SHOP SALES DATA ANALYTICS

Project report submitted by

K.MANPREETH SAI
B130400CS

in partial fulfilment for the award of the Degree of

Bachelor of Technology
in
Computer Science and Engineering

under the guidance of

Dr.S D Madhu Kumar
(Associate Professor, CSED, NITC)

Department of Computer Science and Engineering
National Institute of Technology Calicut
NIT Campus P.O, Calicut
Kerala, India 673601

May 3, 2017

ACKNOWLEDGEMENT

We would like to extend sincere thanks to our project guide Dr.S D Madhu Kumar without whose indelible support our efforts would not have materialized. We would like to express our deep gratitude to our evaluation panel members Dr.T A Sumesh, Ms. Febna and Ms.Shruthi Anand for their valuable feedback at every stage of the project. We also thank Ms.Anu Mary Chacko our faculty advisor and project coordinator for huge support and valuable suggestions regarding project.

CERTIFICATE

This is to certify that the project entitled: “Shop Sales Data Analytics” *submitted by* Mr.K.Manpreeth Sai (B130400CS) *to National Institute of Technology Calicut towards partial fulfilment of the requirements for the award of Degree of Bachelor of Technology in Computer Science Engineering is a bonafide record of the work carried out by them under my supervision and guidance.*

Dr.S.D Madhu Kumar
(Project Guide)

Ms.Anu Mary Chacko
(Course Coordinator)

Dr.Saidalavi Kaladi
(CSED HOD)

Date:

DECLARATION

We hereby declare that this submission is our own work and that,to the best of our knowledge and belief,it contains no material previously published or written by another person nor material which has been accepted for the award of any other degree or diploma of the university or other institute of higher learning,except where due acknowledgement has been made in the text.

Place: NIT Calicut
Date : May 3, 2017

Signature:

Name : K Manpreeth Sai
Roll No : B130400CS

Abstract

The marketing space is in constant shift as new technologies and marketing tactics gain popularity. Hence, for faster and better decision making, **Data Analytics** has garnered significant attention.This project aims at improving the business strategies of **Retail Markets .i.e small datasets** dealing with perishable/non-perishable items. We develop predictive models using mining techniques and packages in R which reports,graphs and analyzes the pulse of our business by taking into consideration the required metrics.The models for quantity prediction and price fixing depending on sales performance metrics of storage,product revenue etc are delivered.

For the second part of this project, we take on the challenge of eradicating issues of seasonal changes, scale of the problem etc.. by **sales forecasting** and attempt to correctly forecast sales at Walmart. Given the reputation Walmart has about its competitive pricing structure, the ability to accurately project sales is key in its ability to function.This has been addressed by leveraging sales data from 45 Walmart stores across different regions. With this data I was able to make predictions on **department-wide sales** at each of the 45 stores.In addition to this,it has been attempted to understand the impact of markdowns (price reductions) on holiday weeks.However, it is important to note that while we have data for each of the 45 stores regarding department-wide sales, we will be modeling the effect of markdowns without possessing complete historical data. Overall, we hope to understand which attributes significantly impact sales at the store level via regression, time series analysis, and decision tree models so that we can ascern those characteristics as key drivers for sales, thus allowing us to generate more accurate forecasts.

Contents

List of Figures

Chapter 1

INTRODUCTION

1.1 DOMAIN

Hindsight. Insight. Foresight.

Data Analytics is the practice of using data to drive business strategy and performance. It includes a range of approaches and solutions, from looking backward to evaluate what happened in the past to looking forward to do **sales forecasting** and **predictive modelling**. These predictive models are implemented on data using R tools.

Cost reduction
BIG DATA ANALYTICS
Faster, better decision making
New products & services

Figure 1.1: Venn Diagram

1.2 KEYWORDS

- **Analysis:** The process of collecting, processing and analyzing data involving software-based analysis using algorithms.
- **Scenario Planning:** Scenarios are inductive narratives of the world at some future point in time. Unlike forecasting, scenario planning is not intended to give a definitive prediction of the future, but rather to communicate a wide range of possible outcomes and reacting sensitively to uncertainties in development.
- **Sales Forecasting:** Sales forecasting is the process of estimating future sales. Accurate sales forecasts enable companies to make informed business decisions and predict short-term and long-term performance. Companies can base their forecasts on past sales data, industry-wide comparisons, and economic trends.
- **Predictive Modeling:** Predictive modeling, an area of data mining, is a process to create a statistical model of future behavior.A predictive model is made up of a number of predictors, which are variable factors that are likely to influence future behavior.
- **R:** A popular open source software environment used for analytics.

1.3 MOTIVATION

Sales and marketing can be a guessing game. Retailers come up with strategies and put them to work, never certain whether their efforts are paying off or not.However, **Predictive analytics has given teams the ability to see for themselves exactly what their strategies are doing** and also retailers have to protrude with impending strategies reacting quickly.This is where **Data analytics** has garnered significant attention.Using data analytics, they can monitor customer activities and tie them to specific sales effort.

1.4 PROBLEM STATEMENT

As this project depicts a look from retailers point of view, the problem statement can be given as: To design and develop a decision support system for store management with the following capabilities:

- A model to find how to fix the price of an item
- A model to judge how much to store per item
- To forecast sales across each department of a store fetching help in business implications.

Chapter 2

DESCRIPTION OF DATA

2.1 DATA SOURCE

The input dataset for first two items in the problem statement is taken as the retail supermarket stores dataset [1] ,here after referred to as datasource-1, which simply contains attributes affecting sales.For the third problem statement, The Walmart Store Sales data [1], here after referred to as datasource-2, is considered. It covers historical sales data for 45 Walmart stores in different regions whose brief outlook is given below.

2.2 DATASETS OF DATASOURCE-2

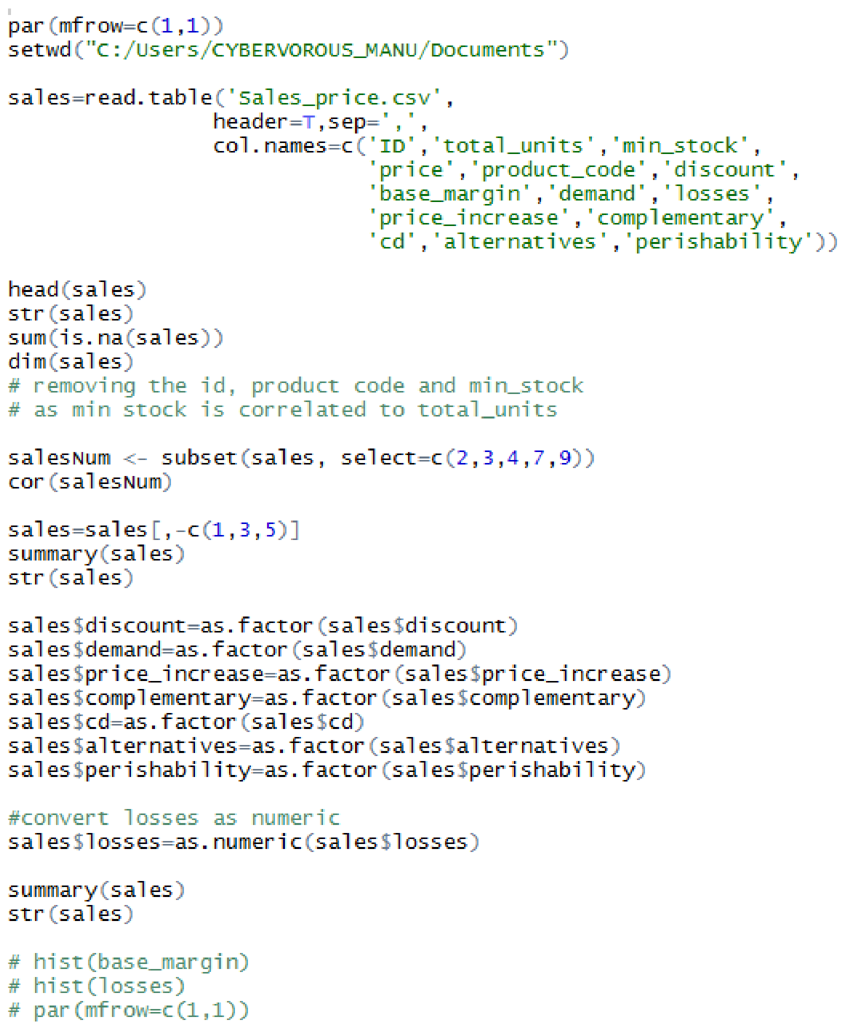
With the Walmart sales data, the following datasets are included:

- **stores.csv:** This file contains anonymized information about the 45 stores, indicating the type and size of store.
- **train.csv:** This is the historical training data, which covers to 2010-02-05 to 2012-11-01. Within this file we have the following fields:
  - Store - the store number
  - Dept - the department number
  - Date - the week
  - Weekly-Sales - sales for the given department in the given store
  - IsHoliday - whether the week is a special holiday week

- **test.csv:** This file is identical to train.csv, except we have withheld the weekly sales. We must predict the sales for each triplet of store, department, and date in this file.
- **features.csv:** This file contains additional data related to the store, department, and regional activity for the given dates. It contains the following fields:
  - Store - the store number
  - Date - the week
  - Temperature - average temperature in the region
  - Fuel-Price - cost of fuel in the region
  - MarkDown1-5 - anonymized data related to promotional markdowns that Walmart is running. MarkDown data is only available after Nov 2011, and is not available for all stores all the time. Any missing value is marked with an NA.
  - CPI - the consumer price index
  - Unemployment - the unemployment rate
  - IsHoliday - whether the week is a special holiday week

2.3 PRE-PROCESSING OF DATASOURCE-1

The datasource-1 to be used in predictive modelling for problem statements 1 and 2 is initially pre-processed and transformed to required format which involves numerical-to-categorical conversion, Dimensionality reduction and Correlation aspects. The R code implementation involving Data cleaning, mining is given as follows in fig 2.1:



```r
par(mfrow=c(1,1))
setwd("C:/Users/CYBERVOROUS_MANU/Documents")

sales=read.table('Sales_price.csv',
                 header=T,sep=',',
                 col.names=c('ID','total_units','min_stock',
                             'price','product_code','discount',
                             'base_margin','demand','losses',
                             'price_increase','complementary',
                             'cd','alternatives','perishability'))

head(sales)
str(sales)
sum(is.na(sales))
dim(sales)
# removing the id, product code and min_stock
# as min stock is correlated to total_units

salesNum <- subset(sales, select=c(2,3,4,7,9))
cor(salesNum)

sales=sales[,-c(1,3,5)]
summary(sales)
str(sales)

sales$discount=as.factor(sales$discount)
sales$demand=as.factor(sales$demand)
sales$price_increase=as.factor(sales$price_increase)
sales$complementary=as.factor(sales$complementary)
sales$cd=as.factor(sales$cd)
sales$alternatives=as.factor(sales$alternatives)
sales$perishability=as.factor(sales$perishability)

#convert losses as numeric
sales$losses=as.numeric(sales$losses)

summary(sales)
str(sales)

# hist(base_margin)
# hist(losses)
# par(mfrow=c(1,1))
```

Figure 2.1: Pre-Processing in R

Chapter 3

LITERATURE SURVEY

There has been significant research conducted on decision support system helping sales management over the past few decades.A group of researchers have worked on the development and implementation of pricing decision support tools for retailers.Over the last decade, several software firms have introduced revenue management software to help retailers make pricing decisions.

PREDICTIVE MODELLING: To design predictive models for the problem statements, we need to gain insight into **general pattern of factors affecting the predictor** which is obtained from this article [2] and then the **process of predictive modelling** [3].This leads to develop a model of higher accuracy by taking into count the flaws in current ones.

From a literature survey on **"how retailers predict price/quantity"** [5] in super-markets ,it has been learnt that: **Retail price-based revenue management also focuses on promotion and markdown dynamic price optimization**.Supermarket's quantity ordered behavior differs across goods, and over time for many individual goods. Recent empirical studies of retailing behavior have revealed several regularities in retail pricing behavior.

- First, most retail price changes reflect changes in retail margins, rather than changes in wholesale prices.
- Second, most price reductions tend to be short-lived.Together these findings conform with the casual observation that sales, in the sense of temporary reductions in retail prices that are unrelated to costs, are an important aspect of retailer pricing behavior.

- Third, the magnitude and frequency of sales differs across types of goods.

There is an existing theoretical research [4] on sales that provides an explanation for some of these pricing and quantity prediction patterns. **One explanation found in this literature is that sales are a means to intertemporally price discriminate for goods that either are infrequently purchased, or that can be inventoried by consumers.**

Looking into the implementation part, the methodologies involved have been understood from the following references cited for ready reference: Classification trees in R [6], Random forests [7], Gradient Boosting eight, Multiple Linear Regression [9], Data binning [13],GBM [14].The knowledge obtained after going through these references has been clearly described in the respective sections of usage.

