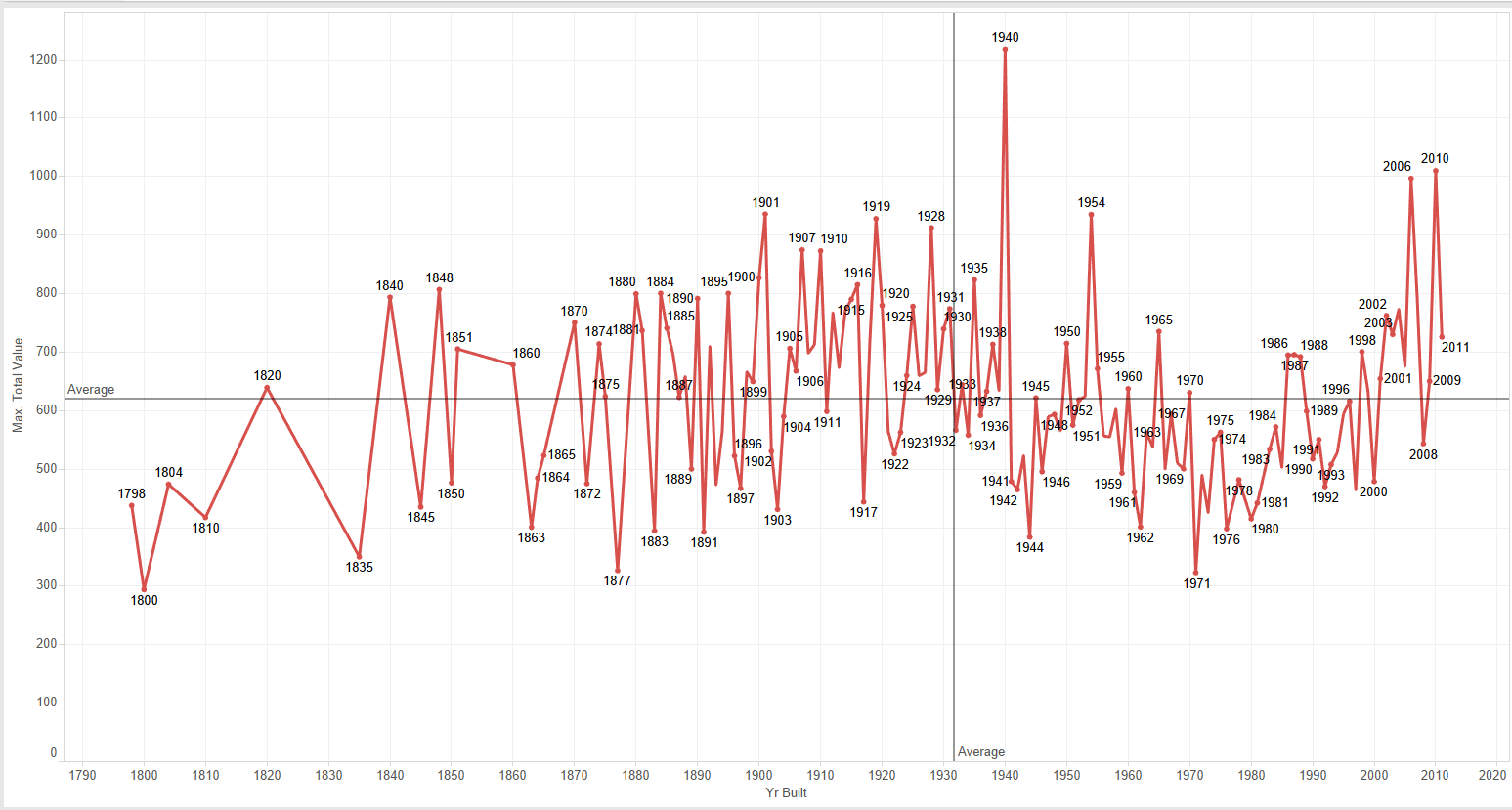
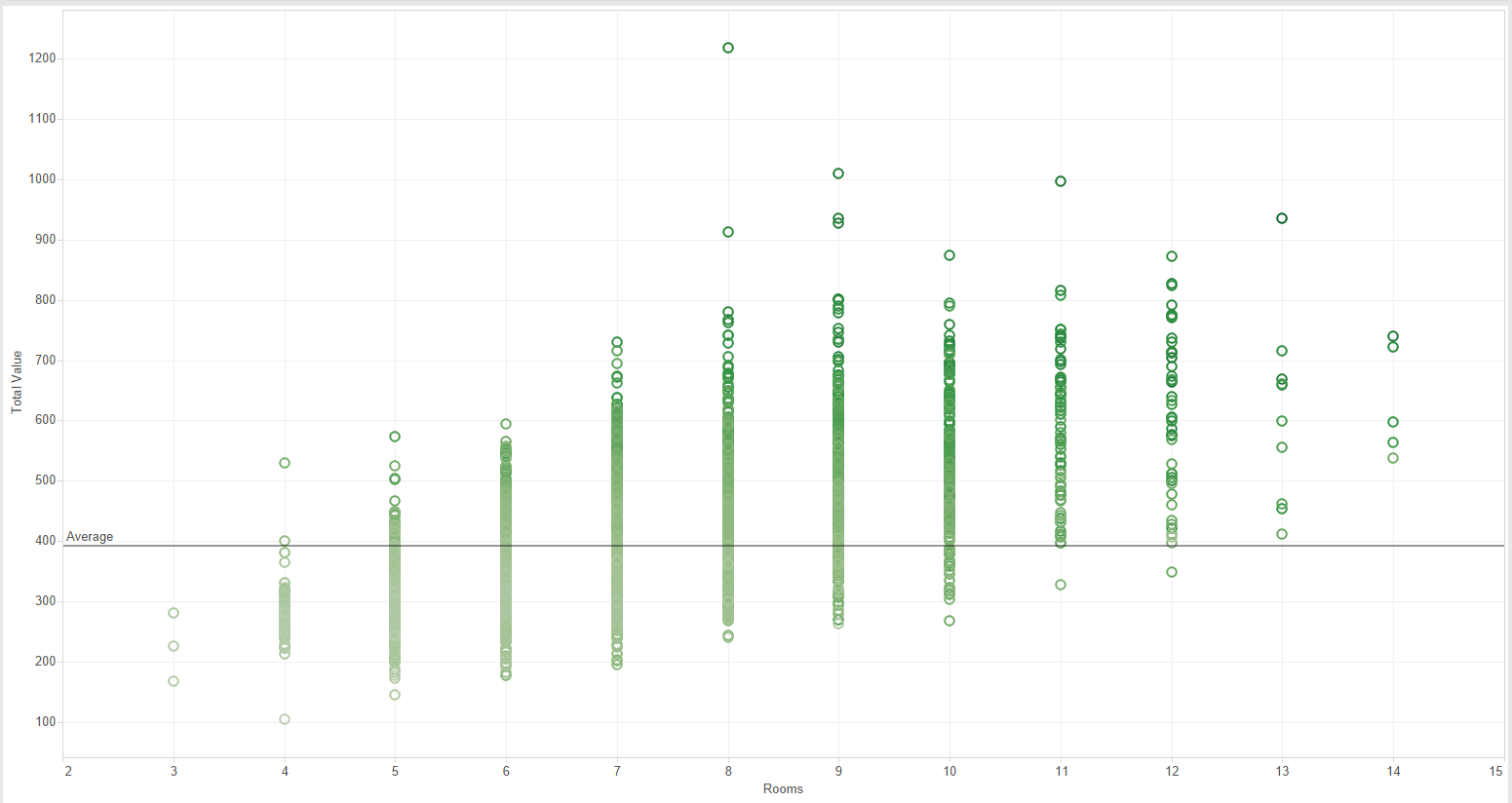
**Question 1:**

