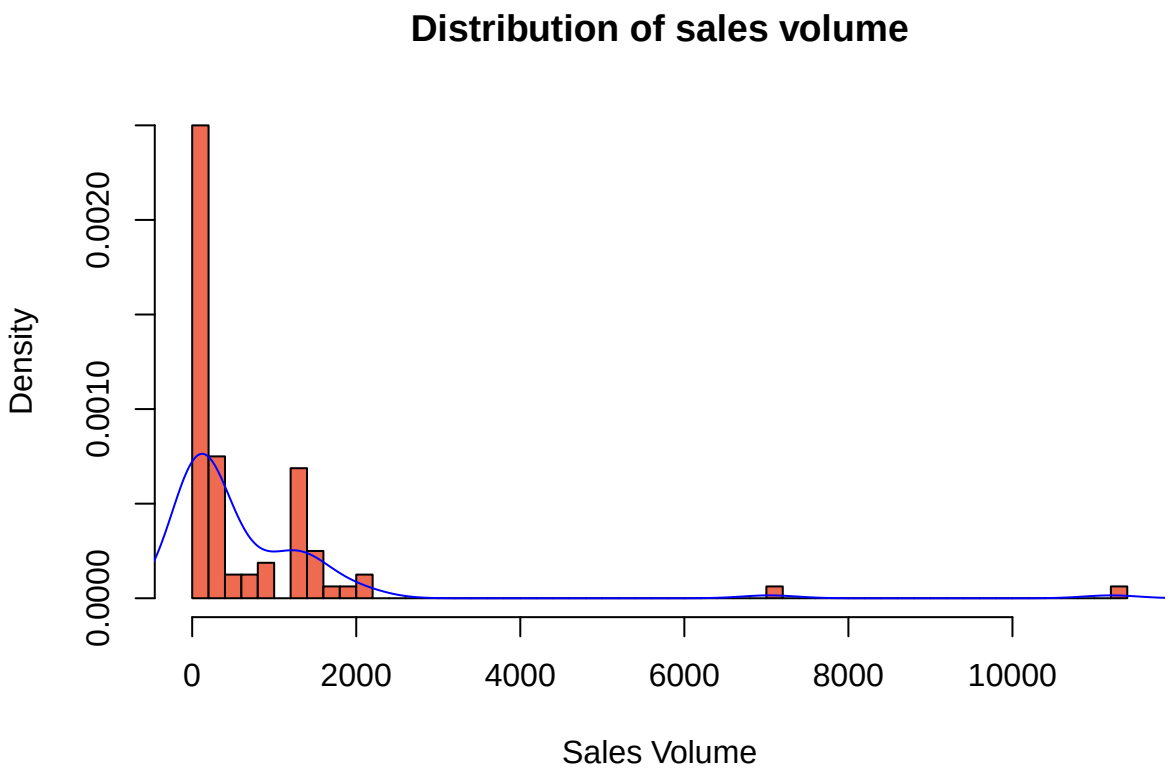
```
## [1] "There are 80 unique ProductNum values out of 80 observations."
```

Check how sales volume is distributed accross different product types

```r
#plot histograms for all product sales volume to see outliers
products <- unique(existing_atributes$ProductType)
par(mfrow=c(4,3), par(mar=c(2,2,2,2)))
for (p in products){
    hist(existing_atributes$Volume[existing_atributes$ProductType == p],
         main = c(p , 'Sales Volume'), col = brewer.pal(11,'Spectral'))
}
```



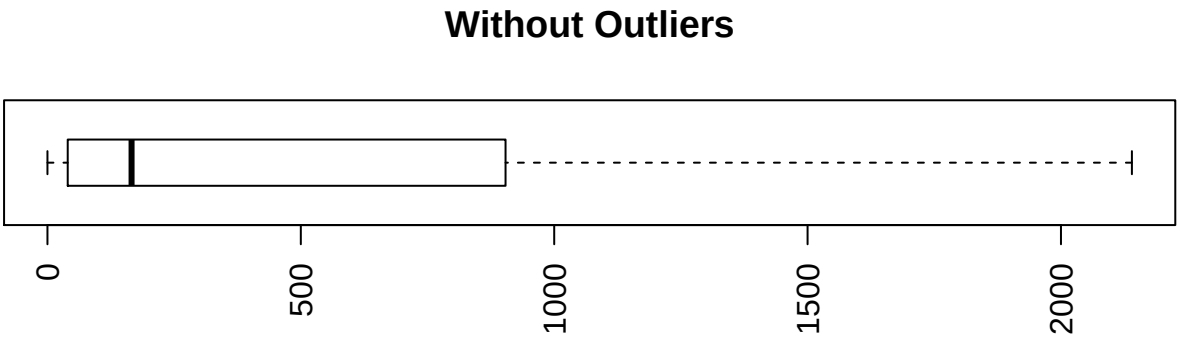

```
hist(existing_atributes$Volume, breaks = 50, probability = TRUE, col = 'coral2',
     main = 'Distribution of sales volume', xlab = 'Sales Volume')
lines(density(existing_atributes$Volume), col='blue')
```



Clearly, sales volume ranges from 0 to 2000. There are two clear outliers. From the matrix plot we can distinguish they belong to Accessories and Game Consoles.