SALES FORECASTING: Under the area of Sales Forecasting, The knowledge about tuning the parameters of GBM obtained from [15], helps us to fine tune several parameters in Stochastic Gradient Boosting Tree models, such as n.trees, interaction.depth, shrinkage and n.minobsinnode (R gbm package terms). From the reference papers [10], [11] the knowledge about Time-series analysis, modelling to improve forecasts in retail merchandising have been obtained respectively.

Chapter 4

HIGH-LEVEL DESIGN

4.1 PREDICTIVE MODEL FOR PRICE-FIXING

4.1.1 INPUT

This predictive model for Price-Fixing is built upon Supermarket Sales dataset [1] cited below for reference.

4.1.2 APPROACH

Concentrating on the problem, A price prediction model that incorporates competitive factors can model different scenarios allowing a company to prepare for potential disruptions.

4.1.2.1 FACTORS AFFECTING PREDICTOR

Examining the dependency of attributes on target variable is done by Graphical analysis using R. **Generally, Competitive factors in pricing strategy include:**

- Price of alternatives
- Customer's behaviour
- Product demand
- Brand loyalty

The effect of competitor 's price changes on pricing depends on the sensitivity of customers to that product.
EX: Apple products - This strategy doesnot work

So,looking into personalization of prices with respect to:

- Customer 's actual needs group
- Customer 's actual value group
- Customer 's analytics for pricing

These descriptive analytics fed into predictive model determine optimal attainable price.
EX: Best-buy strategy has analytics to understand customer 's needs.

4.1.2.2 PROCESS OF PREDICTIVE MODELLING

Breaking down the process of predictive modelling [3] into stages:

- **Descriptive analysis on the Data:**

  We primarily build models based on Logistic Regression and Decision Trees. Most of the algorithms involve greedy approaches, which can **subset the number of features I need to focus on**. Of all these: Linear Regression, Logistic Regression, Decision Tree, SVM, Naive Bayes, KNN, K-Means, Random Forest Dimensionality Reduction Algorithms, Gradient Boost Adaboost

  the required set of algorithms can be applied for Data Exploration.

- **Data Treatment:**

  Restricting it to missing values treatment upon the extracted subset of features,

  - Create dummy flags for missing values
  - Impute missing values with mean/median
  - Impute missing values of categorical variable.

- **Data Modelling:**

  When doing predictive modelling, we come across two situations:

  - We need to fit a well-defined parametrised model to our data, so we require a learning algorithm which can find those parameters on a data set without over-fitting
  - We need a model which can predict our dependent variable as accurately as possible, so we need a learning algorithm which can automatically identify the structure, interactions, and relationships in the data

  For **situation-1**, lasso and elastic-net generalized linear models are a set of modern algorithms which work fast and avoid over-fitting automatically. They are available in the "glmnet" package in R.

  For **situation-2**, ensembles of decision trees (often known as **Random Forests [7]) or The Gradient Boosting (GBA) [8]** have been the most successful general-purpose algorithms in modern times. It is available in the "randomForest" package in R.

4.1.3 VERIFICATION OF CORRECTNESS

There are various methods to validate our model's performance, simplest being to divide our data set into Train and validate (ideally 70:30) and build model based on 70 percent of train data set. Now, cross-validate it using 30 percent of validate data set and evaluate the performance using evaluation metrics to check the accuracy.

4.1.4 METHODOLOGY ADOPTED

The implementation part of this model mainly involves the following concepts of data mining:

- **Random forest** is like a bootstrapping algorithm with Decision tree (CART) model. Say, we have 1000 observation in the complete population with 10 variables. Random forest tries to build multiple CART model with different samples and different initial variables. It will repeat the process (say) 10 times and then make a final prediction on each observation. Final prediction is a function of each prediction. This final prediction can simply be the mean of each prediction.For instance, it will take a random sample of 100 observations and 5 randomly chosen initial variables to build a CART model.

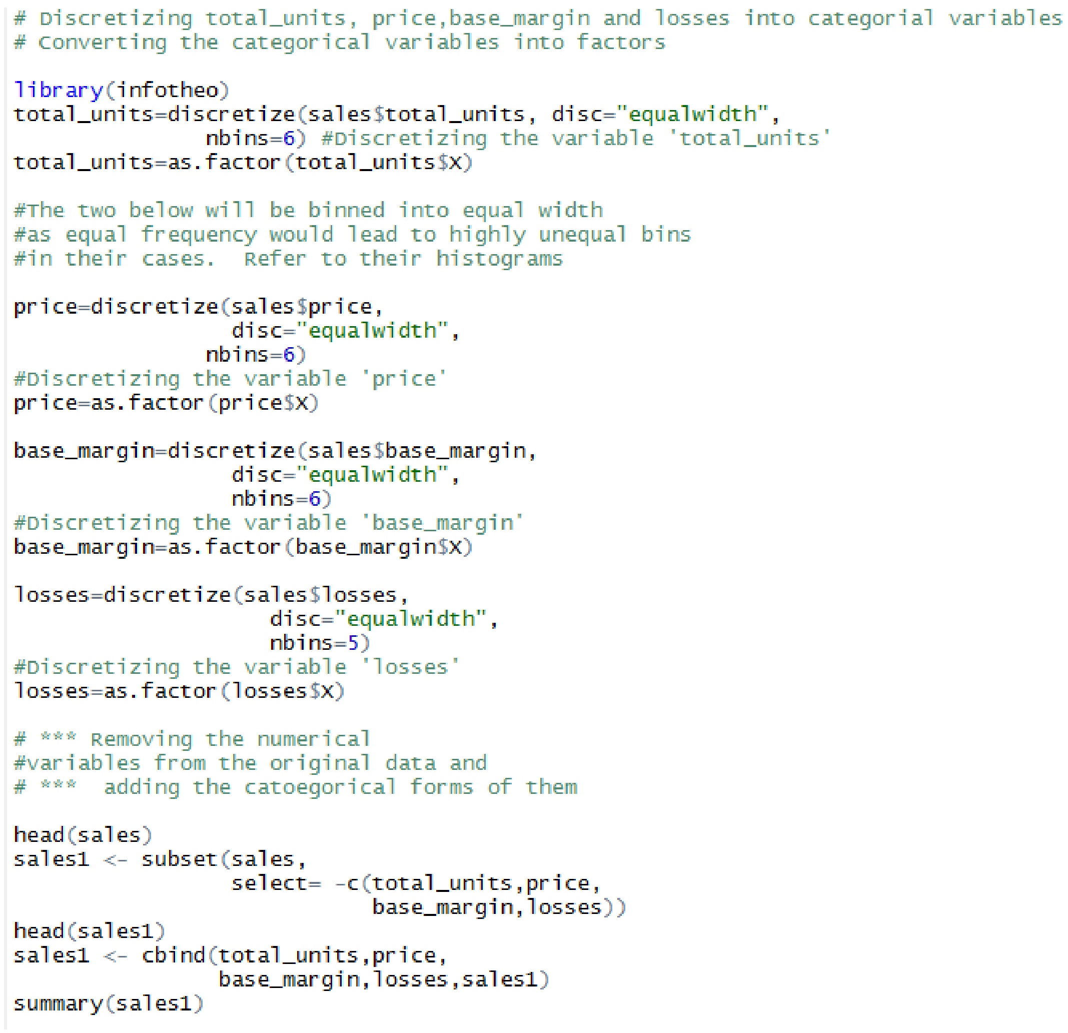
- **Gradient Boosting** The accuracy of a predictive model can be boosted by applying boosting algorithms straight away.There are multiple boosting algorithms like Gradient Boosting, XGBoost, AdaBoost, Gentle Boost etc. Every algorithm has its own underlying mathematics and a slight variation is observed while applying them.While working with boosting algorithms,we come across two frequently occurring buzzwords:

- **Binning** is the term used in scoring modeling for what is also known as Discretization, the process of transforming a continuous characteristic into a finite number of intervals (the bins), which allows for a better understanding of its distribution and its relationship with a binary variable.

4.1.4.1 PROTOTYPE

The prototype for the predictive model of price incrementation (or) decrementation is implemented in R where the attribute **price-change** is treated to be a categorical data with **0** indicating price incrementation and **1** for price decrementation.

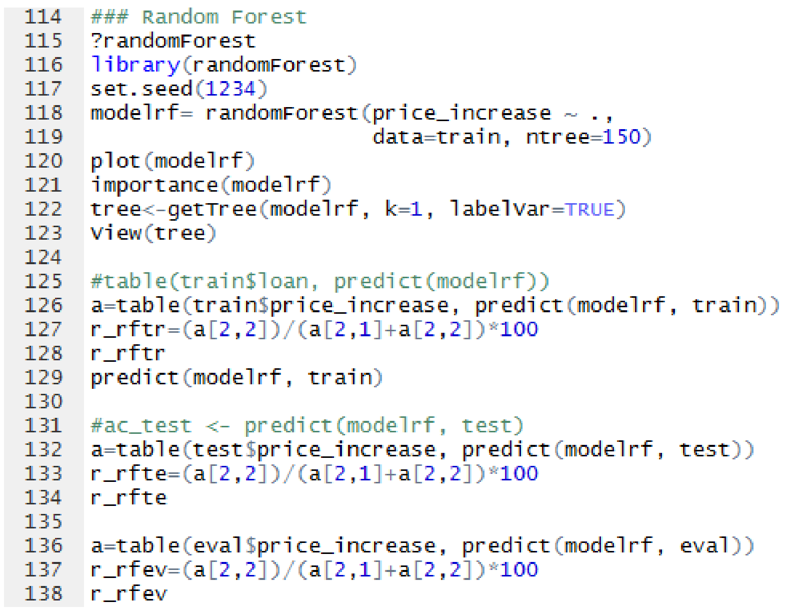
BINNING [13]: The prediction of price itself is not being done directly as **price** is a numerical data which needs to be handled differently.Rather than predicting price directly, we are trying to find out if initially a price increase/ decrease occurs which can later be extended to find out by what amount this takes place. The R code implementation involving Data discretization and binning of **price** into 6 bins is given as in fig 4.1:



```
# Discretizing total_units, price,base_margin and losses into categorial variables
# Converting the categorical variables into factors

library(infotheo)
total_units=discretize(sales$total_units, disc="equalwidth",
               nbins=6) #Discretizing the variable 'total_units'
total_units=as.factor(total_units$X)

#The two below will be binned into equal width
#as equal frequency would lead to highly unequal bins
#in their cases.  Refer to their histograms

price=discretize(sales$price,
                 disc="equalwidth",
               nbins=6)
#Discretizing the variable 'price'
price=as.factor(price$X)

base_margin=discretize(sales$base_margin,
                 disc="equalwidth",
                 nbins=6)
#Discretizing the variable 'base_margin'
base_margin=as.factor(base_margin$X)

losses=discretize(sales$losses,
                    disc="equalwidth",
                    nbins=5)
#Discretizing the variable 'losses'
losses=as.factor(losses$X)

# *** Removing the numerical
#variables from the original data and
# ***  adding the catoegorical forms of them

head(sales)
sales1 <- subset(sales,
                 select= -c(total_units,price,
                            base_margin,losses))
head(sales1)
sales1 <- cbind(total_units,price,
                base_margin,losses,sales1)
summary(sales1)
```

Figure 4.1: Binning in R

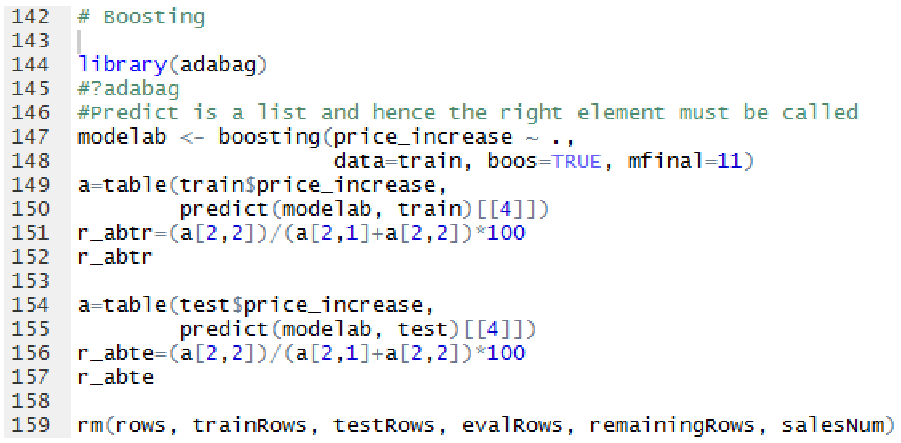
RANDOM FORESTS MODELLING: The predictive model determined using random forests is done by randomForest package in R as in fig 4.2:



```r
### Random Forest
?randomForest
library(randomForest)
set.seed(1234)
modelrf= randomForest(price_increase ~ .,
                      data=train, ntree=150)
plot(modelrf)
importance(modelrf)
tree<-getTree(modelrf, k=1, labelVar=TRUE)
View(tree)

#table(train$loan, predict(modelrf))
a=table(train$price_increase, predict(modelrf, train))
r_rftr=(a[2,2])/(a[2,1]+a[2,2])*100
r_rftr
predict(modelrf, train)

#ac_test <- predict(modelrf, test)
a=table(test$price_increase, predict(modelrf, test))
r_rfte=(a[2,2])/(a[2,1]+a[2,2])*100
r_rfte

a=table(eval$price_increase, predict(modelrf, eval))
r_rfev=(a[2,2])/(a[2,1]+a[2,2])*100
r_rfev
```