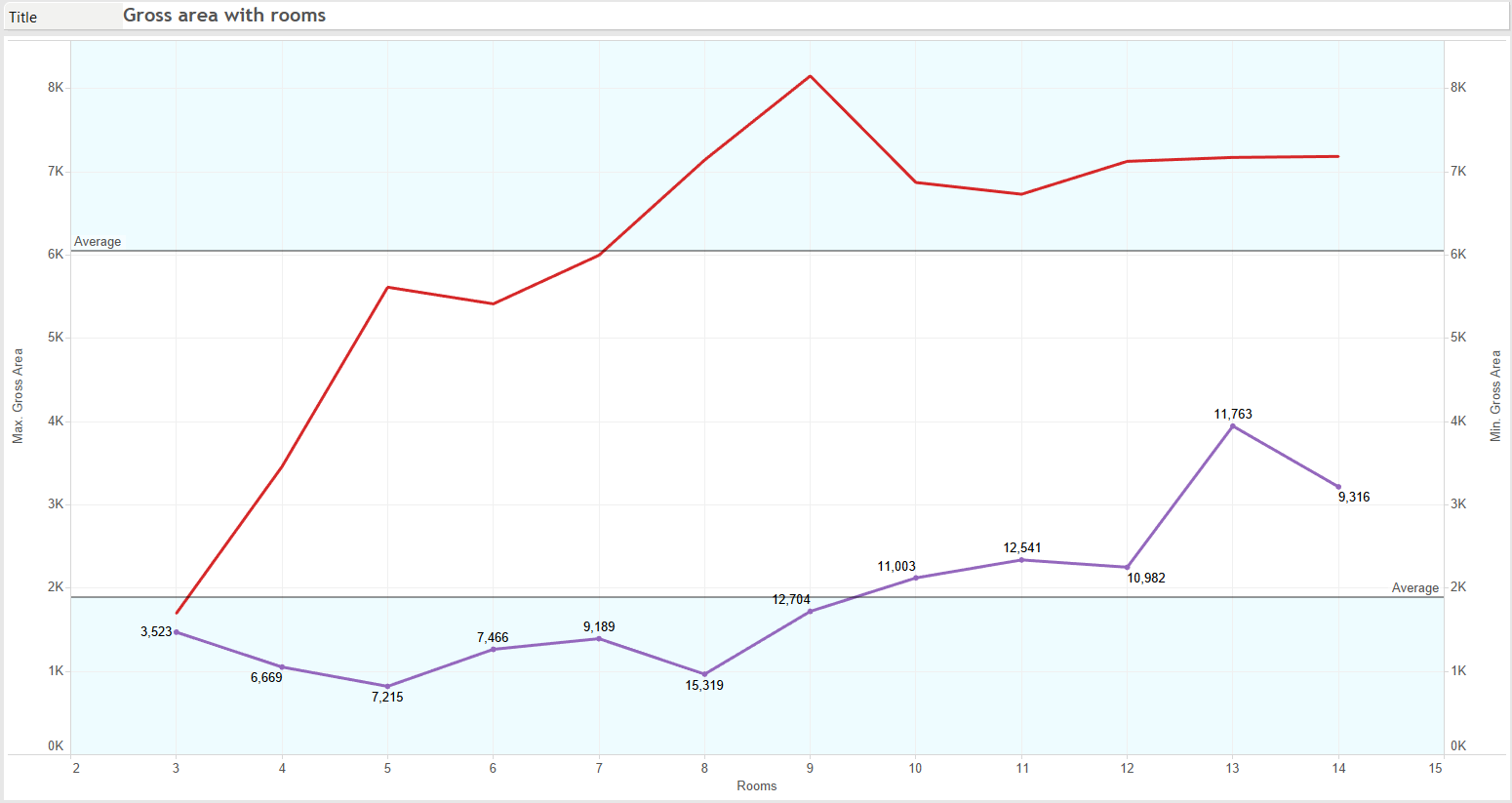
**Cleaning data:**

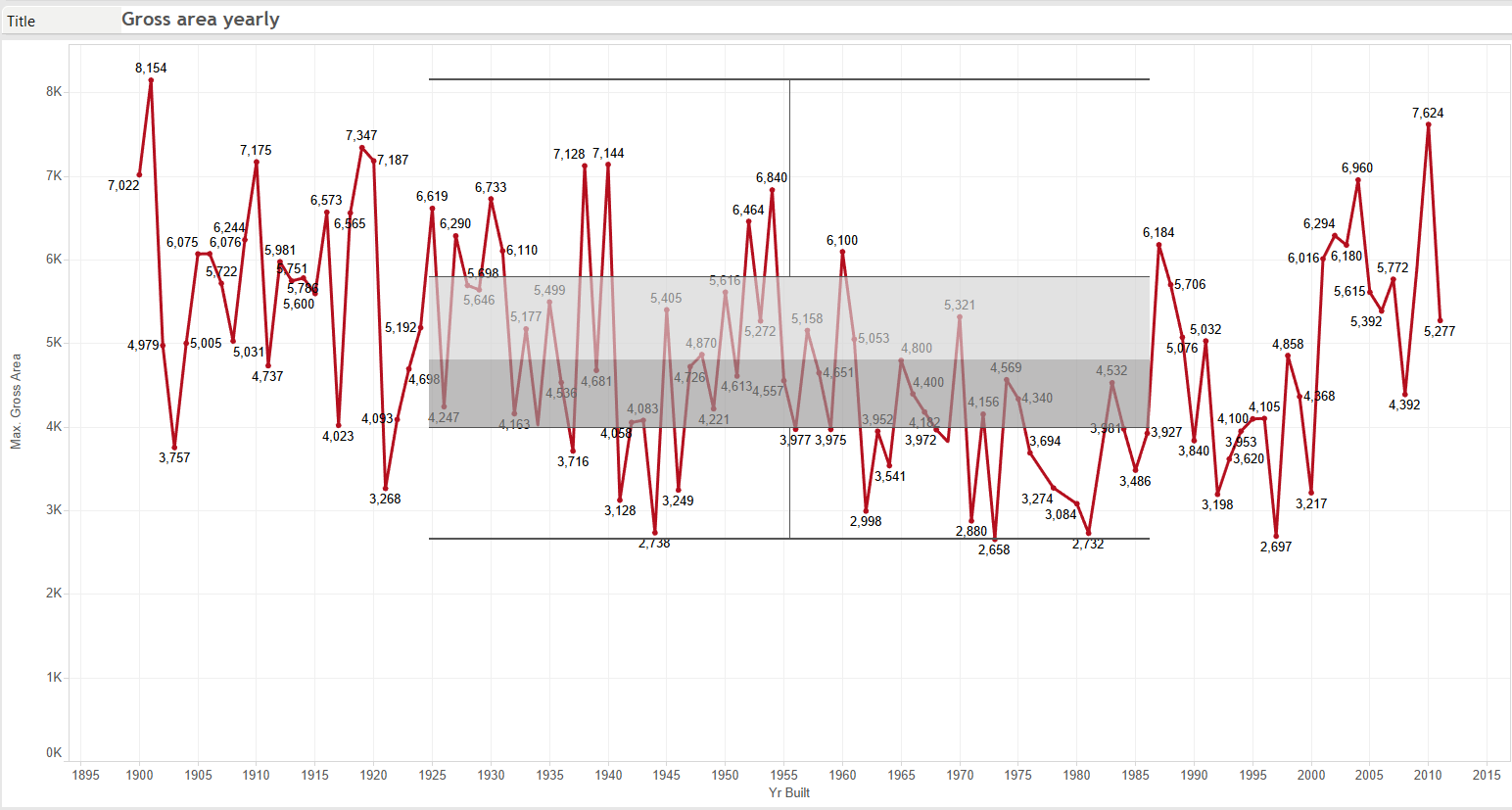
Imported the data set into Wrangler, default datatype for columns was integer. The datatype was changed to decimal to correct the mismatched values in the columns.