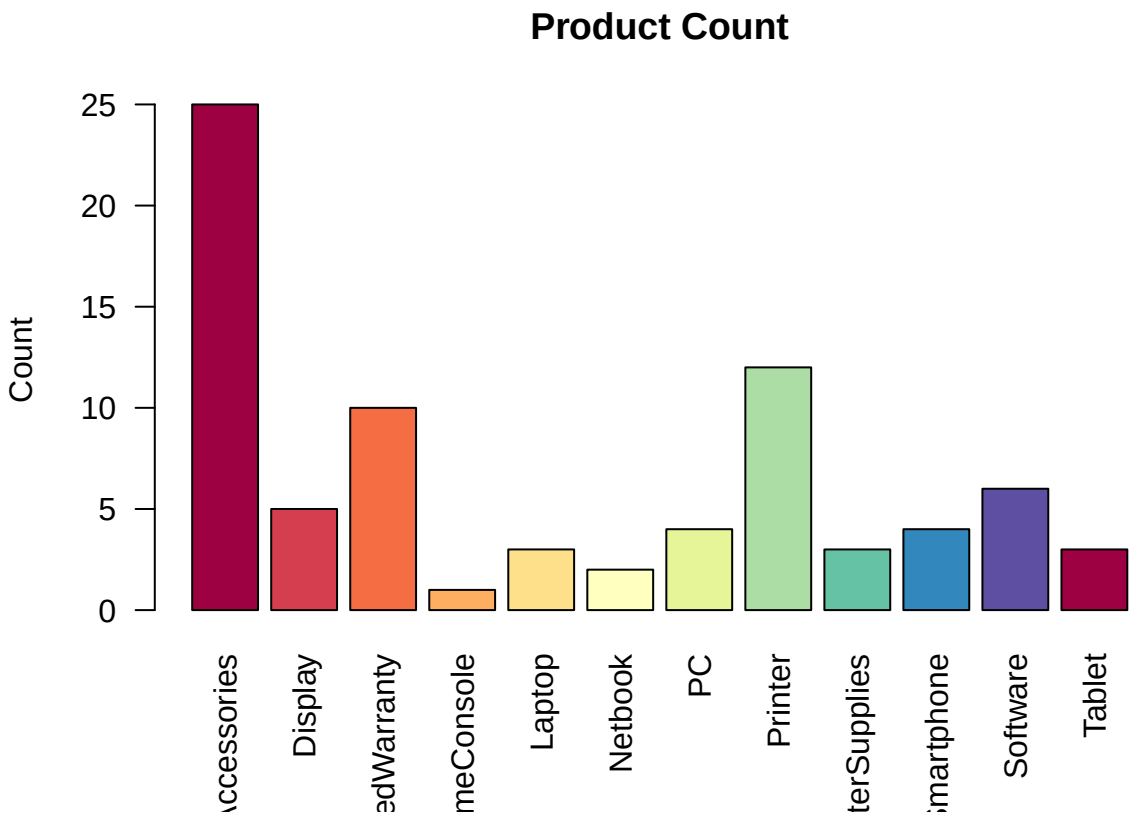
Removing outliers from the dataset

```
#generate oultier indices (detect the row number they belong in the dataset)
out_indices <- which(existing_atributes$Volume %in% boxplot$out)
max_outliers <- c()
for (i in out_indices){
    max_outliers <- sort(c(max_outliers, existing_atributes[i,'Volume']), decreasing = T)
}

#remove the two outliers from the dataset
row_outlier1 <- which(existing_atributes$Volume == max_outliers[1])
row_outlier2 <- which(existing_atributes$Volume == max_outliers[2])
existing_atributes2 <- existing_atributes[c(-row_outlier1, -row_outlier2),]
```



Visualizing the product distribution

```
product_freq <- as.data.frame(table(existing_atributes2$ProductType))
barplot(product_freq$Freq, col = brewer.pal(11, 'Spectral'),
        names.arg = product_freq$Var1, las = 2, main = 'Product Count',
        ylab = 'Count')
```



Feature Selection: Selecting appropiate attributes for the models

Check for missing values and remove them

```
#Let's find out which ProductType has zero sales volume and remove them from the dataset
zero_index <- which(existing_atributes2$Volume == 0)
existing_atributes2[zero_index,1]
```

```
## [1] PrinterSupplies  ExtendedWarranty Printer
## 12 Levels: Accessories Display ExtendedWarranty GameConsole ... Tablet
```

```
#remove zero volume items from dataset
existing_atributes2 <- existing_atributes2[-c(zero_index),]
```

```
#First let's check if we have missing values at all
any(is.na(existing_atributes2))
```

```
## [1] TRUE
```

```
#Secondly, let's check which column has missing values
apply(existing_atributes2, 2, function(x) any(is.na(x) | is.infinite(x)))
```