Figure 4.2: Random forests in R

BOOSTING ALGORITHMS: The obtained random forest model is inturn modelled using Gradient boosting algorithms to improve the accuracy as in fig 4.3 :



```r
# Boosting

library(adabag)
#?adabag
#Predict is a list and hence the right element must be called
modelab <- boosting(price_increase ~ .,
                    data=train, boos=TRUE, mfinal=11)
a=table(train$price_increase,
        predict(modelab, train)[[4]])
r_abtr=(a[2,2])/(a[2,1]+a[2,2])*100
r_abtr

a=table(test$price_increase,
        predict(modelab, test)[[4]])
r_abte=(a[2,2])/(a[2,1]+a[2,2])*100
r_abte

rm(rows, trainRows, testRows, evalRows, remainingRows, salesNum)
```

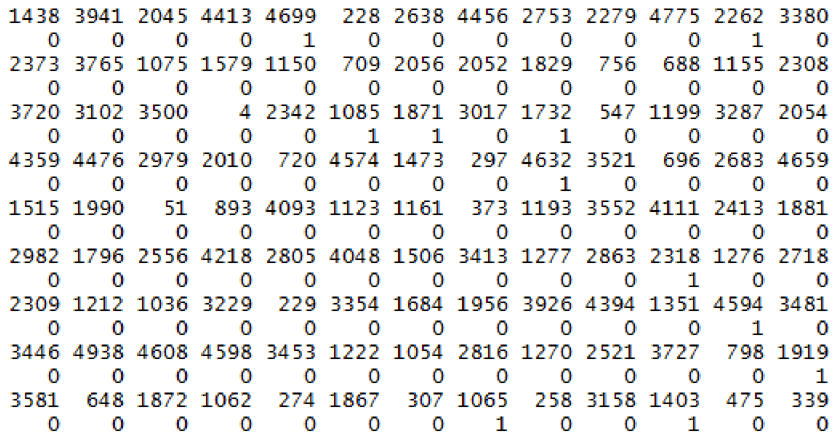
Figure 4.3: G-boosting in R

4.1.5 OUTPUTS

As the number of tuples are very large, the entire output can't be shown here. However, upon running the corresponding code mentioned in github, we can relate these results.

NOTE: The outputs are inter-releated with reference to the fact that they are just being represented in a different way

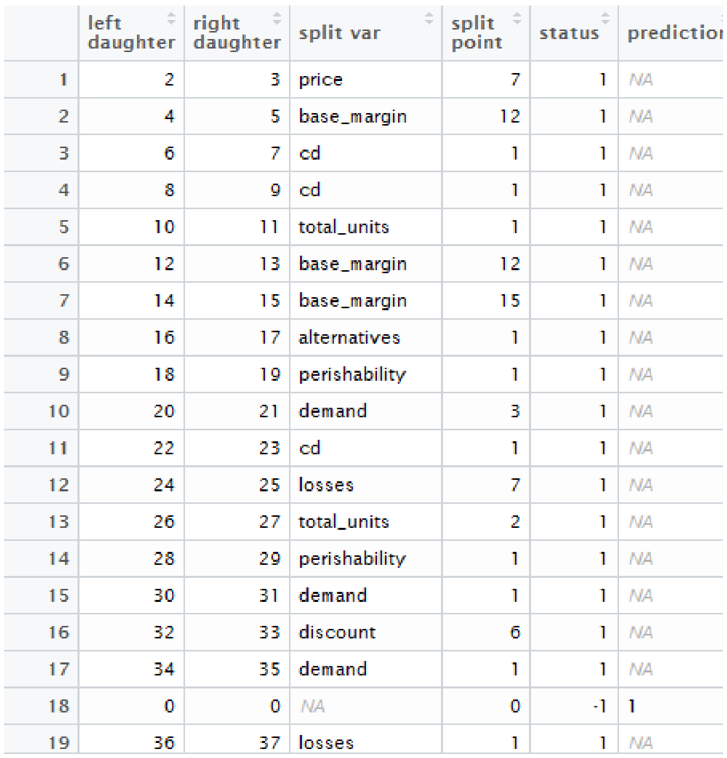
- The prediction of price-increase label to each tuple is obtained as in fig 4.4:



```
1438 3941 2045 4413 4699  228 2638 4456 2753 2279 4775 2262 3380
   0    0    0    0    1    0    0    0    0    0    0    1    0
2373 3765 1075 1579 1150  709 2056 2052 1829  756  688 1155 2308
   0    0    0    0    0    0    0    0    0    0    0    0    0
3720 3102 3500    4 2342 1085 1871 3017 1732  547 1199 3287 2054
   0    0    0    0    0    1    1    0    1    0    0    0    0
4359 4476 2979 2010  720 4574 1473  297 4632 3521  696 2683 4659
   0    0    0    0    0    0    0    0    1    0    0    0    0
1515 1990   51  893 4093 1123 1161  373 1193 3552 4111 2413 1881
   0    0    0    0    0    0    0    0    0    0    0    0    0
2982 1796 2556 4218 2805 4048 1506 3413 1277 2863 2318 1276 2718
   0    0    0    0    0    0    0    0    0    0    1    0    0
2309 1212 1036 3229  229 3354 1684 1956 3926 4394 1351 4594 3481
   0    0    0    0    0    0    0    0    0    0    0    1    0
3446 4938 4608 4598 3453 1222 1054 2816 1270 2521 3727  798 1919
   0    0    0    0    0    0    0    0    0    0    0    0    1
3581  648 1872 1062  274 1867  307 1065  258 3158 1403  475  339
   0    0    0    0    0    0    0    1    0    0    1    0    0
```

Figure 4.4: Predicted labels

- The tree obtained as a result of Random forest and boosting is described as fig 4.5:



	left daughter	right daughter	split var	split point	status	predictio
1	2	3	price	7	1	NA
2	4	5	base_margin	12	1	NA
3	6	7	cd	1	1	NA
4	8	9	cd	1	1	NA
5	10	11	total_units	1	1	NA
6	12	13	base_margin	12	1	NA
7	14	15	base_margin	15	1	NA
8	16	17	alternatives	1	1	NA
9	18	19	perishability	1	1	NA
10	20	21	demand	3	1	NA
11	22	23	cd	1	1	NA
12	24	25	losses	7	1	NA
13	26	27	total_units	2	1	NA
14	28	29	perishability	1	1	NA
15	30	31	demand	1	1	NA
16	32	33	discount	6	1	NA
17	34	35	demand	1	1	NA
18	0	0	NA	0	-1	1
19	36	37	losses	1	1	NA

Figure 4.5: Classification Tree

- The RMSE error improves upon repeated learning of historical data as the tree can classify accurately which can be shown by fig 4.6:

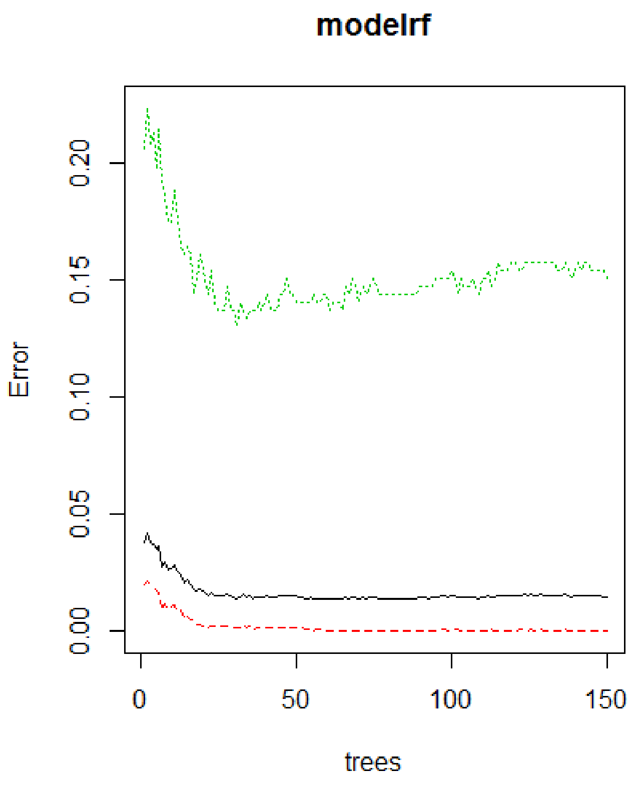


Figure 4.6: Perpetual Learning

- The accuracy of the model obtained upon running Random forests was **87.2093** which is later on improved upon running Gradient boosting over this leading to an accuracy of **90.19608**

4.2 PREDICTIVE MODEL FOR QUANTITY PREDICTION

4.2.1 INPUT

This predictive model for Quantity-Prediction is built upon Supermarket Sales dataset [1] cited below for reference.

4.2.2 APPROACH

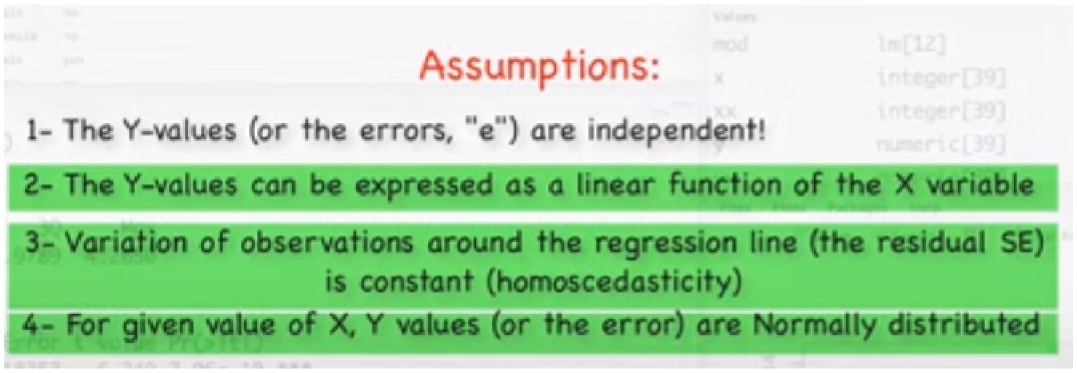
Concentrating on the problem, A quantity prediction model that incorporates competitive factors can model different scenarios allowing a company to prepare for potential disruptions.

- Identifying the factors affecting target variable generally such as price of alternatives, product demand etc,. can be included as parameters.
- Generally,the choice of attributes as parameters of model is based upon criterion of p-values generated by MLR model.
- Following the process of Predictive Modelling as mentioned earlier in Price-fixing model and verifying the correctness, a model of optimal accuracy can be obtained by perpetual learning of model.

4.2.3 METHODOLOGY ADOPTED

The implementation part of this model mainly involves the following concepts of data mining:

- **Multiple Linear Regression** [9] We will now fit a (multiple) linear regression, which is probably the best known statistical model. In simple terms, the dependent variable is assumed to be a linear function of several independent variables (predictors), where each of them has a weight (regression coefficient) that is expected to be statistically significant in the final model. The results are usually highly interpretable and, provided some conditions are met, have good accuracy.
- Usually, few assumptions are made while fitting a Multiple Linear Regression model as in fig 4.7:



Assumptions:

1- The Y-values (or the errors, "e") are independent!

2- The Y-values can be expressed as a linear function of the X variable

3- Variation of observations around the regression line (the residual SE) is constant (homoscedasticity)

4- For given value of X, Y values (or the error) are Normally distributed

Figure 4.7: Assumptions of MLR

- These assumptions can be validated by examining the residuals using 4 built-in diagnostic plots in R obtained by **plot(model-name)** command.
  - In this diagnostic plot-1, the red-line being fairly flat implies linearity assumption is met .i.e we can see no pattern over that region indicating variation is constant.