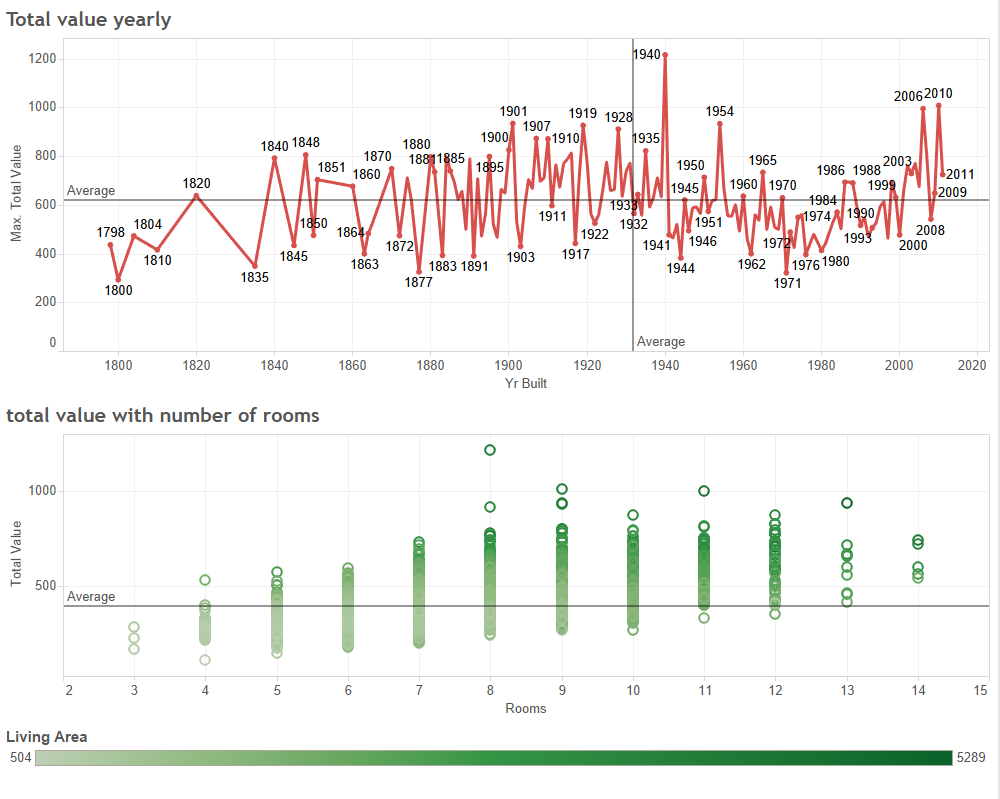
**Exploratory analysis:**













**Creating dummies:**

Variables REMODEL, FLOORS, ROOMS, BEDROOMS,FULL\_BATH, HALF\_BATH, KITCHEN, FIREPLACE had categorical values. Created dummy variables for the mentioned variables. FIREPLACE and FLOORS were ignored as their value did not bring significant change to the result.

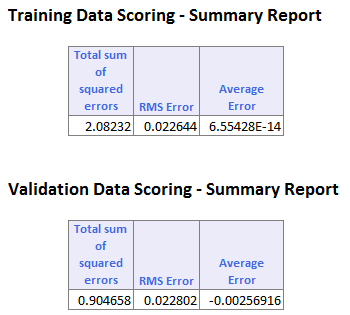
**Partitioning the data:**

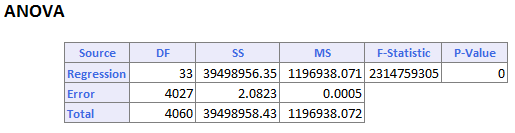
Partitioned data into 60% and 40%.

60% partition is used as training data and 40% partition is used as validation data.

**MULTIPLE LINEAR REGRESSION:**

The cutoff probability value for success was kept at 0.5.

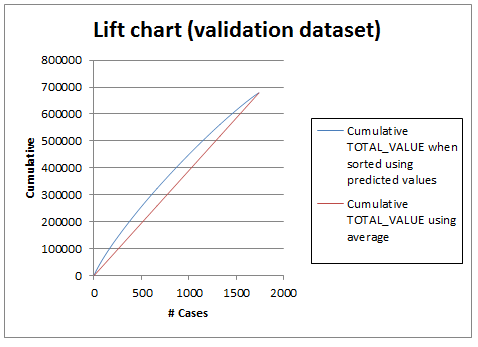
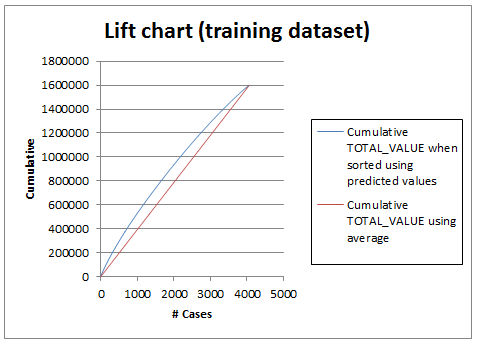




The RMS Error for training data set is 0.0226 whereas for validation data set it is 0.0228.

Used ANOVA to perform regression.

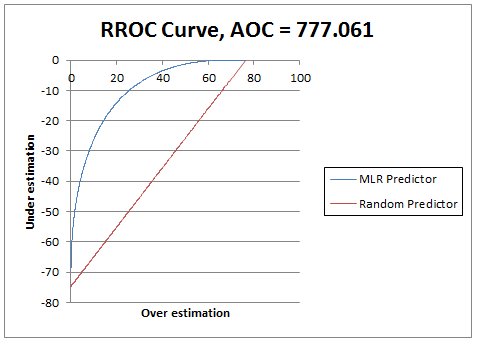
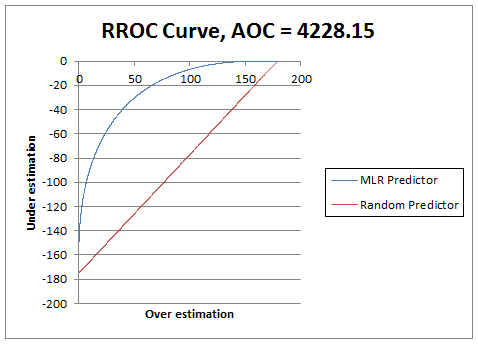
**Lift Chart:**



A lift chart is the graphical way to assess the predictive performance of a model. It compares the model with the baseline model that has no predictors. It can be said that model’s predictive performance is better than baseline model.

**ROC Curve:**

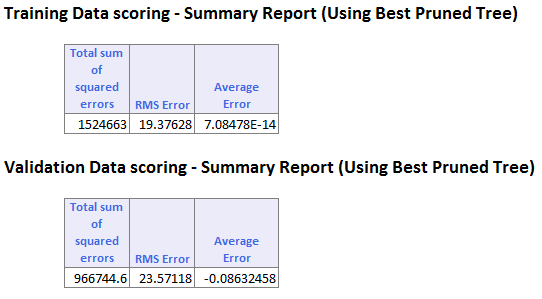
**Training dataset and Validation dataset:**



The ROC (Receiver Operating Characteristic) curve plots the sensitivity and 1-specificity. Better performance are given by the curves that are near the Y axis. Area under the Curve is 777.061 for validation data set as shown in the figure.