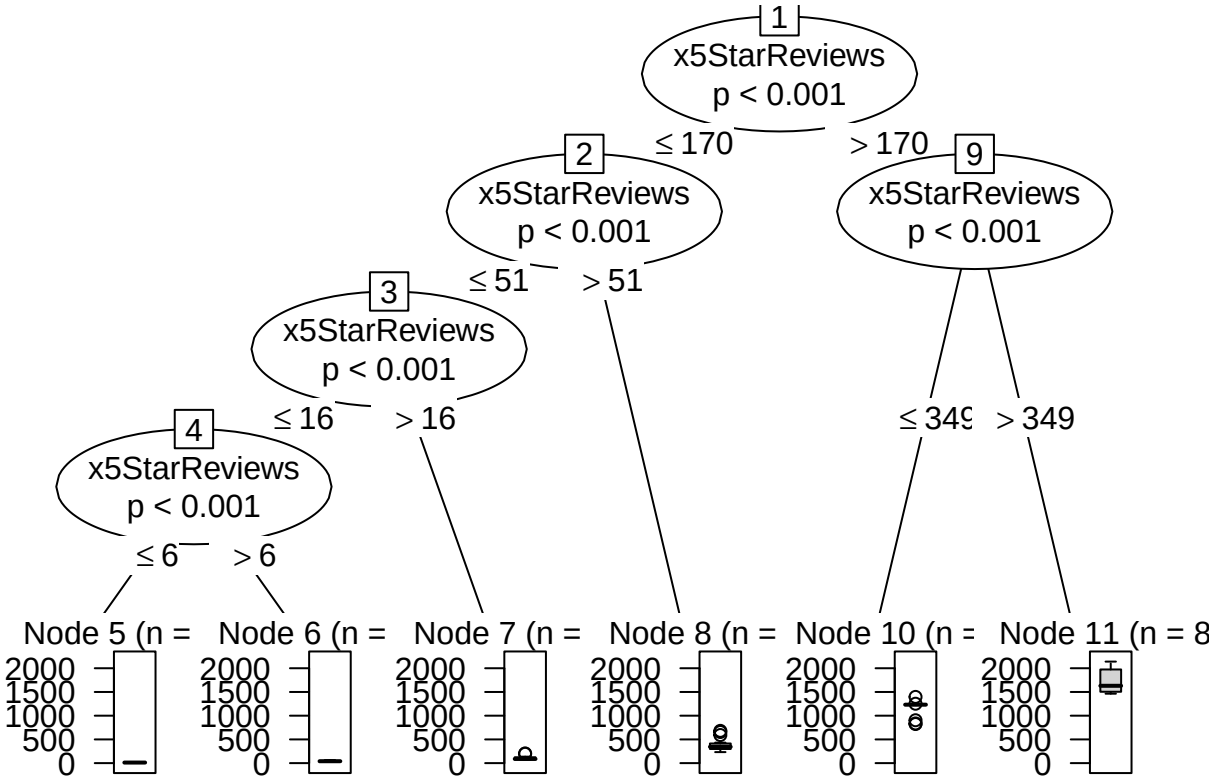
```
##           ProductType            ProductNum                 Price
##                 FALSE                 FALSE                 FALSE
##         x5StarReviews         x4StarReviews         x3StarReviews
##                 FALSE                 FALSE                 FALSE
##         x2StarReviews         x1StarReviews PositiveServiceReview
##                 FALSE                 FALSE                 FALSE
```

```
## NegativeServiceReview      Recommendproduct       BestSellersRank
##                 FALSE                 FALSE                  TRUE
##        ShippingWeight          ProductDepth          ProductWidth
##                 FALSE                 FALSE                 FALSE
##         ProductHeight          ProfitMargin                Volume
##                 FALSE                 FALSE                 FALSE
```

```
#Remove BestSellersRank
existing_atributes2$BestSellersRank <- NULL
```

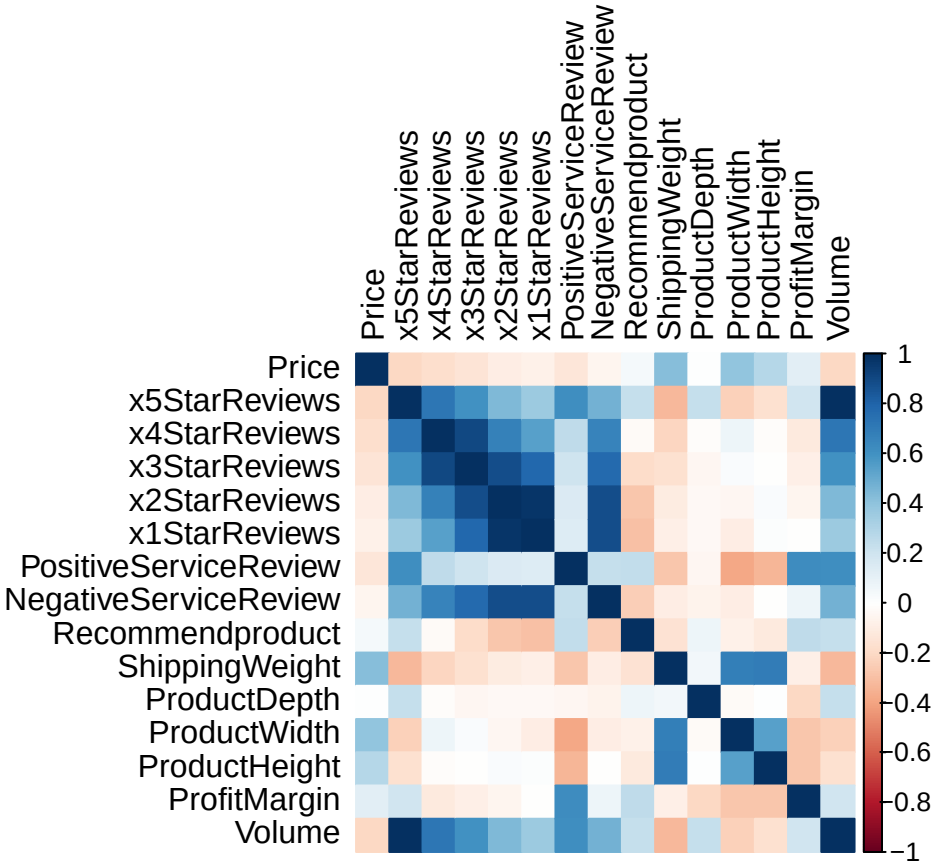
Decision tree

```
# Dummify product type and carry out correlations between all the variables for removing redundance.
Dummify <- dummyVars(" ~ .", data = existing_atributes2)
existing_atributes2Dum <- data.frame(predict(Dummify, newdata = existing_atributes2))

# Decision Tree for knowing the relevant variables
tree <-ctree(Volume~., existing_atributes2Dum, control = ctree_control(maxdepth = 6))
plot(tree)
```