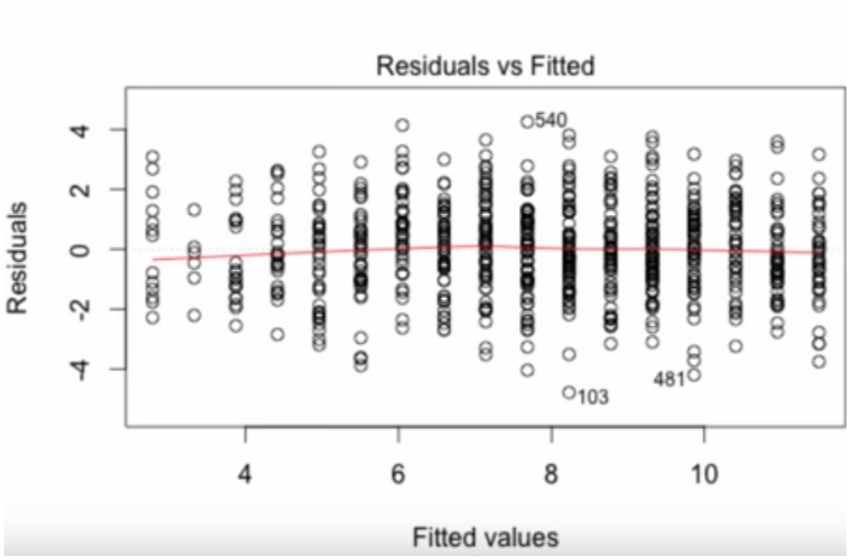


Figure 4.8: Diagnostic plot-1

- Plot-2(Quantile-Quantile plot) indicates that expected residuals lie in a straight line if the X, Y values are normally distributed validating assumption 4.

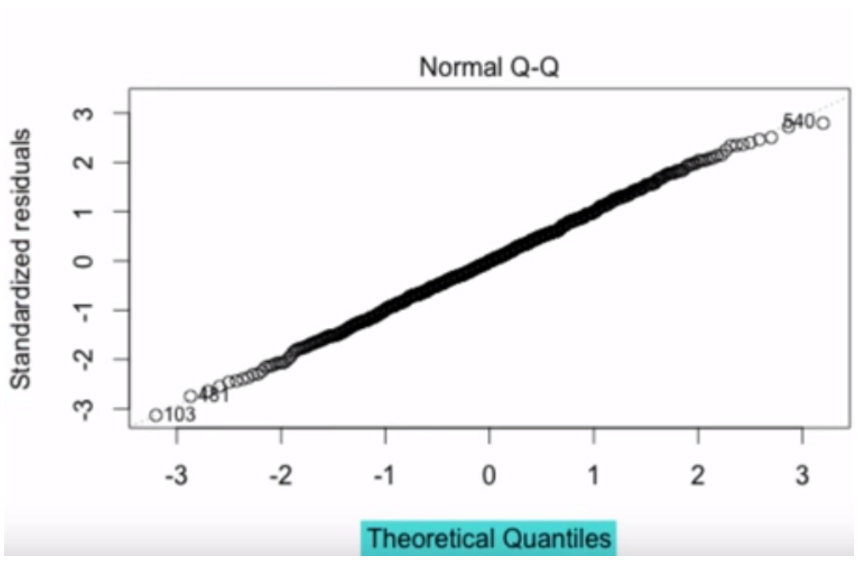


Figure 4.9: Diagnostic plot-2

- These two diagnostic plots helps us examine non-linearities, non-constant variations and trouble-some observations.

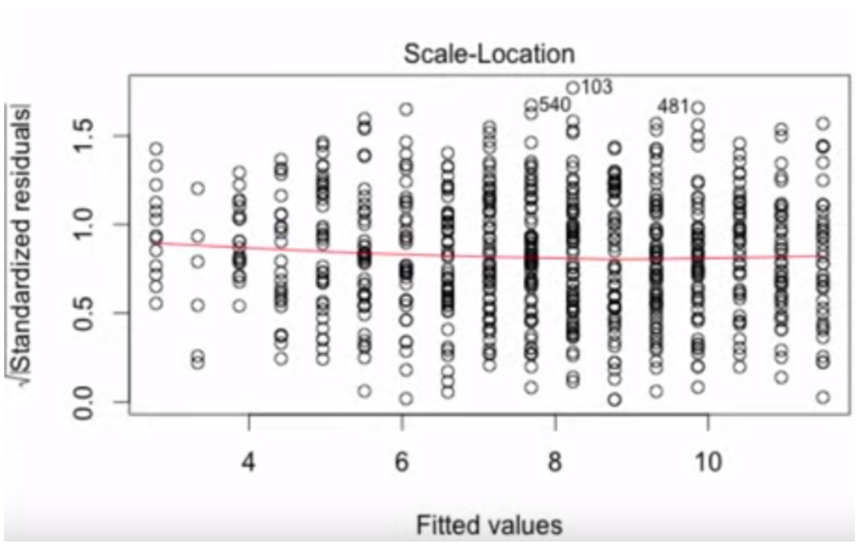


Figure 4.10: Diagnostic plot-3

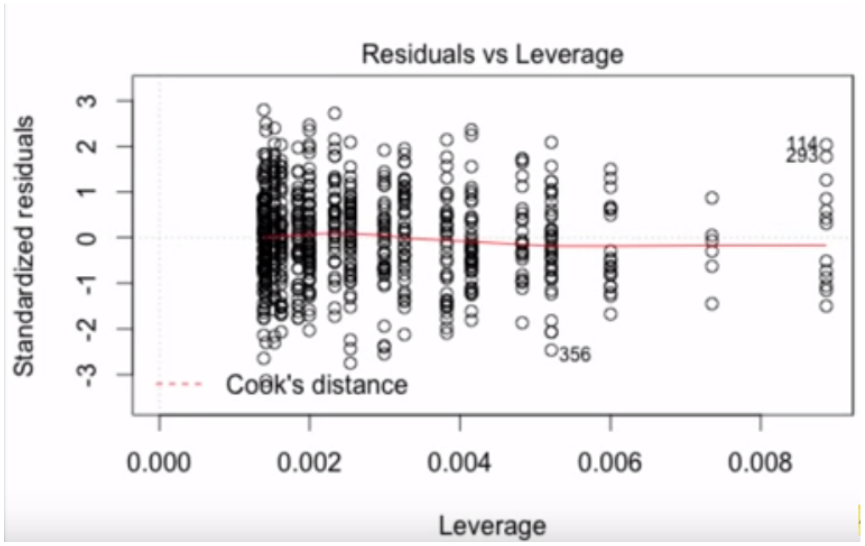


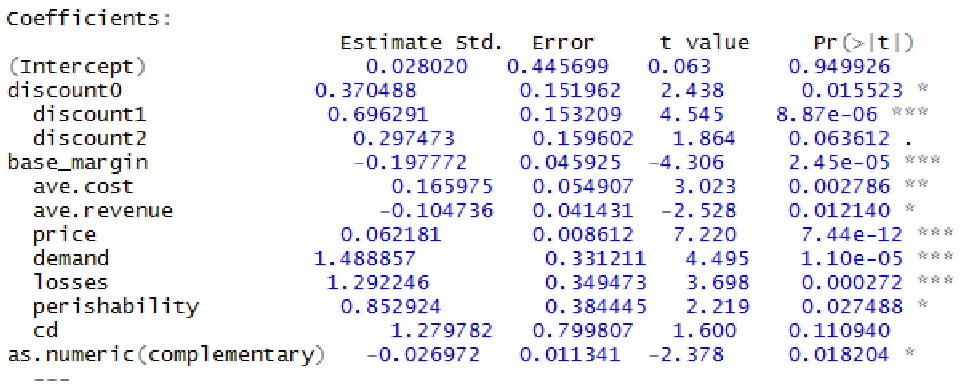
Figure 4.11: Diagnostic plot-4

- Using the concepts of Random forests, Binning, Gradient boosting described earlier would gradually increase the accuracy of the predicting model.

4.2.3.1 PROTOTYPE

The prototype for the quantity prediction model is implemented in R where the attribute 'quantity-ordered' **numerical data** is discretized into 6 bins for the prediction of bin-label for each tuple and assuming the ultimate value of predicted quantity to be a mean/median of the corresponding bin.The prediction of quantity itself is being done directly as 'quantity' is a numerical data which needs to be handled differently using Multiple Linear regression.

MULTIPLE LINEAR REGRESSION: Operating on pre-processed dataset, The predictive model determined using linear regresssion is done by GLM package in R as:



```
Coefficients:
                          Estimate Std.  Error     t value     Pr(>|t|)
(Intercept)                 0.028020   0.445699   0.063      0.949926
discount0                 0.370488       0.151962   2.438      0.015523 *
  discount1                0.696291      0.153209   4.545    8.87e-06 ***
  discount2                  0.297473     0.159602   1.864     0.063612 .
base_margin                 -0.197772    0.045925  -4.306     2.45e-05 ***
  ave.cost                     0.165975   0.054907   3.023     0.002786 **
  ave.revenue                 -0.104736   0.041431  -2.528     0.012140 *
  price                     0.062181      0.008612   7.220      7.44e-12 ***
  demand                  1.488857         0.331211   4.495     1.10e-05 ***
  losses                   1.292246        0.349473   3.698     0.000272 ***
  perishability             0.852924       0.384445   2.219     0.027488 *
  cd                           1.279782   0.799807   1.600     0.110940
as.numeric(complementary)   -0.026972    0.011341  -2.378     0.018204 *
  ---
```

Figure 4.12: Output of MLR

- **P-VALUE:** This indicates the probability value for the hypothesis test that the coefficients of corresponding attributes is zero. .i.e **lesser p-value is preferred**

- **CRITERION FOR CHOOSING PARAMS:** From the output obtained, all the attributes with higher p-values are eliminated and iteratively doing this, we arrive at the set of parameters that can be used for modelling.

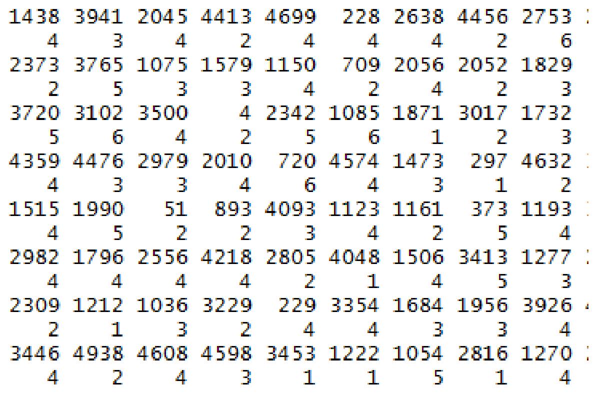
- The ultimate equation of regression-line is given as: **Y = (Estimate std.)*(Parameter) + Intercept** which can be inferred from the above table.

- This model upon undergoing through Random Forest and Gradient Boosting procedures as earlier gives the ultimate result.

4.2.4 OUTPUTS

As the number of tuples are very large, the entire output can't be shown here. However, upon running the corresponding code mentioned in github, we can relate these results.