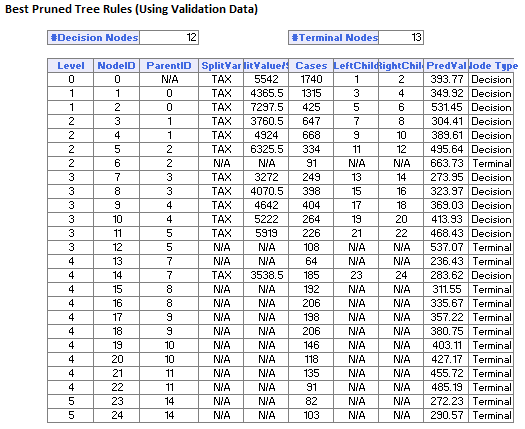
**CART**

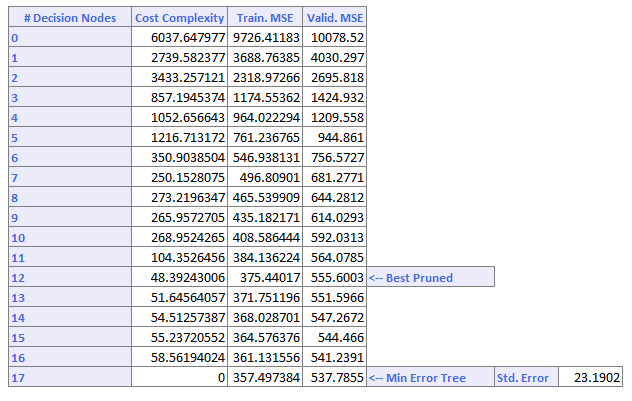
The cutoff probability value for success was kept at 0.5.



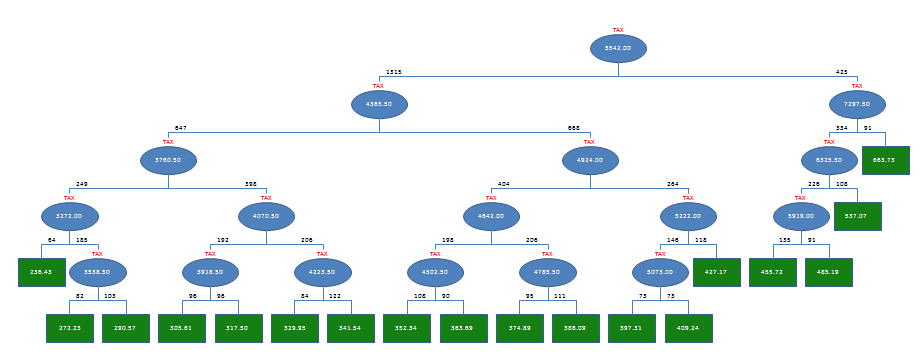
The RMS Error for training data set is 19.37 whereas for validation data set it is 23.57.

**Best Pruned Tree:**

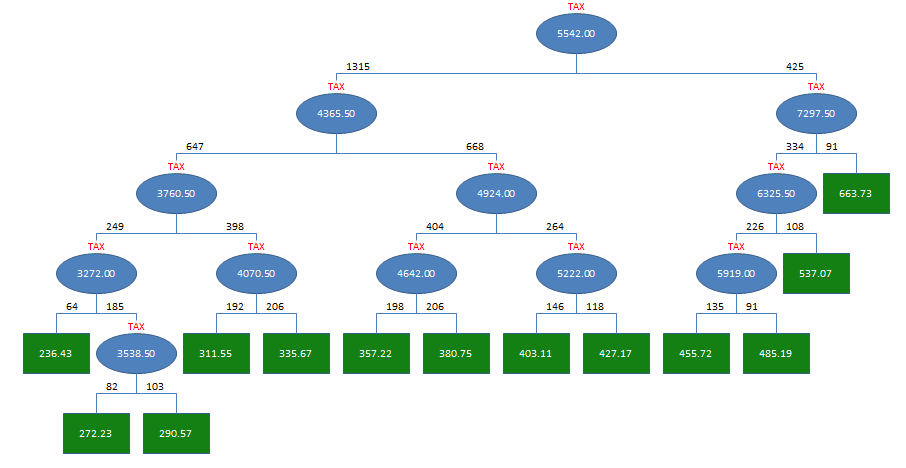




**Minimum Error Tree**

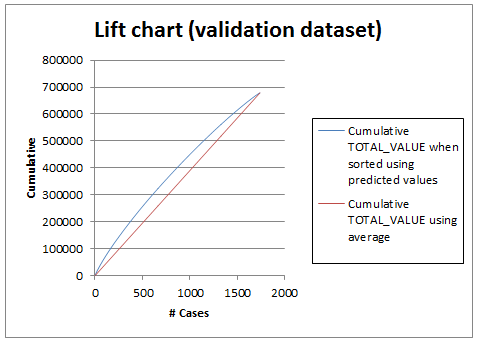
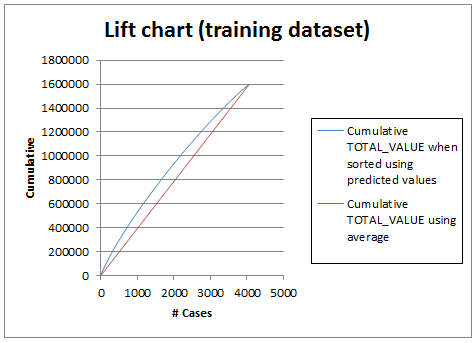


**Best Pruned Tree:**



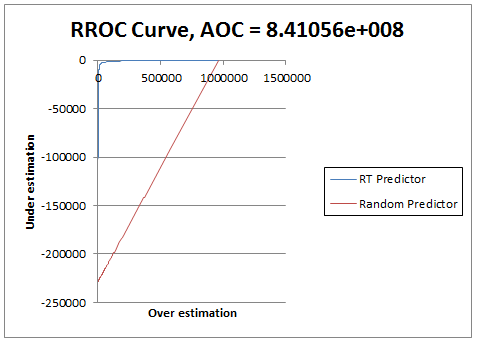
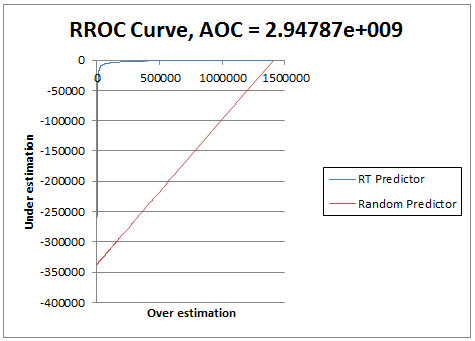
**Lift Chart:**

**Training dataset and Validation dataset:**

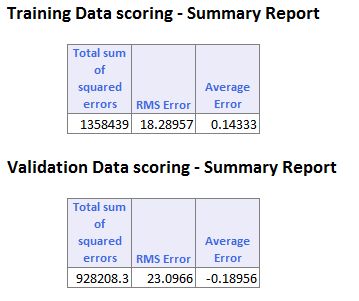


**ROC Curve:**

**Training dataset and Validation dataset:**



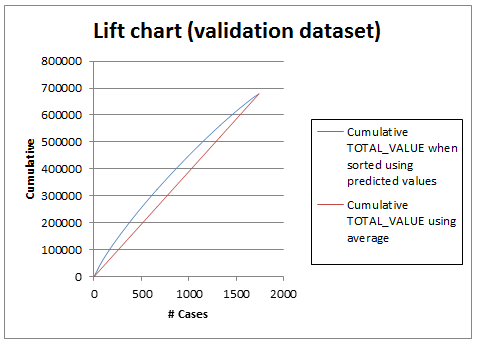
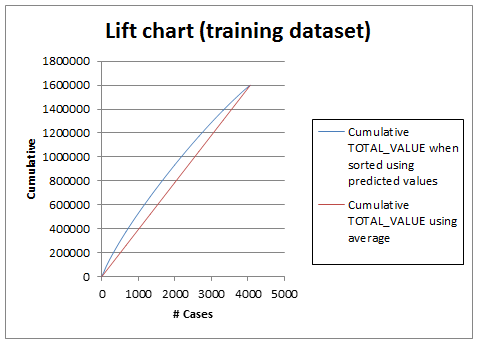
**Random Forest:**



The RMS Error for training data set is 18.28957 whereas for validation data set it is 23.0966.