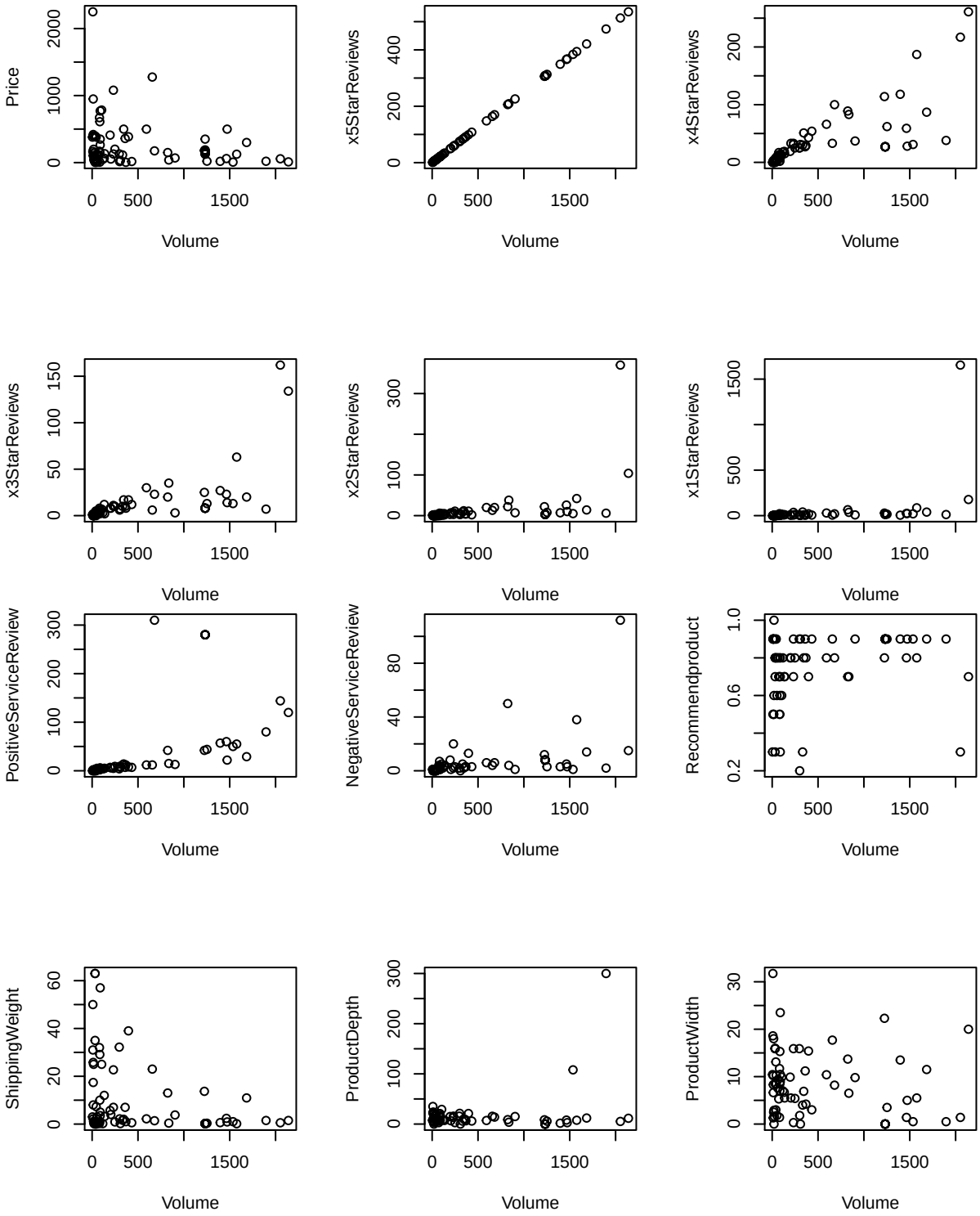


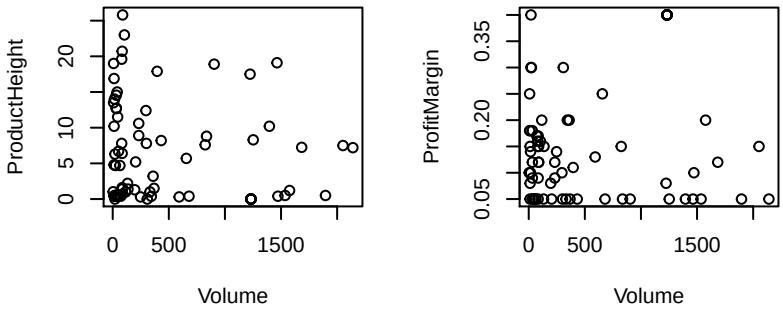
Correlation between Attributes

```
#Generate a first correlation matrix with all attributes
CM <- cor(subset(existing_atributes2, select = c(-ProductType, -ProductNum)))#exclude ProductNum since
corrplot(CM, method = 'color', tl.col = 'black')
```