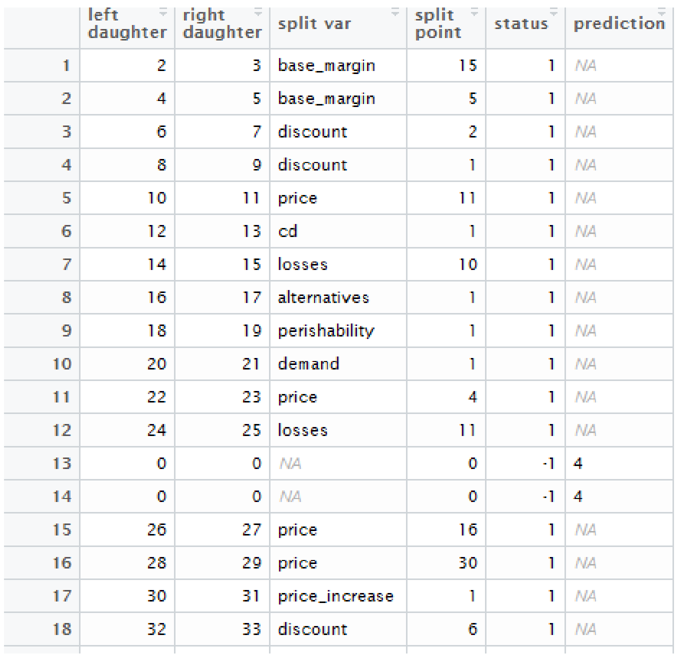
- The prediction of quantity bin-label to each tuple is obtained as in fig 4.13:



```
1438 3941 2045 4413 4699  228 2638 4456 2753
   4    3    4    2    4    4    4    2    6
2373 3765 1075 1579 1150  709 2056 2052 1829
   2    5    3    3    4    2    4    2    3
3720 3102 3500    4 2342 1085 1871 3017 1732
   5    6    4    2    5    6    1    2    3
4359 4476 2979 2010  720 4574 1473  297 4632
   4    3    3    4    6    4    3    1    2
1515 1990   51  893 4093 1123 1161  373 1193
   4    5    2    2    3    4    2    5    4
2982 1796 2556 4218 2805 4048 1506 3413 1277
   4    4    4    4    2    1    4    5    3
2309 1212 1036 3229  229 3354 1684 1956 3926
   2    1    3    2    4    4    3    3    4
3446 4938 4608 4598 3453 1222 1054 2816 1270
   4    2    4    3    1    1    5    1    4
```

Figure 4.13: Sample Predicted labels

- The tree obtained as a result of Random forest and boosting is described as in fig 4.14:



	left daughter	right daughter	split var	split point	status	prediction
1	2	3	base_margin	15	1	NA
2	4	5	base_margin	5	1	NA
3	6	7	discount	2	1	NA
4	8	9	discount	1	1	NA
5	10	11	price	11	1	NA
6	12	13	cd	1	1	NA
7	14	15	losses	10	1	NA
8	16	17	alternatives	1	1	NA
9	18	19	perishability	1	1	NA
10	20	21	demand	1	1	NA
11	22	23	price	4	1	NA
12	24	25	losses	11	1	NA
13	0	0	NA	0	-1	4
14	0	0	NA	0	-1	4
15	26	27	price	16	1	NA
16	28	29	price	30	1	NA
17	30	31	price_increase	1	1	NA
18	32	33	discount	6	1	NA

Figure 4.14: Classification Tree

- The RMSE error improves upon repeated learning of historical data as the tree can classify accurately which can be shown by 4.15:

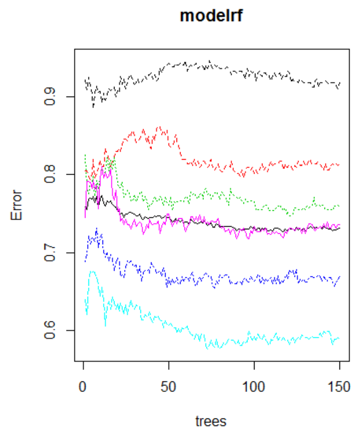


Figure 4.15: Perpetual Learning

- The gradual increase in the accuracy of the final model upon carrying out above algorithms is depicted as in fig 4.16:

```
# Predicted vs. actual for each model
ggplot(data = all.predictions,aes(x = actual, y =
predictions)) +
  geom_point(colour = "blue") +
  geom_abline(intercept = 0, slope = 1, colour = "red") +
  geom_vline(xintercept = 23, colour = "green", linetype =
"dashed") +
  facet_wrap(~ model,ncol = 2) +
  coord_cartesian(xlim = c(0,70),ylim = c(0,70)) +
  ggtitle("Predicted vs. Actual, by model")
```

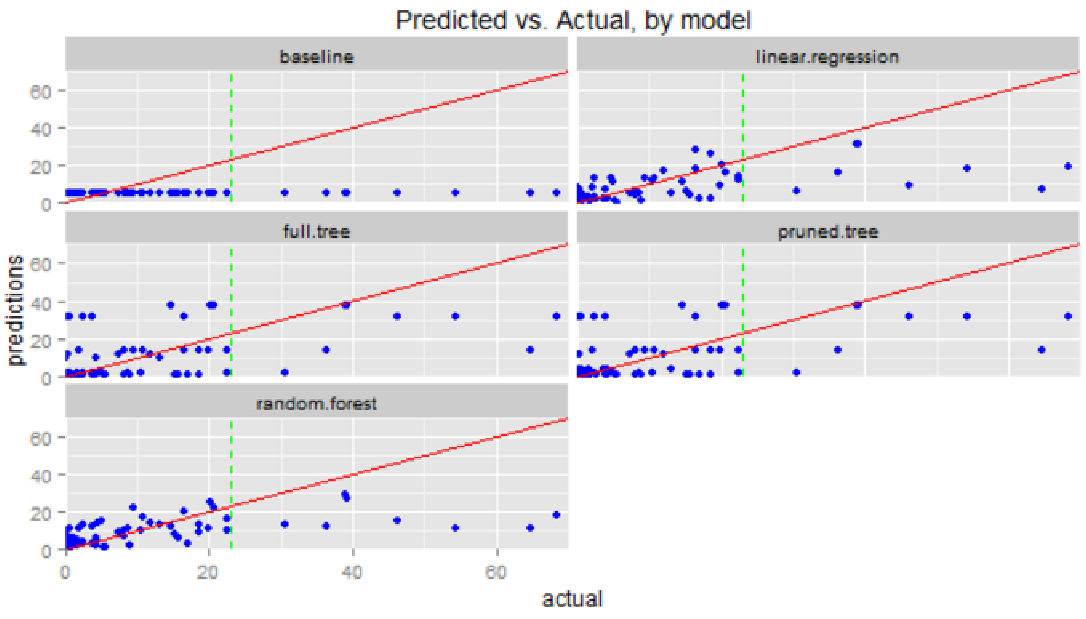


Figure 4.16: Improving Accuracies

4.3 SALES FORECASTING

4.3.1 INPUT

This section deals with sales forecasting to predict weekly sales across each department operating on Data source-2 [1] cited for reference.

4.3.2 ABSTRACTION OF APPROACH

Supervised learning: Decision trees using Random forests

- **3-way splitting:** By discretizing the weekly-sales into three classes of Low, Medium, High and proceeding with the approaches mentioned above; (This can be used as an overview of weekly-sales prediction .i.e this simply gives the class to which the weekly-sales belong to using tree but cannot give the exact value of sales occured.)

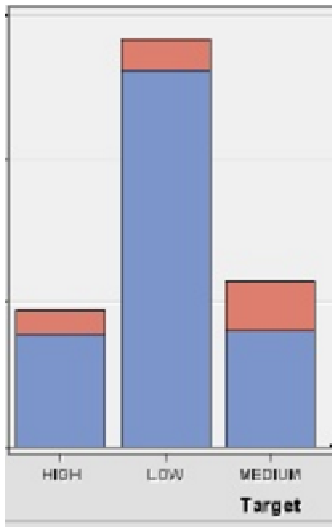
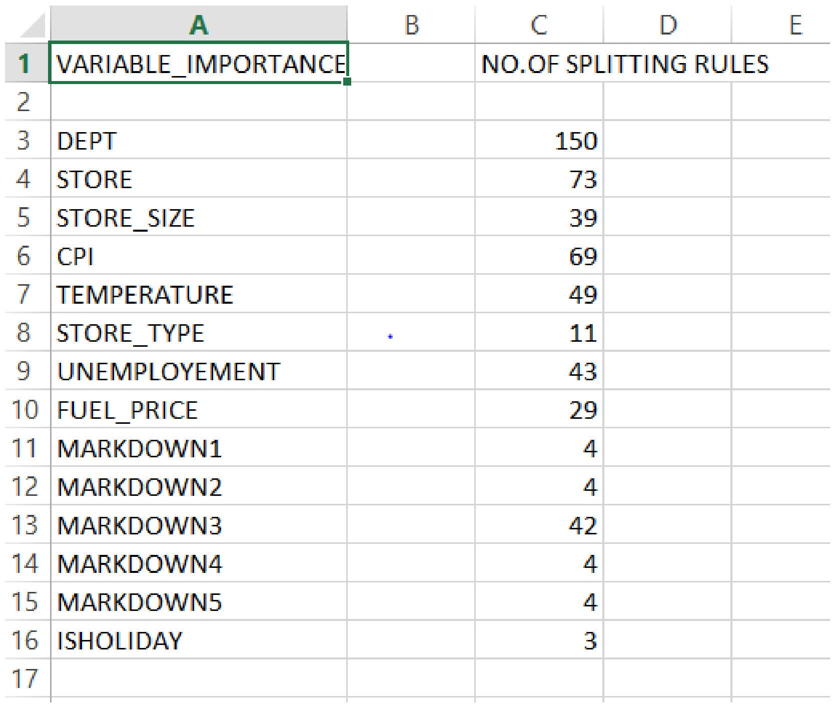


Figure 4.17: Weekly-sales-class vs No.of tuples

- A three split decision tree can be generated on the train dataset using random forests as in price-prediction case.The weekly sales classes predicted earlier stand out as target classes.

- Similar to price prediction, a tree table will be generated giving out the attribute for split criterion at each node. As there are 10,000 tuples, my depth of the tree would be very high. So, by using excel sheet, just grouping up how many times each attribute was used for split criterion; we obtain the following table:



	A	B	C	D	E
1	VARIABLE_IMPORTANCE		NO.OF SPLITTING RULES		
2					
3	DEPT		150		
4	STORE		73		
5	STORE_SIZE		39		
6	CPI		69		
7	TEMPERATURE		49		
8	STORE_TYPE		11		
9	UNEMPLOYEMENT		43		
10	FUEL_PRICE		29		
11	MARKDOWN1		4		
12	MARKDOWN2		4		
13	MARKDOWN3		42		
14	MARKDOWN4		4		
15	MARKDOWN5		4		
16	ISHOLIDAY		3		
17					

Figure 4.18: Criterion count of attributes

- This says that the model is heavily dependent on department (DEPT) and store (STORE). Majority of the splitting rules were based on these two attributes.

- However, the three-way split provided more flexibility to the model in terms of decision making and hence the errors in classifying them into the weekly sales classes, were less as expected. The 3-way split decision tree generated an average square error of 0.027.

4.3.2.1 SIGNIFICANCE OF MODELS ACROSS EACH DEPARTMENT:

Explaining why a model [10] for departments across each store, it says that target depends heavily on Store, dept attributes. Hence, we can try bringing out models (GBM models) specifically for weekly-sales across each combination of (store + dept) so that this leads to a level deeper into the abstraction given by trees and comes handy in prediction of weekly-sales accurately .This output can be explained by my **final dataset** which gives average weekly-sales for every id (where id= store + dept + weekdate). All the graphs of date-index (vs) weekly-sales are called time series analysis.

4.4 METHODOLOGY ADOPTED

This proceeds with the use of a machine learning approach. We can draw an observation that for the same department, the weekly sales (or the sales pattern) are very similar despite of different magnitudes across all the stores. So, it might seem that the same department sells the same kind of products.

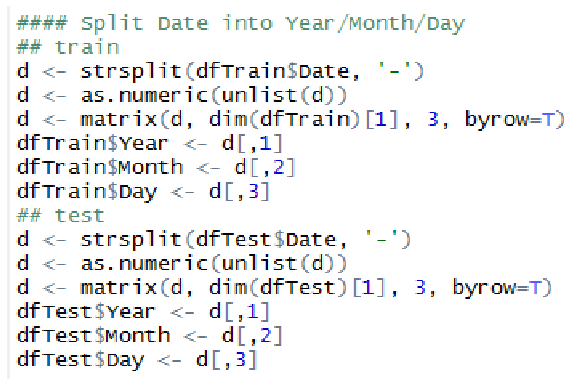
For each department, we can train a GBM (Gradient Boosted Regression Models) across all the stores available in the training data. Using all the raw features (given in features.csv) and additionally generating a few more which deem intuitive and useful is necessary.

4.4.1 FEATURE EXTRACTION

- Generating features is basically done by operating upon the datasets train.csv, test.csv, features.csv and the resulting useful datasets include attributes collectively from train.csv, features.csv together with newly generated attributes.

- Applying few operations upon current attributes leads to extraction of new attributes which turn out to be handy in feature extraction.