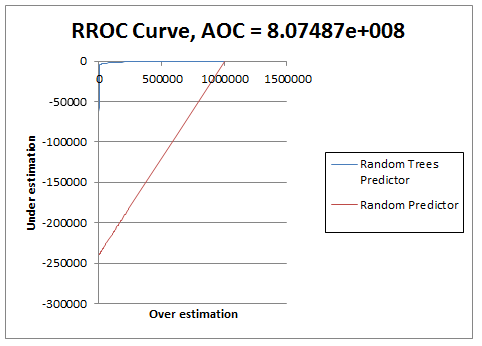
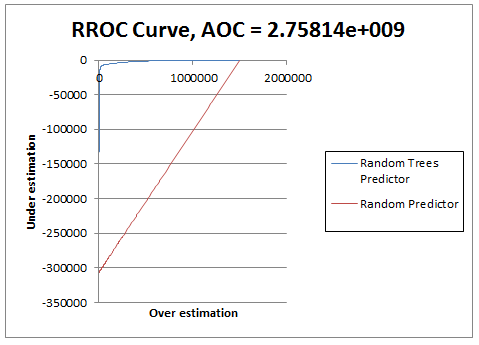
**Lift Chart:**

**Training dataset and Validation dataset:**



**ROC Curve:**

**Training dataset and Validation dataset:**



**Comparison between the models:**

Based on overall percentage error for validation datasets and maximum area under the curve (ROC), Random Forest turns out to be the best model amongst all models.

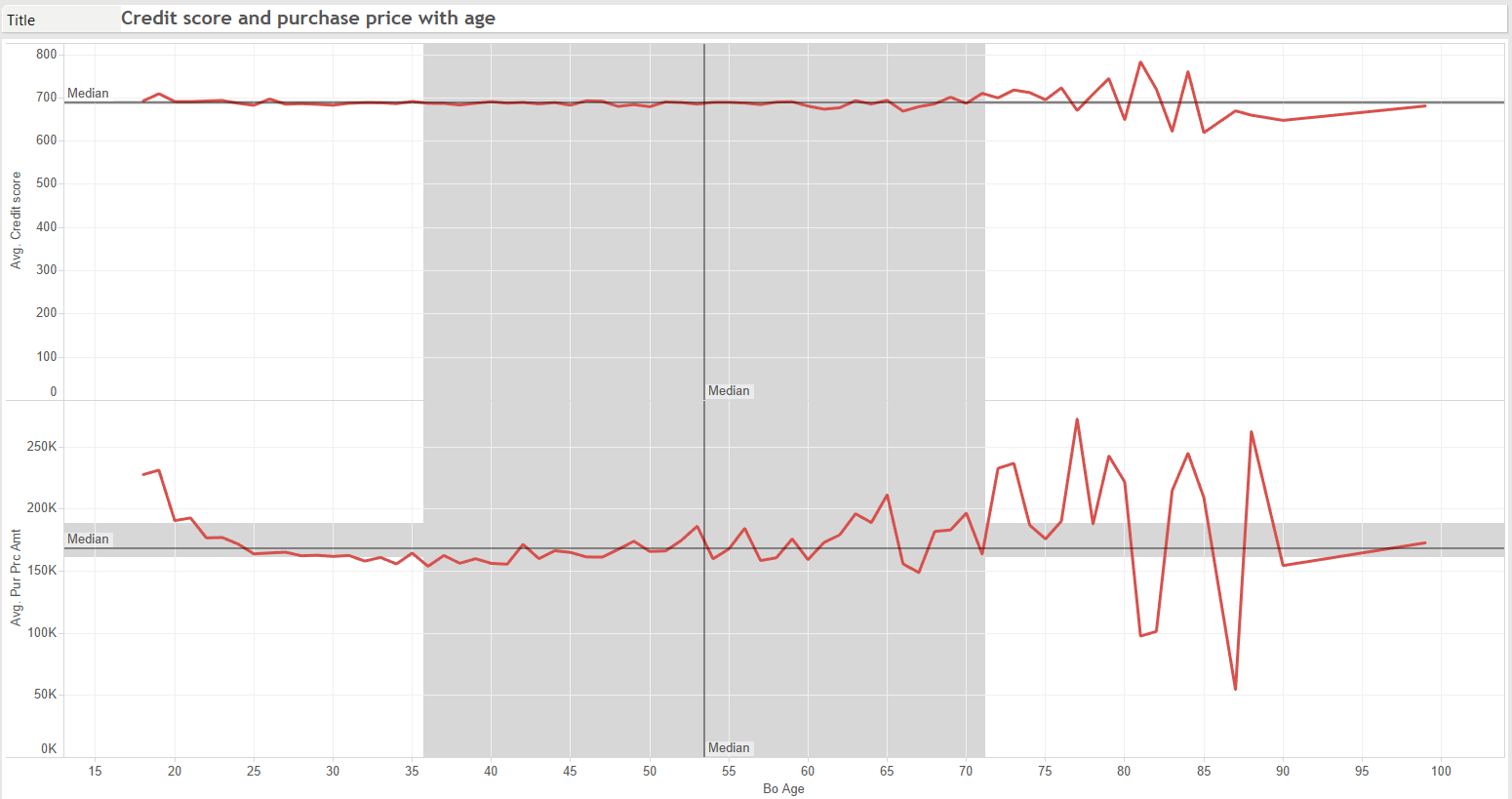
To calculate the Value of a home, random forest model is useful because it gives minimum RMS error and the validation data set has an average error of -0.18.