Price
x5StarReviews
x4StarReviews
x3StarReviews
x2StarReviews
x1StarReviews
PositiveServiceReview
NegativeServiceReview
Recommendproduct
ShippingWeight
ProductDepth
ProductWidth
ProductHeight
ProfitMargin
Volume
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1

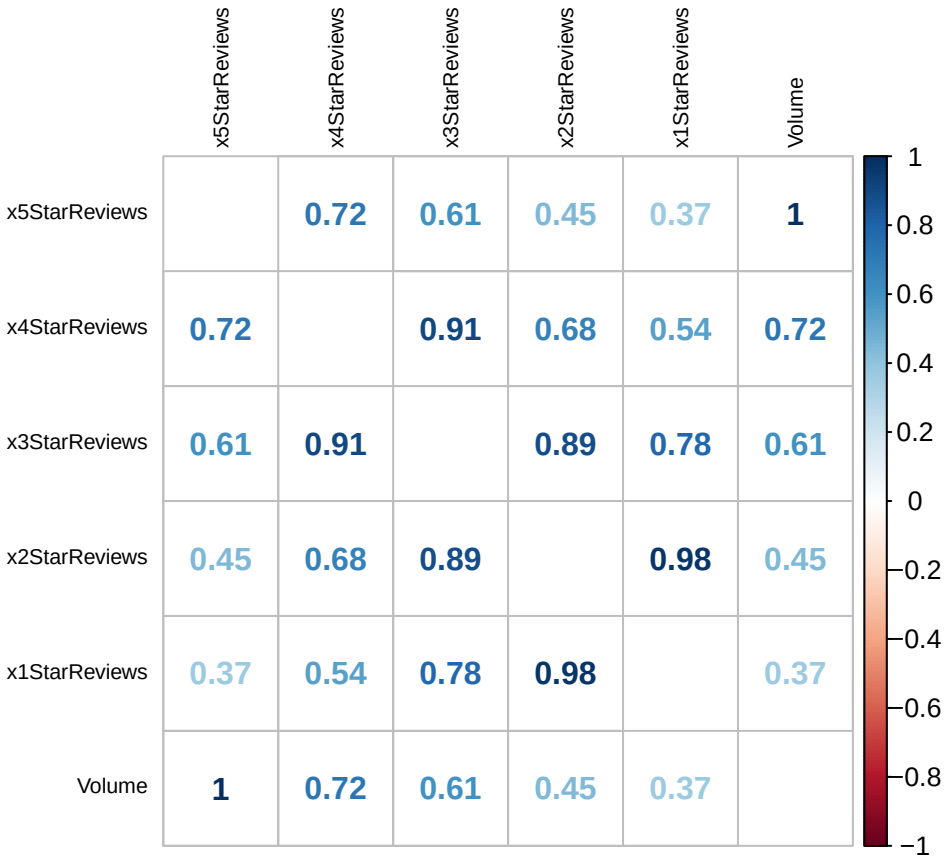
Visualize the distribution of our target variable and its relation with the other features





Colinearity

```
#Star ratings seem to have a 'suspicious' colinearity
#generate a new correlation matrix to investigate further
cor_stars = as.matrix(existing_atributes2[,c(4:8, 17)])
star_corr_mat = cor(cor_stars)
corrplot(corr = star_corr_mat, diag = F, method = "number", tl.col = 'black', tl.cex = .7)
```



Volume and 5 star have correlation of 1. A correlation of 1 means that there is a perfect linear relation. Hence, knowing the number of 5 stars it's possible to calculate the sales volume.

In the real world this makes no sense. It shows that this dataset might be created artificially. It is true that positive reviews are proportionally correlated to Volume of sales, but not to the point where the proportionality is mathematically perfect.

For tat reason, 5 star reviews attribute will be removed completely. Feature engineering will allow us to avoid collinearity among attributes and improve correlation and machine learning processes.