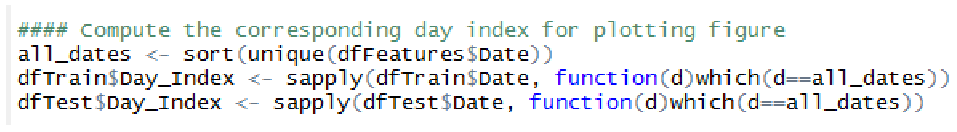
- Here, this idea servers the purpose of obtaining date, month, year separately which can be used to generate Number of days to next markdown event and other interesting features. Corresponding R code implementations are given as in fig 4.19:



```
#### Split Date into Year/Month/Day
## train
d <- strsplit(dfTrain$Date, '-')
d <- as.numeric(unlist(d))
d <- matrix(d, dim(dfTrain)[1], 3, byrow=T)
dfTrain$Year <- d[,1]
dfTrain$Month <- d[,2]
dfTrain$Day <- d[,3]
## test
d <- strsplit(dfTest$Date, '-')
d <- as.numeric(unlist(d))
d <- matrix(d, dim(dfTest)[1], 3, byrow=T)
dfTest$Year <- d[,1]
dfTest$Month <- d[,2]
dfTest$Day <- d[,3]
```

Figure 4.19: Splitting Date

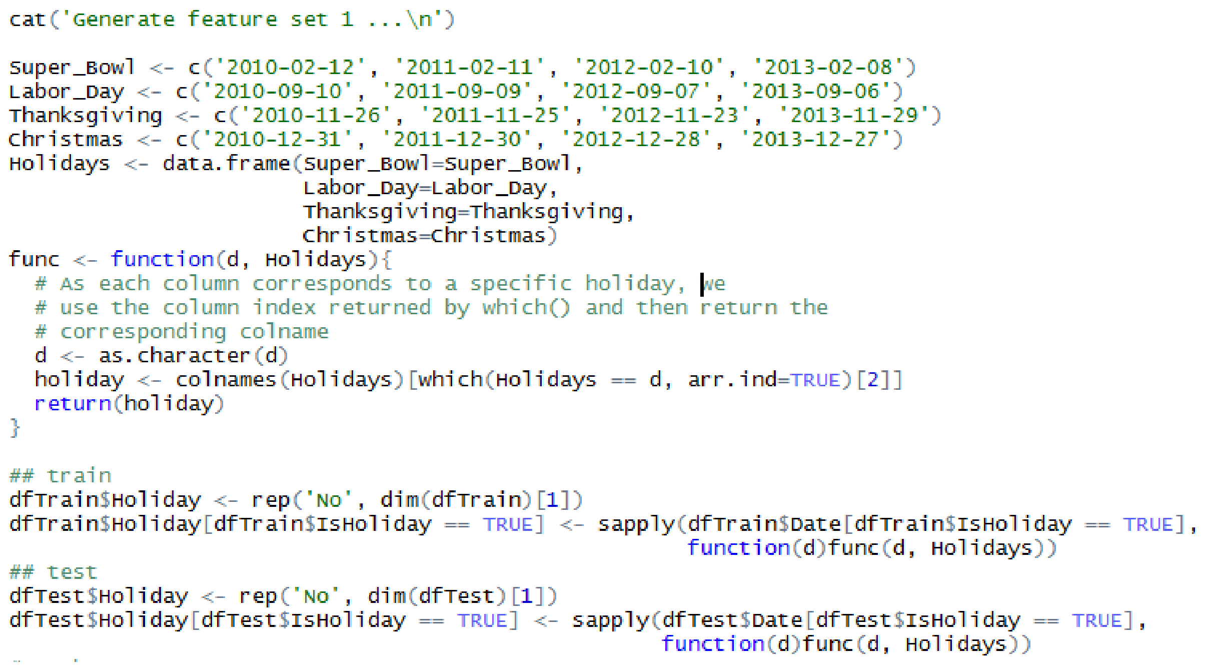
- The ultimate Time series analysis graph is plot between date-index (vs) weekly-sales where **date-index refers to rank of the date after sorting all dates**.Corresponding R code implementations are given as in fig 4.20:



```
#### Compute the corresponding day index for plotting figure
all_dates <- sort(unique(dfFeatures$Date))
dfTrain$Day_Index <- sapply(dfTrain$Date, function(d)which(d==all_dates))
dfTest$Day_Index <- sapply(dfTest$Date, function(d)which(d==all_dates))
```

Figure 4.20: Date-index

- The first feature set is generated to specify for those stores with attribute IsHoliday=1, we further indicate *which of the four holidays they are* Super-bowl, Labour-day, Thanks-giving, Christmas.Corresponding R code implementations are given as in fig 4.21:



```
cat('Generate feature set 1 ...\n')

Super_Bowl <- c('2010-02-12', '2011-02-11', '2012-02-10', '2013-02-08')
Labor_Day <- c('2010-09-10', '2011-09-09', '2012-09-07', '2013-09-06')
Thanksgiving <- c('2010-11-26', '2011-11-25', '2012-11-23', '2013-11-29')
Christmas <- c('2010-12-31', '2011-12-30', '2012-12-28', '2013-12-27')
Holidays <- data.frame(Super_Bowl=Super_Bowl,
                       Labor_Day=Labor_Day,
                       Thanksgiving=Thanksgiving,
                       Christmas=Christmas)
func <- function(d, Holidays){
  # As each column corresponds to a specific holiday, we
  # use the column index returned by which() and then return the
  # corresponding colname
  d <- as.character(d)
  holiday <- colnames(Holidays)[which(Holidays == d, arr.ind=TRUE)[2]]
  return(holiday)
}

## train
dfTrain$Holiday <- rep('No', dim(dfTrain)[1])
dfTrain$Holiday[dfTrain$IsHoliday == TRUE] <- sapply(dfTrain$Date[dfTrain$IsHoliday == TRUE],
                                                     function(d)func(d, Holidays))
## test
dfTest$Holiday <- rep('No', dim(dfTest)[1])
dfTest$Holiday[dfTest$IsHoliday == TRUE] <- sapply(dfTest$Date[dfTest$IsHoliday == TRUE],
                                                   function(d)func(d, Holidays))
```

Figure 4.21: Generating feature-set 1

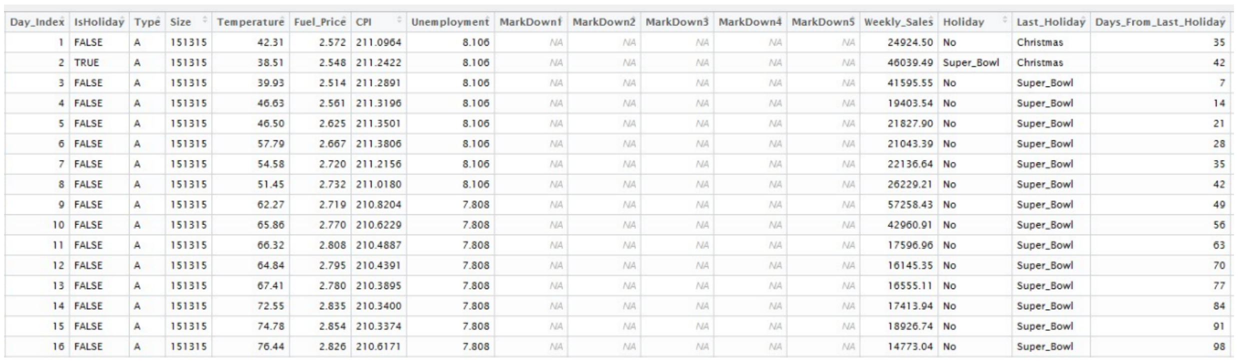
- Before a holiday comes, more and more people are going shopping, and thus the weekly sales before/during holiday weeks increase. And after that, weekly sales begin to fall.To analyze this observation, feature set-2 is generated which contains *What is the last/next holiday? How many days are there from/to that?* Corresponding R code implementations are given as in fig 4.22:

```r
cat('Generate feature set 2 ...\n')

HolidayTimeLine <- rbind(Super_Bowl,Labor_Day,Thanksgiving,Christmas)
HolidayTimeLine <- unlist(HolidayTimeLine)
# Since the most front holiday in the training data is '2010-02-12', so we have to
# insert the date of Christmas in 2009 to deal with it
Christmas_2009 <- as.character(as.Date('2010-12-31')-364)
# to know how to get 364, check: diff(as.Date(Christmas))/diff(as.Date(Labor_Day))
HolidayTimeLine <- c(Christmas_2009, HolidayTimeLine)
holiday_names <- c('Super_Bowl', 'Labor_Day', 'Thanksgiving', 'Christmas')
holiday_names <- c('Christmas', rep(holiday_names, 4))
# convert to Date class
HolidayTimeLine <- as.Date(HolidayTimeLine)

func <- function(d, HolidayTimeLine, holiday_names){
  # find the closest date to d in HolidayTimeLine
  dif <- as.numeric(d-HolidayTimeLine)
  ind <- which.min(abs(dif))
  # d comes after the closest date
  if(dif[ind] > 0){
    last_holiday_ind <- ind
  }else{
    last_holiday_ind <- ind - 1
  }
  last_holiday <- holiday_names[last_holiday_ind]
  days_from_last_holiday <- abs(dif[last_holiday_ind])
  next_holiday <- holiday_names[last_holiday_ind+1]
  days_to_next_holiday <- abs(dif[last_holiday_ind+1])
  # while days_from_last_holiday/days_to_next_holiday are numeric, they are
  # coerced to characters due to the fact to last_holiday/next_holiday are
  # characters and the use of c() function.
  # Thus, outside of this function, we have to convert them back to numeric
  # if we want to use them in numerical computation.
  return(c(last_holiday, days_from_last_holiday,
           next_holiday, days_to_next_holiday))
}
```

Figure 4.22: Generating feature-set 2

- Ultimately, we end up obtaining many derived results that would count in modelling the GBM for each department such as **train,test sets with new attributes,trainWeights, testID**

- As the dataset consists of 10,000 tuples a sample of the obtained training set is being shown here in fig 4.23:



Day_Index	IsHoliday	Type	Size	Temperature	Fuel_Price	CPI	Unemployment	MarkDown1	MarkDown2	MarkDown3	MarkDown4	MarkDown5	Weekly_Sales	Holiday	Last_Holiday	Days_From_Last_Holiday
1	FALSE	A	151315	42.31	2.572	211.0964	8.106	NA	NA	NA	NA	NA	24924.50	No	Christmas	35
2	TRUE	A	151315	38.51	2.548	211.2422	8.106	NA	NA	NA	NA	NA	46039.49	Super_Bowl	Christmas	42
3	FALSE	A	151315	39.93	2.514	211.2891	8.106	NA	NA	NA	NA	NA	41595.55	No	Super_Bowl	7
4	FALSE	A	151315	46.63	2.561	211.3196	8.106	NA	NA	NA	NA	NA	19403.54	No	Super_Bowl	14
5	FALSE	A	151315	46.50	2.625	211.3501	8.106	NA	NA	NA	NA	NA	21827.90	No	Super_Bowl	21
6	FALSE	A	151315	57.79	2.667	211.3806	8.106	NA	NA	NA	NA	NA	21043.39	No	Super_Bowl	28
7	FALSE	A	151315	54.58	2.720	211.2156	8.106	NA	NA	NA	NA	NA	22136.64	No	Super_Bowl	35
8	FALSE	A	151315	51.45	2.732	211.0180	8.106	NA	NA	NA	NA	NA	26229.21	No	Super_Bowl	42
9	FALSE	A	151315	62.27	2.719	210.8204	7.808	NA	NA	NA	NA	NA	57258.43	No	Super_Bowl	49
10	FALSE	A	151315	65.86	2.770	210.6229	7.808	NA	NA	NA	NA	NA	42960.91	No	Super_Bowl	56
11	FALSE	A	151315	66.32	2.808	210.4887	7.808	NA	NA	NA	NA	NA	17596.96	No	Super_Bowl	63
12	FALSE	A	151315	64.84	2.795	210.4391	7.808	NA	NA	NA	NA	NA	16145.35	No	Super_Bowl	70
13	FALSE	A	151315	67.41	2.780	210.3895	7.808	NA	NA	NA	NA	NA	16555.11	No	Super_Bowl	77
14	FALSE	A	151315	72.55	2.835	210.3400	7.808	NA	NA	NA	NA	NA	17413.94	No	Super_Bowl	84
15	FALSE	A	151315	74.78	2.854	210.3374	7.808	NA	NA	NA	NA	NA	18926.74	No	Super_Bowl	91
16	FALSE	A	151315	76.44	2.826	210.6171	7.808	NA	NA	NA	NA	NA	14773.04	No	Super_Bowl	98

Figure 4.23: New train.csv

- This workspace itself is being saved as a whole rather than each of the intermediate outputs and being directly used in the following codes for efficient memory storage.

4.4.2 SALES PATTERN OF DEPARTMENTS

We can draw an observation that for the same department, the weekly sales (or the sales pattern) are very similar despite of different magnitudes across all the stores. So, it might seem that the same department sells the same kind of products.