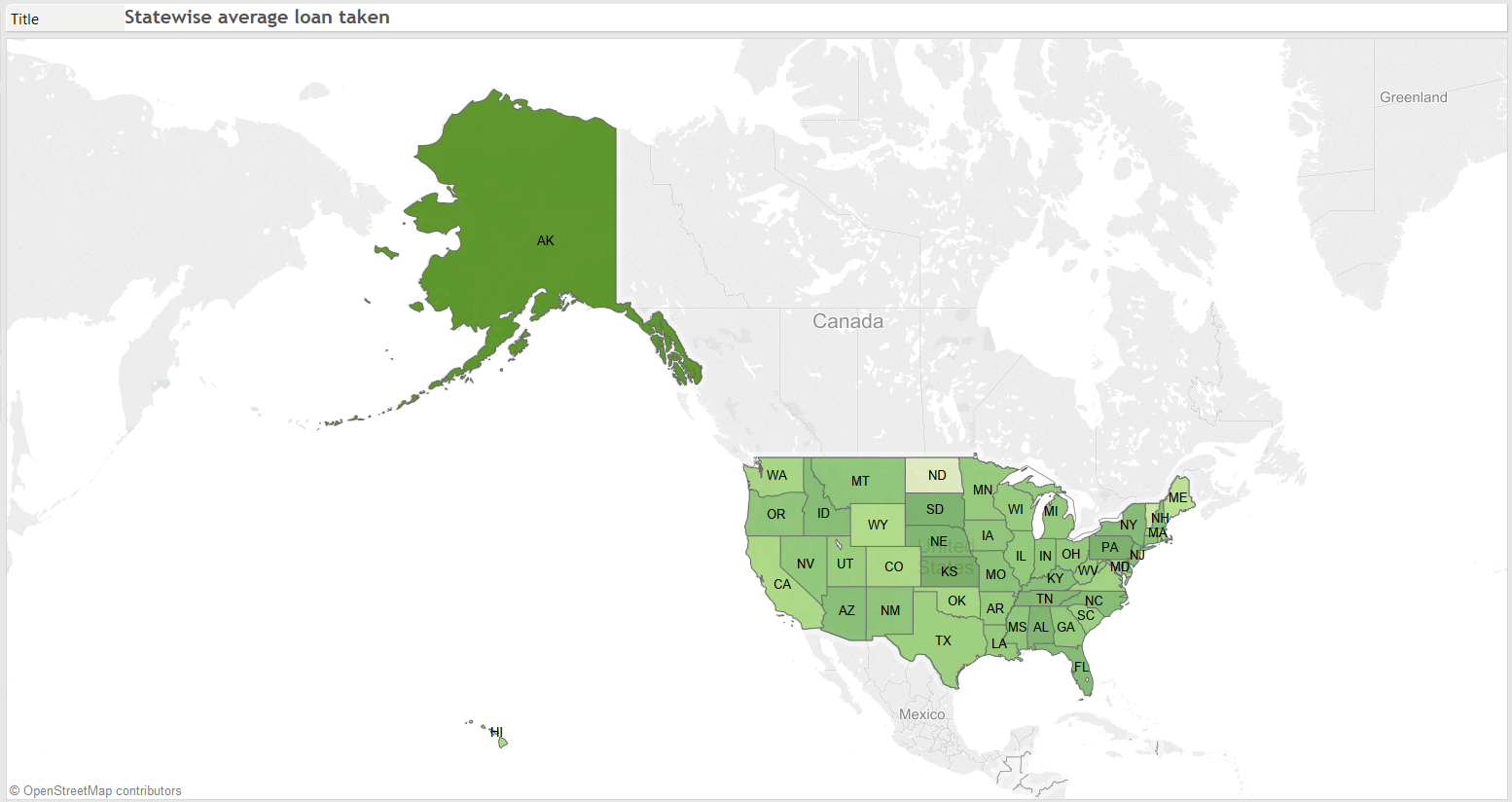
**Question 2:**

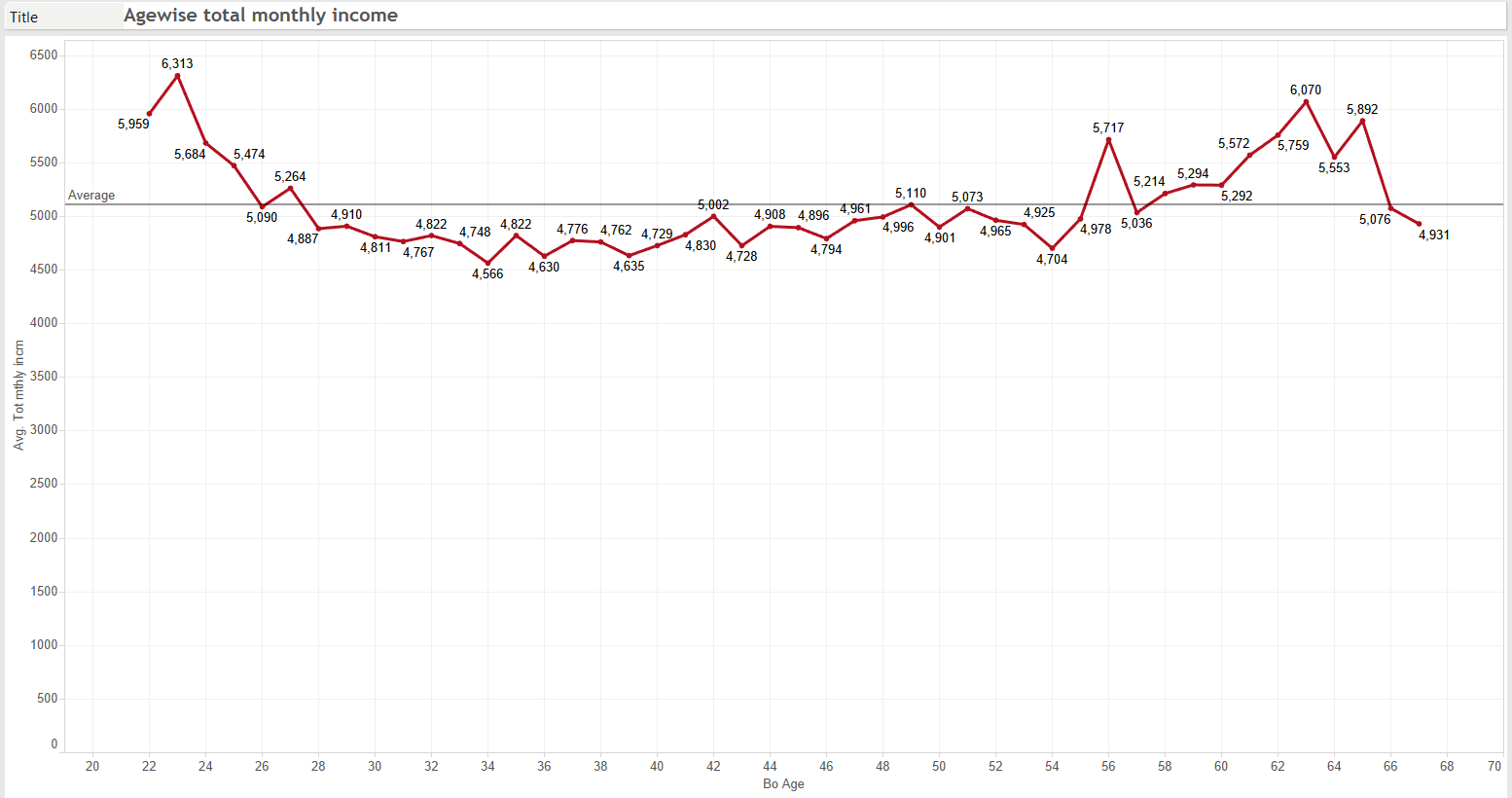
**Cleaning data:**

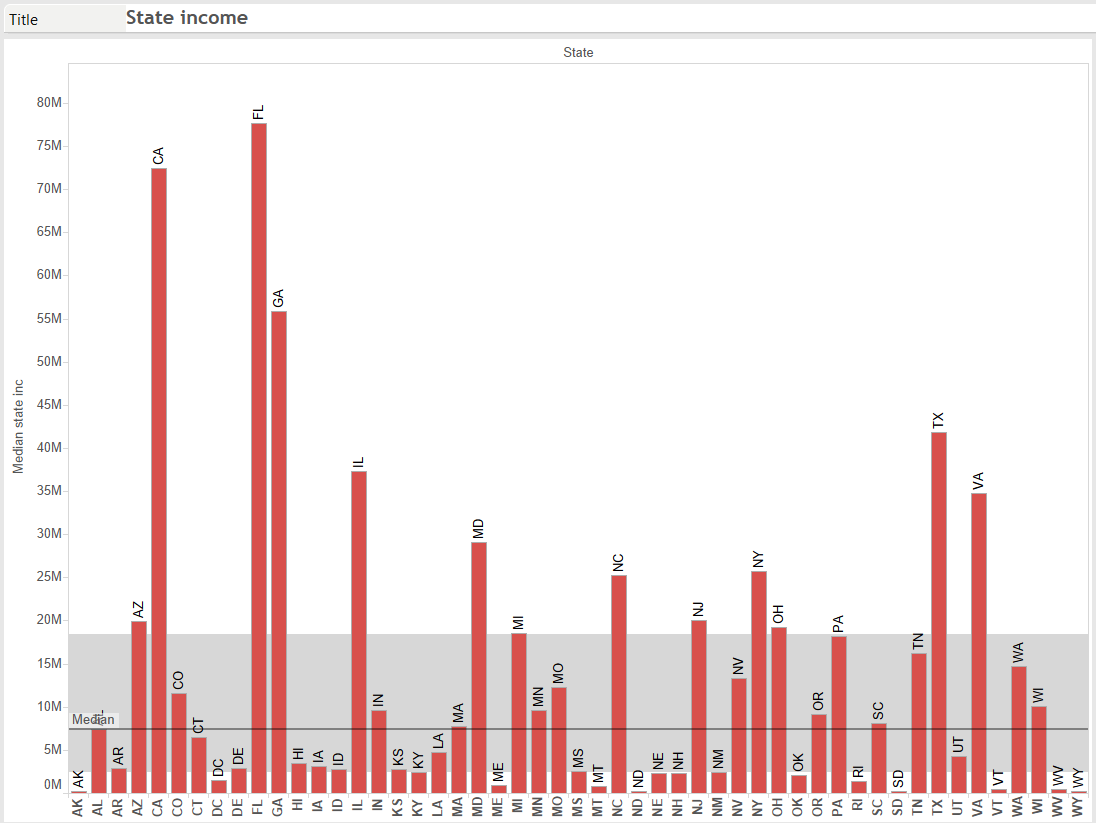
Imported the data set into Wrangler. Some columns had no data and some had missing and mismatched values. Columns without data were dropped missing values were deleted.

**Exploratory analysis:**

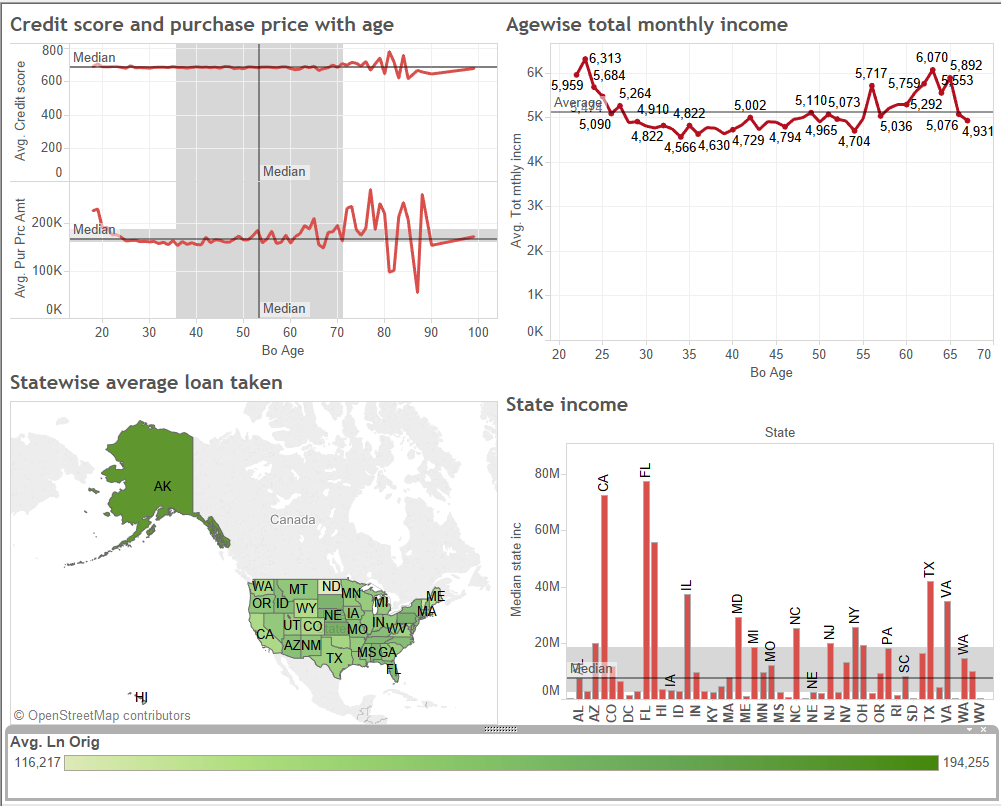








**Dashboard:**



**Creating dummies:**

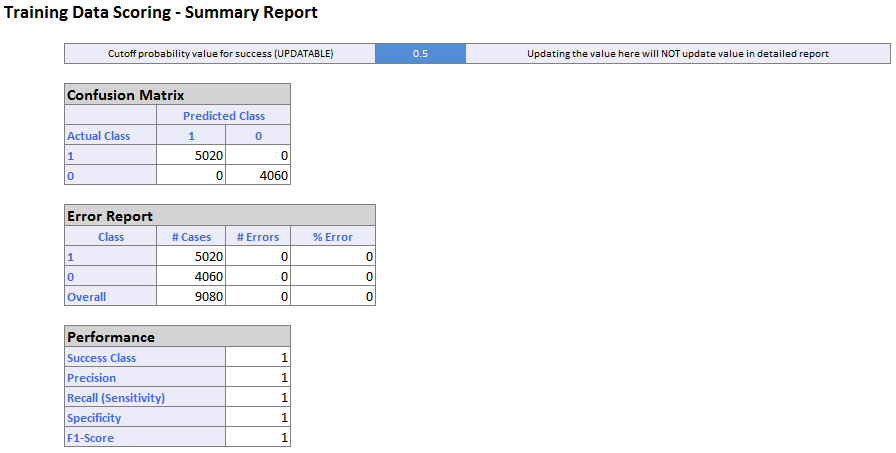