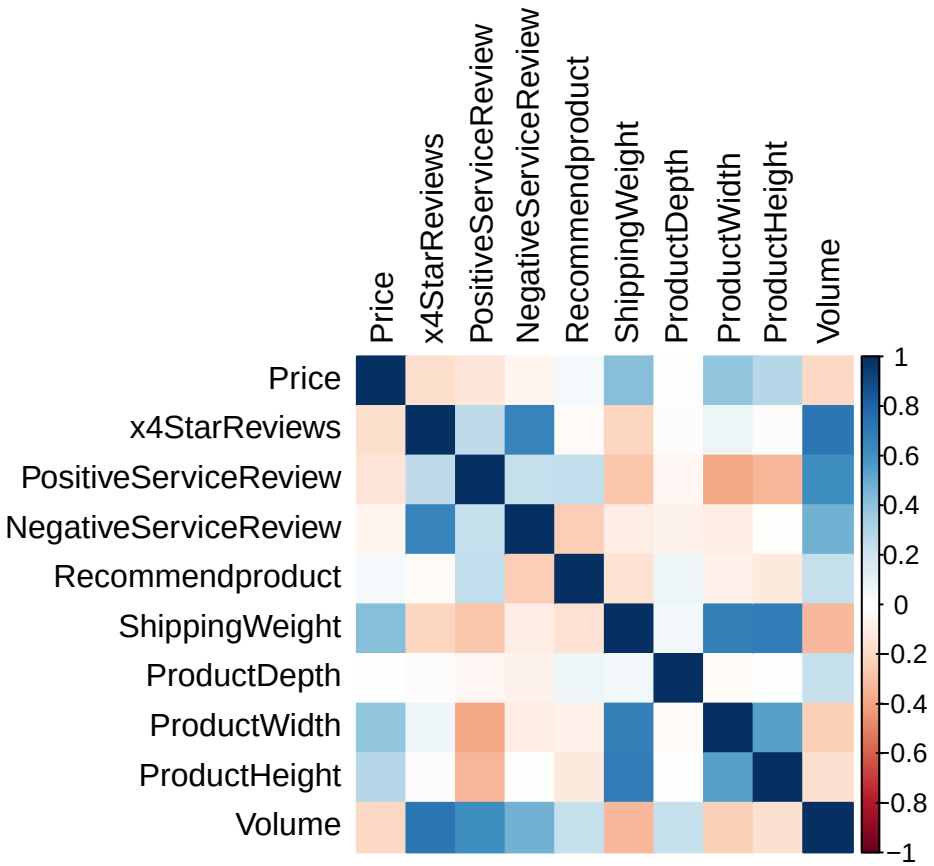
Remove Attributes

Deleting attributes that are colinear and also attributes that customers are not aware and are irrelevant for a customers' point of view

```
existing_atributes2$x5StarReviews <- NULL
existing_atributes2$x3StarReviews <- NULL
existing_atributes2$x2StarReviews <- NULL
existing_atributes2$x1StarReviews <- NULL
existing_atributes2$ProfitMargin <- NULL
```

New Correlation Matrix

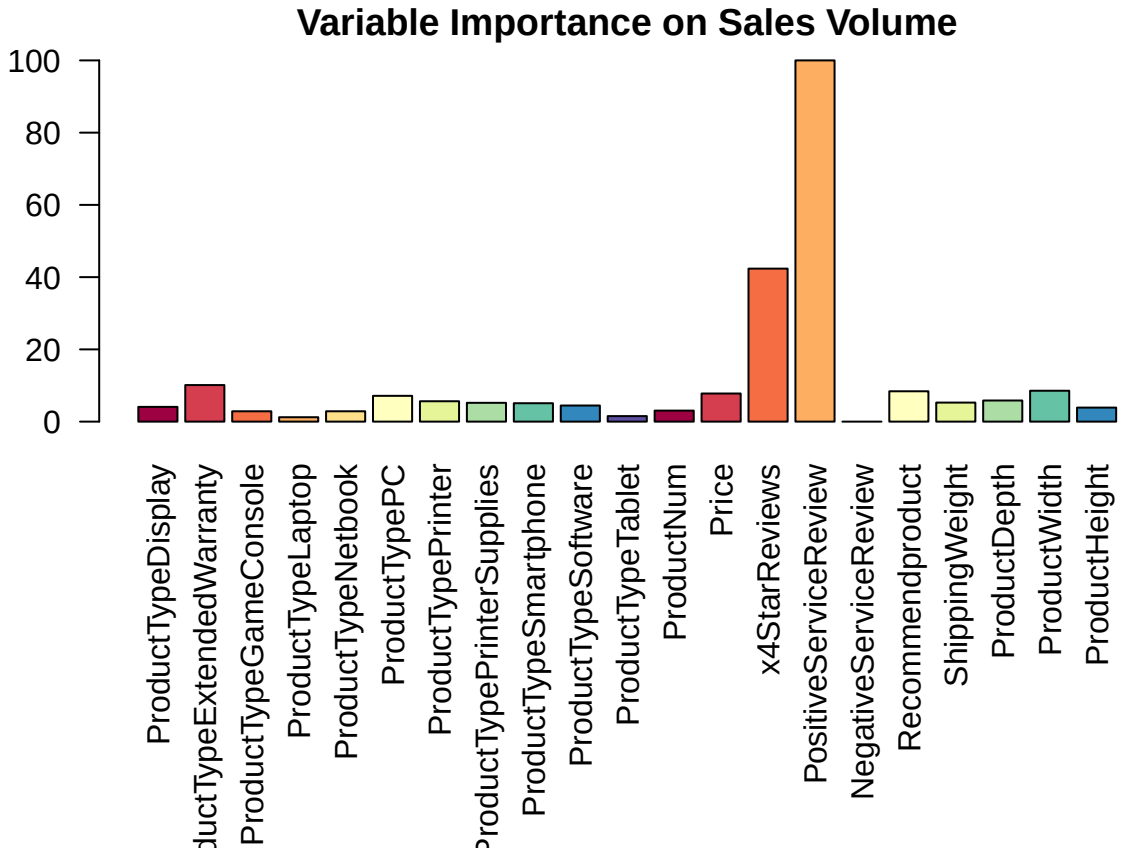
```
#create second CM
CM2 <- cor(subset(existing_atributes2, select = c(-ProductType, -ProductNum)))
corrplot(CM2, method = 'color', tl.col = 'black')
```