As there are 99 departments in total, for sample 2 departments are being shown in fig 4.26:

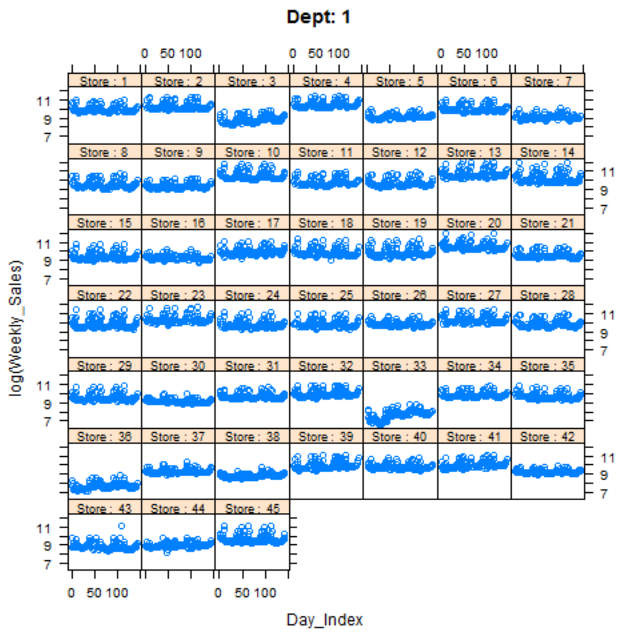


Figure 4.24: Dept-1 across all stores

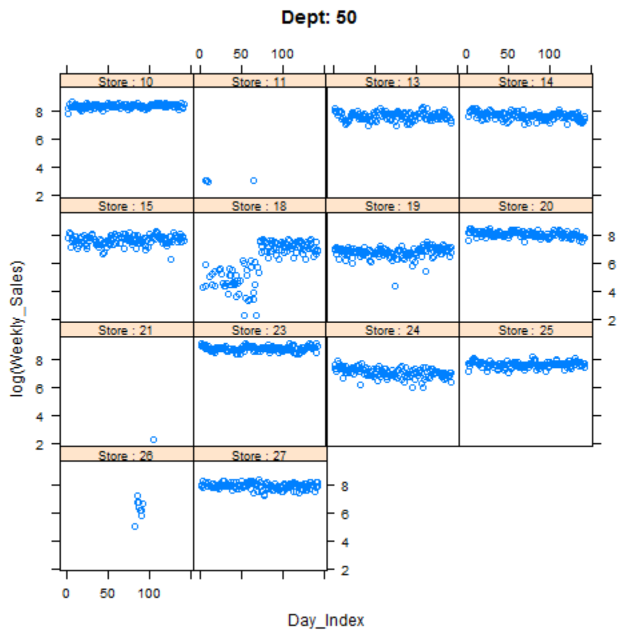


Figure 4.25: Dept-50 across all stores

Figure 4.26:

4.4.3 GENERALIZED BOOSTED REGRESSION MODELS

- As mentioned in significance, digging a level deeper into the abstraction proved by tree model, we are here with a GBM [14] for each department across all stores.
- For simplicity, we may assume that the GBM follows a Gaussian / Laplace distribution standard deviation as 7 (1 week) by tuning parameters of gbm package [15].
- This code calls the previously saved workspace of feature extraction to operate upon derived datasets.
- The previously derived intuition of sales patterns of departments is being used in predTest() function of this code when we come across **a situation where we have to predict weekly sales across a (store + dept) combination that isn't available in train sets.**
- As model should learn from historical data, in such situations using the above observation, we can consider combinations of required department with all the stores available in train set, predict weekly-sales and finally the desired solution is obtained from the mean/median of the above combinations.

4.4.3.1 CONCEPTS INVOLVED

BOX-COX TRANSFORM: [16]

- It is a procedure to identify an appropriate exponent (lambda) to transform the data to improve its normality.The Lambda value indicates the power to which all data should be raised.
- There are 2 methods for calculating lambda, the *guerrero* and *loglik method* (reference is the BoxCox.lambda function in the forecast package).
- In Guerrero method, lambda minimizes the coefficient of variation for subseries of data. In the loglik method, the value of lambda is chosen to maximize the profile log likelihood of a linear model fitted to data.

- Generally, guerreo method gives good values of lambda as compared to the loglik method, which help in better forecasting results.
- For instance, considering a simple linear regression:

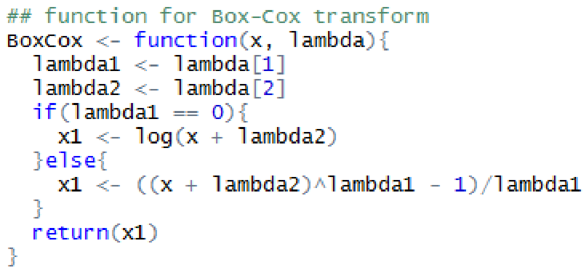
$$Y_i = \beta_0 + \beta_1 X_i + \varepsilon_i$$

Let us introduce as suggested in Box-Cox the following family of (power) transformations on the variable of interest.

$$Y_i^{(\lambda)} = \begin{cases} \dfrac{Y_i^\lambda - 1}{\lambda} & (\lambda \neq 0) \\ \log(Y_i) & (\lambda = 0) \end{cases}$$

Then assume that

$$Y_i^{(\lambda)} = \beta_0 + \beta_1 X_i + \varepsilon_i$$



```
## function for Box-Cox transform
BoxCox <- function(x, lambda){
  lambda1 <- lambda[1]
  lambda2 <- lambda[2]
  if(lambda1 == 0){
    x1 <- log(x + lambda2)
  }else{
    x1 <- ((x + lambda2)^lambda1 - 1)/lambda1
  }
  return(x1)
}
```

Figure 4.27: Corresponding usage in R

INVERSE TRANSFORM:

Considering a cumulative distribution function (F) for any distribution, the inverse transform is given as:

$$F_X^{-1}(u) = x$$

The idea is simple: it is easy to sample values from U(0,1). So, if we want to sample values from some F, just consider values u from U(0,1) and pass u through inverse of F to obtain x as above.But this is not always possible as every function doesnot have an inverse but here the distribution being Gaussian this works.

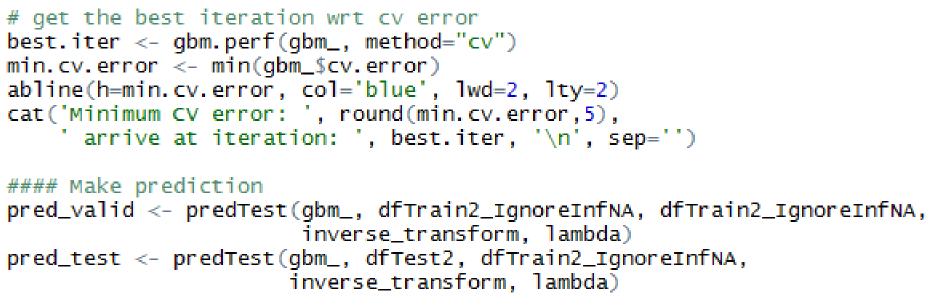
k-FOLD CROSS VALIDATION: [17]

- **Statement:** Whenever we run K-fold cross-validation, we use K subsets of the training data, and end up with K different models.
- **Significance:**
  - So, when we do K-fold cross validation, we are testing how well our model is able to get trained by some data and then predict data it hasn't seen. We use cross validation for this because training using all the data we have, we are left with no data for testing. We could do this once, say by using 80 percent of the data to train and 20 percent to test, but what if the 20 percent we happened to pick to test contains a bunch of points that are particularly easy (or particularly hard) to predict? We will not have come up with the best estimate possible of the models ability to learn and predict.
  - We want to use all of the data. So to continue the above example of an 80/20 split, we would do 2-fold cross validation by training the model 2 times on 80 percent of the data and testing on 20 percent. We ensure that each data point ends up in the 20 percent test set exactly once. We have therefore used every data point we

have, to contribute to an understanding of how well our model performs the task of learning from some data and predicting some new data.

- **Purpose:**
  - For instance, say we have two models, a linear regression model and a neural network. How can we say which model is better? We can do K-fold cross-validation and see which one proves better at predicting the test set points. But once we have used cross-validation to select the better performing model, we train that model (whether it be the linear regression or the neural network) on all the data. We don't use the actual model instances we trained during cross-validation for our final predictive model. **The purpose of cross-validation is model checking, not model building.**

With this method, we find the best iteration number corresponding to Min CV error and move forward with the corresponding tree for predicting weekly sales.



```
# get the best iteration wrt cv error
best.iter <- gbm.perf(gbm_, method="cv")
min.cv.error <- min(gbm_$cv.error)
abline(h=min.cv.error, col='blue', lwd=2, lty=2)
cat('Minimum CV error: ', round(min.cv.error,5),
    ' arrive at iteration: ', best.iter, '\n', sep='')

#### Make prediction
pred_valid <- predTest(gbm_, dfTrain2_IgnoreInfNA, dfTrain2_IgnoreInfNA,
                       inverse_transform, lambda)
pred_test <- predTest(gbm_, dfTest2, dfTrain2_IgnoreInfNA,
                      inverse_transform, lambda)
```

Figure 4.28: Usage in R

4.5 OUTPUTS

- Using lambda (guerrero method) few records (out of 10000 records) were obtained as normal data which were not initially normal. For example the plot, for sample, shows one of the time series data before and after the transformation. In this way, we obtain histograms for every department

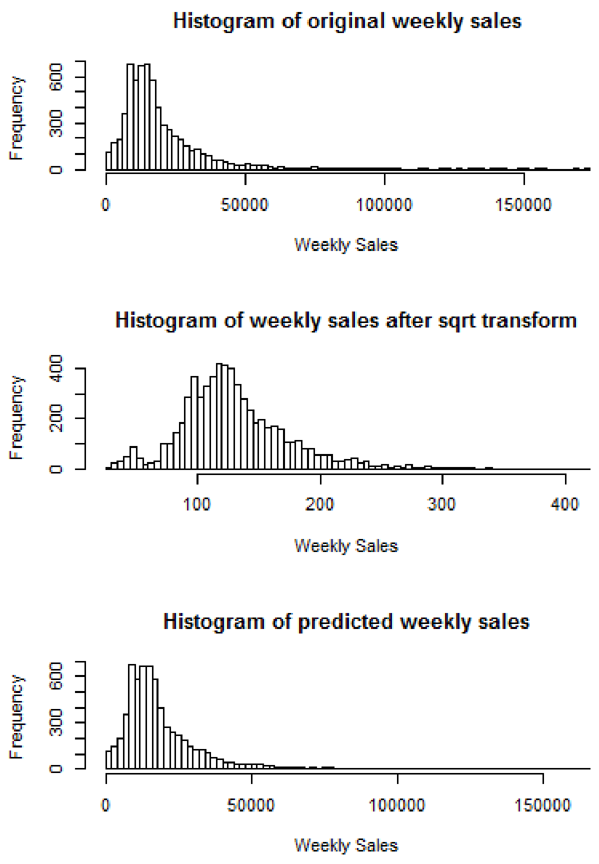


Figure 4.29: Histogram before and after Box-Cox transform

- As mentioned earlier, to obtain the best iteration number with respect to CV error,a graph is plotted as follows:

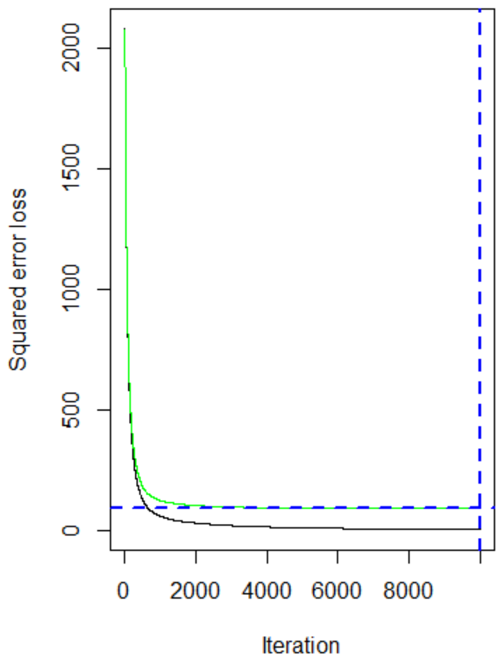


Figure 4.30: Choice of iterations

- **Time Series Analysis:** The desired output of weekly-sales (in terms of revenue) vs date-index (a derived attribute) for each combination of department and store are obtained as follows. As there are 99 departments across each of the 45 stores, a GBM following Gaussian distribution is given for each of them.
  - —>If the test record to be predicted is available in the training data, then the actual value was represented by black colour and the predicted value by red.
  - —>This facilitates comparison of predicted, actual values and accuracy of prediction too.
  - —>If not, directly the predicted value of weekly-sales is represented by red alone.