There were several categorical variables like First\_home, Status, OUTCOME, UPB\_Appraisal. Created dummy variables for the mentioned categorical variables.

**Partitioning the data:**

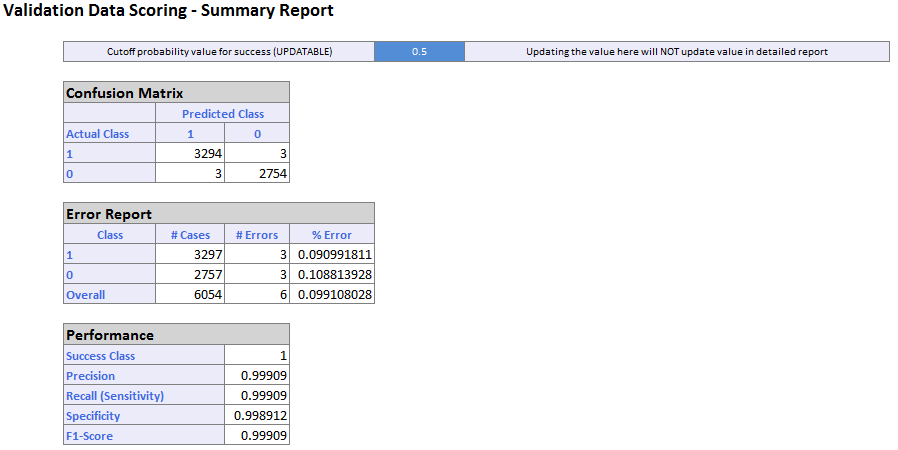
Partitioned data into 60% and 40%. 60% partition is used as training data and 40% partition is used as validation data.

**Logistic Regression:**

The cutoff probability value for success was kept at 0.5.