Attribute Importance

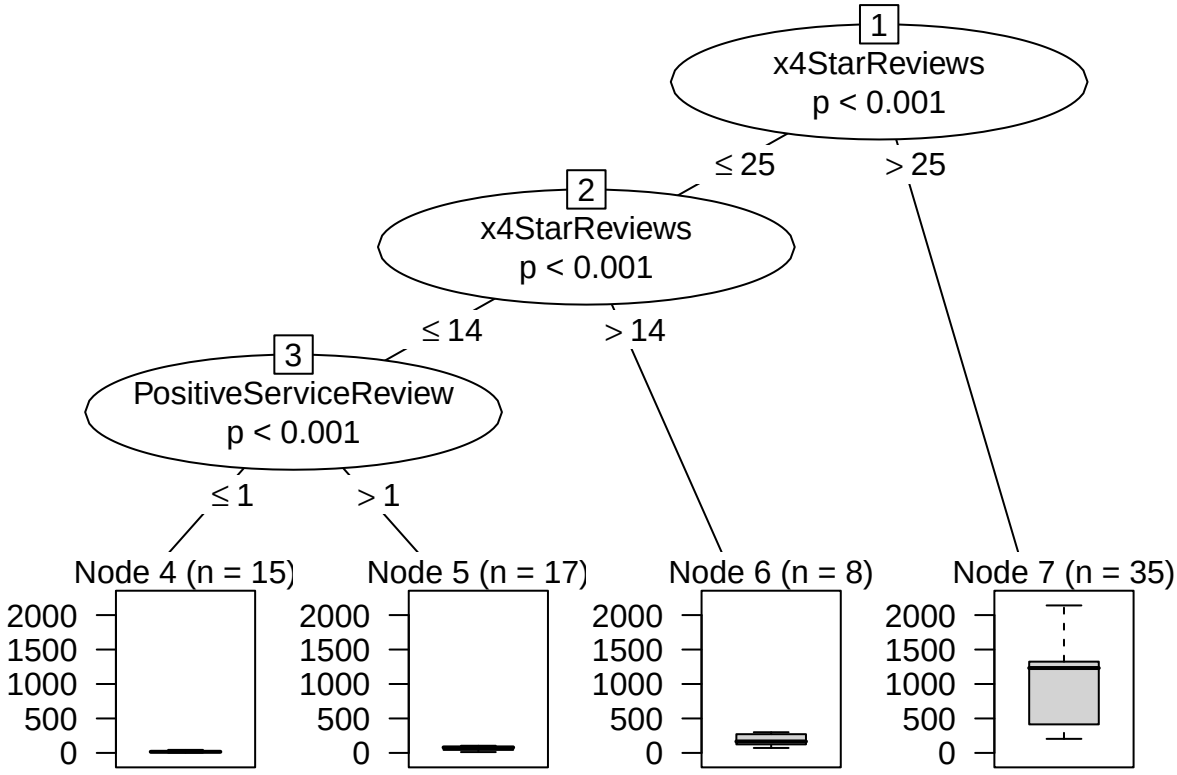
```
#by doing a random forest and running VarImp we can see also attributes worth keeping
fitControl <- trainControl(
    method = "repeatedcv", #repeated cross validation
    number = 10, #number of folds
    repeats = 3, #times repeated the cv
```

```
    classProbs = FALSE) #shows probability of the output

set.seed(123)
rffit <- train(Volume~.,
               data = existing_atributes2,
               method = "rf",
               trControl = fitControl,
               #preProcess = c("range"),
               verbose = FALSE,
               importance = TRUE)

#plot variable importance
rf_highly_correlated <- data.frame(c(varImp(rffit)))
high_cor_lab <- rownames(rf_highly_correlated)
par(mfrow=c(1,1), par(mar=c(11,3,2,2)))
barplot(rf_highly_correlated$Overall, names.arg = high_cor_lab, las = 2,
        main = 'Variable Importance on Sales Volume', col = brewer.pal(11,'Spectral'))
```



###Decision Tree 2; (with irrelecant attributes)

```
# Dummify product type and carry out correlations between all the variables for removing redundance.
Dummify2 <- dummyVars(" ~ .", data = existing_atributes2)
existing_atributes3Dum <- data.frame(predict(Dummify2, newdata = existing_atributes2))

# Decision Tree for knowing the relevant variables
tree <-ctree(Volume~., existing_atributes3Dum, control = ctree_control(maxdepth = 6))
plot(tree)
```