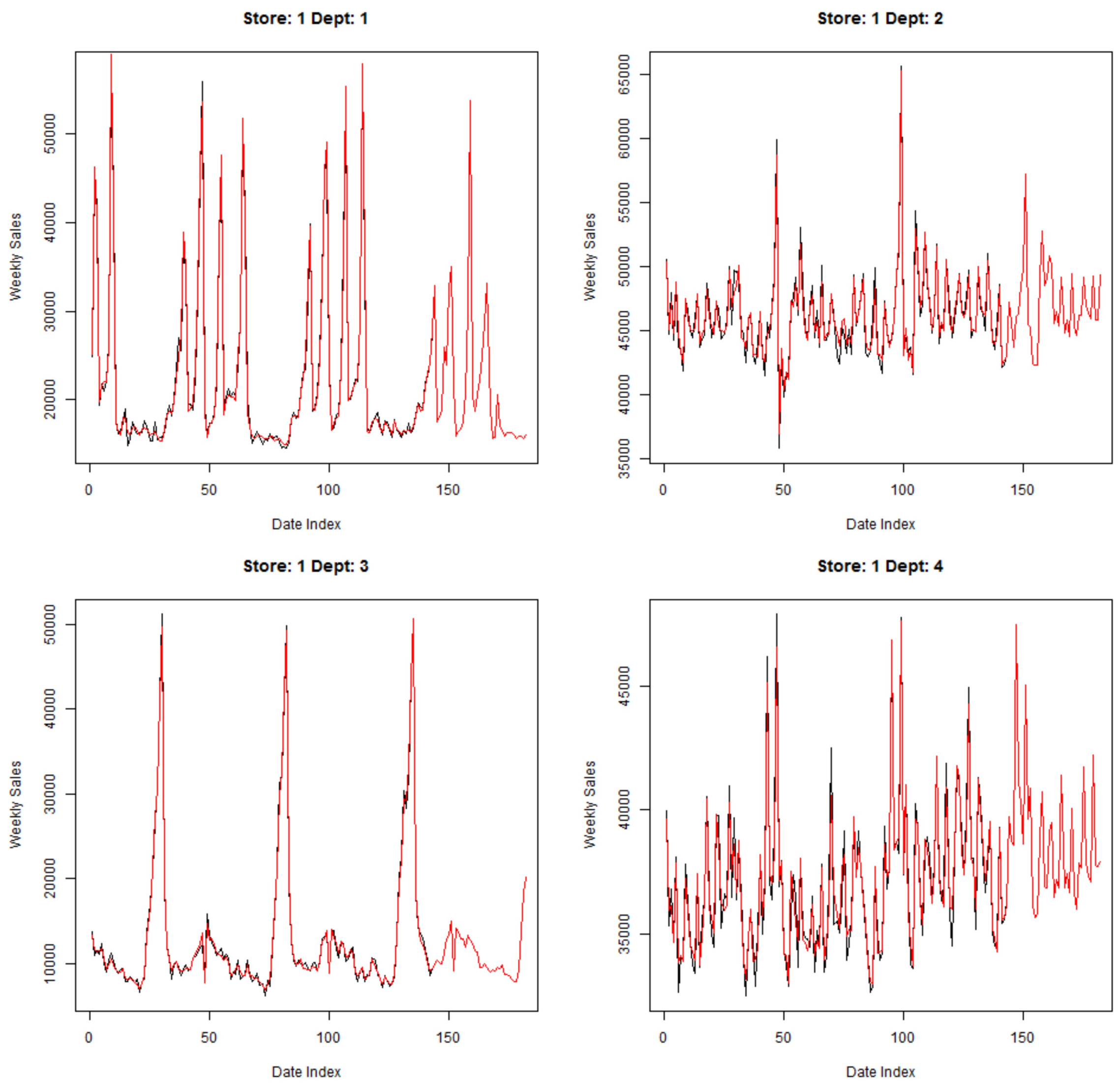
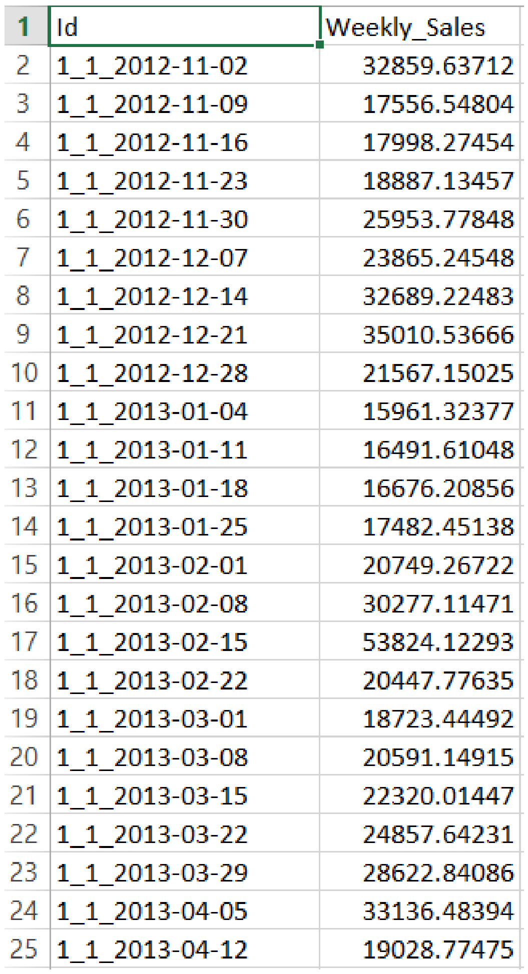


Figure 4.31: Time Series Analysis

- For convenience, A dataset has been generated predicting the weekly sales (in terms of revenue) for each id (id = store+dept+date).The weekly-sales represented by the mean of their corresponding graph distributions. A sample of the dataset can be given as in fig 4.32:



1	Id	Weekly_Sales
2	1_1_2012-11-02	32859.63712
3	1_1_2012-11-09	17556.54804
4	1_1_2012-11-16	17998.27454
5	1_1_2012-11-23	18887.13457
6	1_1_2012-11-30	25953.77848
7	1_1_2012-12-07	23865.24548
8	1_1_2012-12-14	32689.22483
9	1_1_2012-12-21	35010.53666
10	1_1_2012-12-28	21567.15025
11	1_1_2013-01-04	15961.32377
12	1_1_2013-01-11	16491.61048
13	1_1_2013-01-18	16676.20856
14	1_1_2013-01-25	17482.45138
15	1_1_2013-02-01	20749.26722
16	1_1_2013-02-08	30277.11471
17	1_1_2013-02-15	53824.12293
18	1_1_2013-02-22	20447.77635
19	1_1_2013-03-01	18723.44492
20	1_1_2013-03-08	20591.14915
21	1_1_2013-03-15	22320.01447
22	1_1_2013-03-22	24857.64231
23	1_1_2013-03-29	28622.84086
24	1_1_2013-04-05	33136.48394
25	1_1_2013-04-12	19028.77475

Figure 4.32: Sample of dataset generated

- **Exploratory Analysis:**
  - The pie-chart shows the top 10 stores in terms of sales revenue and their percentage contribution to the total sales generated between them. Store 20 was the highest contributor with a total of 301 Million.Together, these 10 stores accounted for 39 percent of the revenue generated by the given 45 stores.

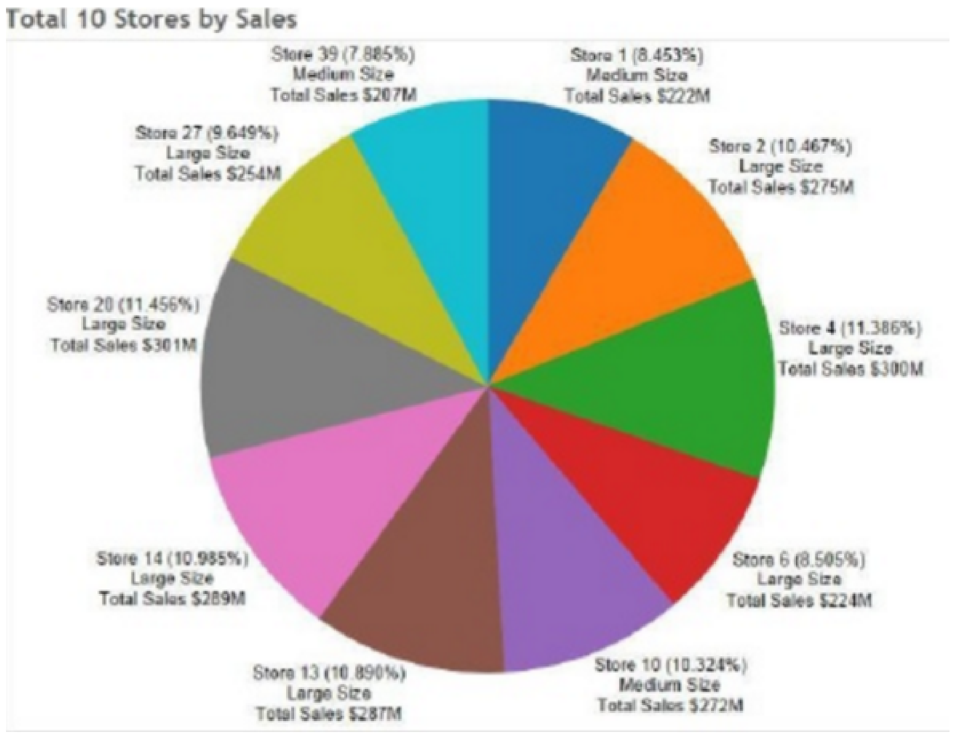


Figure 4.33: Pie chart analysis

– We can extend the analysis further to show the top 3 departments across the 3 store types namely A,B C. Interestingly,the Department number showing a significant hike in sales across stores can be recruited with more man-power. Similarly, incentives can be provided to stores fetching the least sales.

Chapter 5

CONCLUSION

- In conclusion, predictive models are found to be the assets of a retail market management which helps the retailers to graph and analyze the pulse of the business and react quickly to the constantly shifting market space.

- In this project, we have tried to unfold the essence of predictive modelling by developing a tree based model to determine the choice of price, quantity prediction for every product. If the attribute being predicted is a categorical data, all of the dataforms are finally being transformed from 'categorical' to 'Factor form' for implementation of random forests. The numeric data are being discretized by binning to convert them into categorical data and this inturn is converted to factor data involving a 2-step transformtion.

- The process of modelling has been further enhanced by trying to obtain time series analytic graphs to handle numeric data and contribute to an overall sales forecasting of the firm.

5.1 BUSINESS IMPLICATIONS

- Based on the analysis made, we can interpret as: that combination of (store + dept) with higher weekly-sales can recruit more man-power for managing sales or keep higher stocks of quantities in accordance with demand. This allows them to perform better when the sales go up gradually or when the holidays get closer.

- Retail stores should keep a close eye on the stores which are running out of business implied by lowest sales and provide incentives to them to improve their sales, and hire the right sales representatives.

5.2 SOURCE CODE LINK

The implemented code and the corresponding outputs whose sample is shown over here are uploaded in the github for ready reference.

`https://github.com/manpreethsai/Major-Project`

Bibliography

[1] Link for source code and Datasets (https://github.com/manpreethsai/Major-Project)

[2] Vin Vashishta, *(Predictive Analytics For Pricing Strategy: Why and How For Non-Data Scientists)*, "https://www.linkedin.com/pulse/predictive-analytics-pricing-strategy-why-how-vin-vashishta".

[3] Tavish Srivasthava, *(Process of predictive modelling)*, "https://www.analyticsvidhya.com/blog/2015/09/perfect-build-predictive-model".

[4] *[kris Analytics for an Online Retailer price demand prediction]* http://www.hbs.edu/faculty/Publication/Files/kris/Analytics.pdf

[5] Gold price predictor (http://waset.org/publications/467/using-data-mining-methodology-to-build-the-predictive-model-of-gold-passbook-price)

[6] http://www.analyticsvidhya.com/blog/2016/04/complete-tutorial-tree-based-modeling-scratch/nine

[7] Random forests - https://www.analyticsvidhya.com/blog/2014/06/introduction-random-forest-simplified/

[8] Gradient Boosting - https://www.analyticsvidhya.com/blog/2015/09/complete-guide-boosting-methods/

[9] Lecture series on Regression analysis (https://www.youtube.com/watch?v=66z_MRwtFJM&list=PLqzoL9-eJTNBJrvFcN-ohc5G13E7Big0e)

[10] M. Gilliland, "Demand Forecasting in Retail," [Online]. (Available: http://www.sas.com/news/feature/retail/aug06forecast.html)

[11] M. K. . R. R. Nitin Patel, "Clustering Models to Improve Forecasts in Retails Merchandising," [Online]. (Available: http://www.cytel.com/Papers/INFORMS _Prac _%2004.pdf)

[12] S. J. Satyajit Dwivedi, "Time-series Data Mining," [Online]. (Available: http://www.iasri.res.in/sscnars/data_mining/10-SAS %20Enterprise%20Miner%207.1%20Time%20Series%20Data%20Mining.pdf)

[13] Tutorial on Data Binning (https://www.youtube.com/watch?v=20K6fGuTBgw)

[14] A lecture on using GBM for classification in R (https://vimeo.com/71992876)

[15] Tuning parametrs in GBM (http://www.listendata.com/2015/07/gbm-boosted-models-tuning-parameters.html)

[16] Box-cox transforms (https://www.r-bloggers.com/on-box-cox-transform-in-regression-models/)

[17] k-fold cross validation (https://www.analyticsvidhya.com/blog/2015/11/improve-model-performance-cross-validation-in-python-r/)