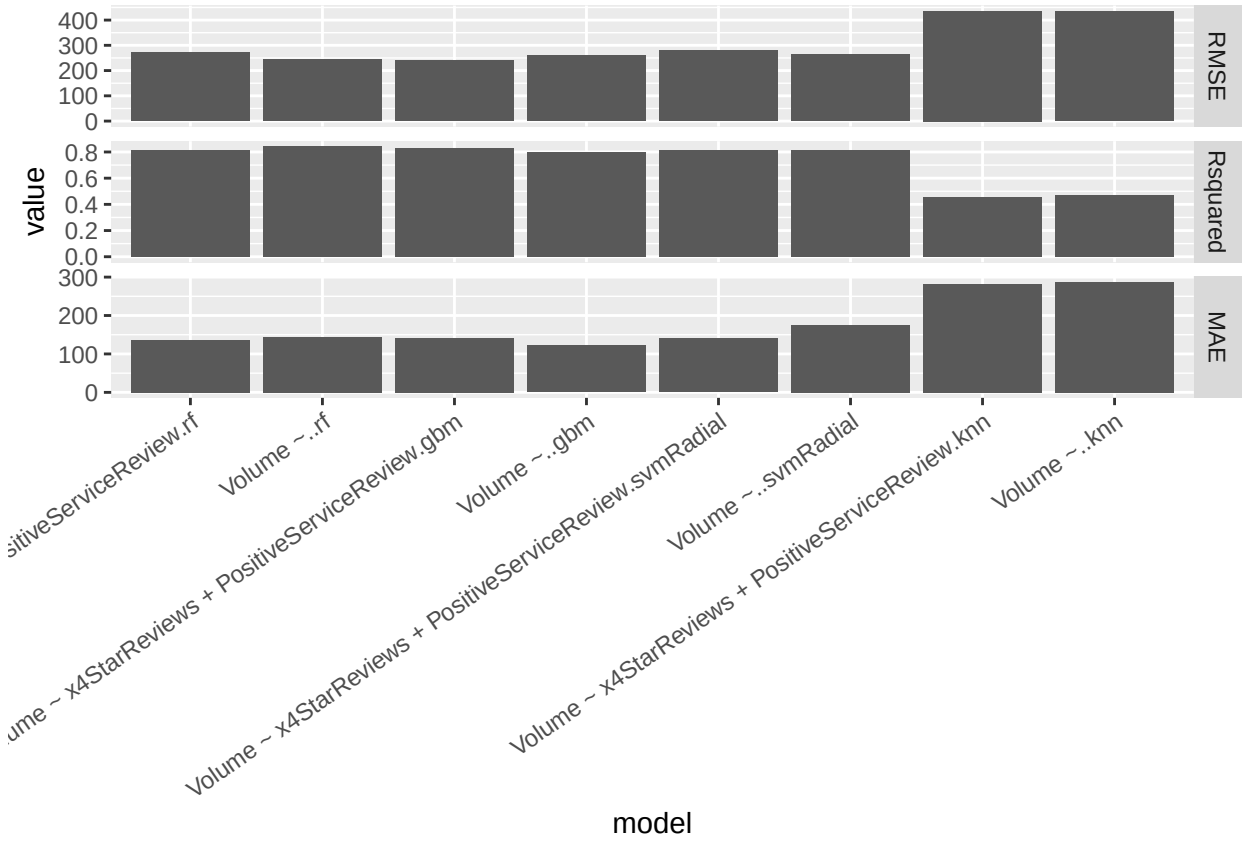
Training Machine Learning Algorithms

```
set.seed(123)
train_index <- createDataPartition(existing_atributes2$Volume,
                                   p = 0.7,
                                   list = FALSE,
                                   times = 1)

training <- existing_atributes2[train_index,]
testing <- existing_atributes2[-train_index,]
```

```
methods <- c('rf', 'gbm', 'svmRadial', 'knn')
formulae <- c('Volume ~ x4StarReviews + PositiveServiceReview', 'Volume ~.')
error_df <- c()
for(f in formulae){
    for(m in methods){
        model <- train(formula(f),
                       data = training,
                       method = m,
                       trControl = fitControl,
                       verbose = F)
        prediction <- predict(model, newdata = testing)
        errors <- postResample(testing$Volume, prediction)
        error_df <- cbind(error_df,errors)
    }
}
```

```
## note: only 1 unique complexity parameters in default grid. Truncating the grid to 1 .
```

```
ggplot(error_dfmelt, aes(x=model, y=value))+
 geom_col()+
 facet_grid(metric~., scales="free") + theme(axis.text.x = element_text(angle = 35, hjust = 1))
```



Predictions with new dataset

```
#import new dataset
new_attributes <- read.csv('newproductattributes20172.csv')
```

```
model_rf <- train(Volume~.,
                        data = training,
                        method = 'rf',
                        trControl = fitControl,
                        verbose = F)
prediction <- predict(model_rf, newdata = new_attributes)
new_attributes$PredVolume <- prediction
```

```
f <- new_attributes[,c(1,19)] %>% arrange(desc(PredVolume))
pc_index <- which(f$ProductType=='PC')
laptop_index <- which(f$ProductType=='Laptop')
net_index <- which(f$ProductType=='Netbook')
sma_index <- which(f$ProductType=='Smartphone')

ind <- c(pc_index, laptop_index, net_index, sma_index)
f[ind,]%>% arrange(desc(PredVolume))
```

```
##    ProductType PredVolume
## 1      Netbook 1577.38573
## 2   Smartphone  581.80453
## 3           PC  445.55240
## 4   Smartphone  349.80493
## 5       Laptop  273.27480
## 6           PC  212.91320
## 7   Smartphone  133.75187
## 8      Netbook  116.54840
## 9   Smartphone   74.54840
## 10     Netbook   39.30853
## 11      Laptop   29.58147
## 12      Laptop   16.95280
## 13     Netbook   16.04707
```