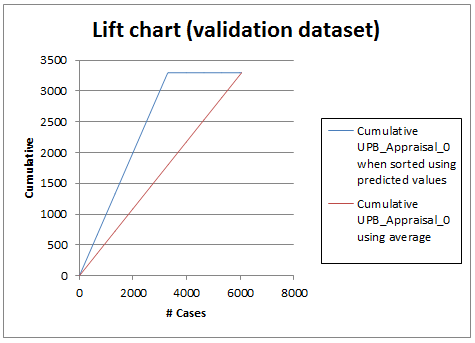
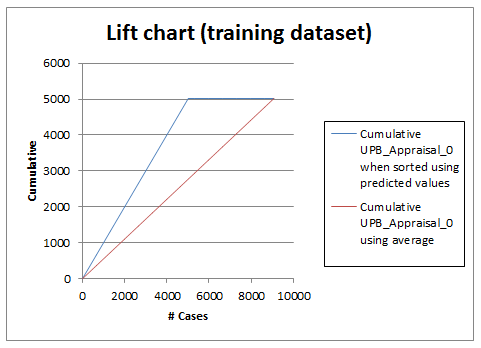


The overall % error rate is 0.00 and model classifies 5020 as Default Outcome and 982 NON Default Outcome correctly.



The overall error rate 0.099% and the model classify 3294 as Default Outcome and 2754 NON Default Outcome correctly.

**Lift Chart:**

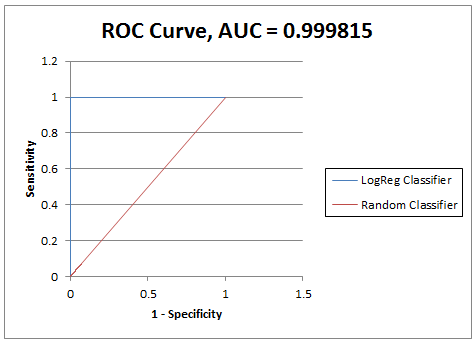
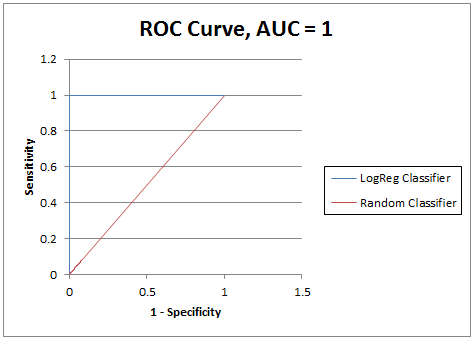


A lift chart is the graphical way to assess the predictive performance of a model. It compares the model with the baseline model that has no predictors.

Lift chart is based on 6000 validation records. It can be said that model’s predictive performance is better than baseline model.

**ROC Curve:**

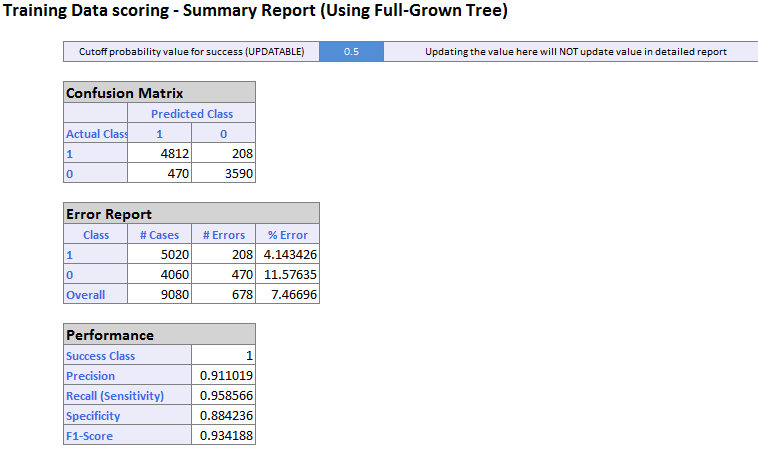
**Training dataset and Validation dataset:**



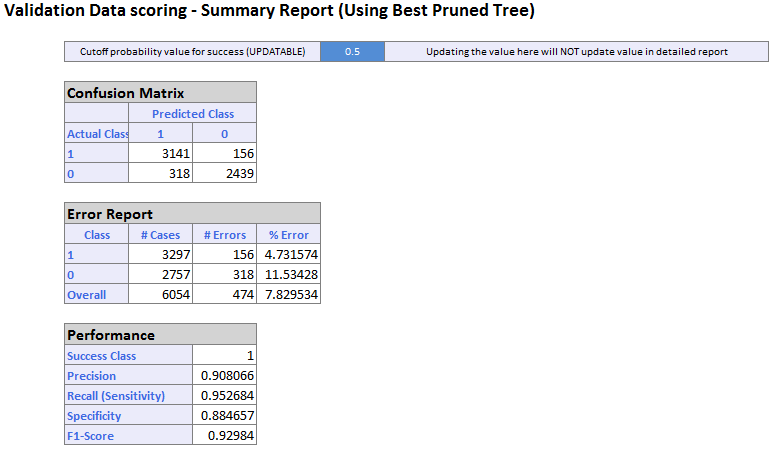
The ROC (Receiver Operating Characteristic) curve plots the sensitivity and 1-specificity. Better performance are given by the curves that are near the Y axis. Area under the Curve for validation dataset is 0.999 as shown in the figure.

**CART**

The cutoff probability value for success was kept at 0.5.

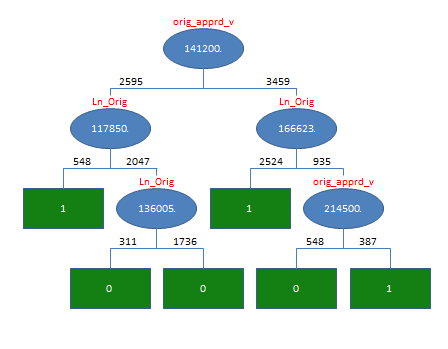


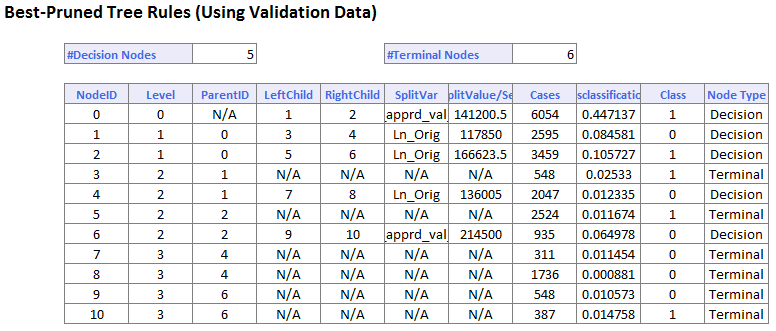
The overall % error rate 7.46 and model classifies 4812 as Default Outcome and 3590 as NON Default Outcome correctly.



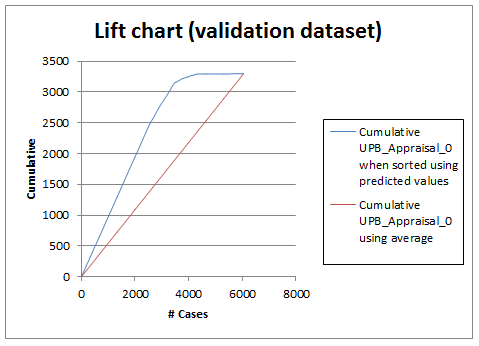
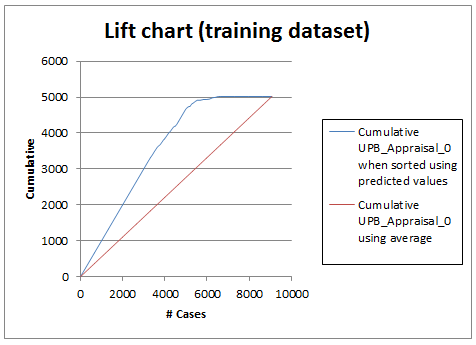
The overall % error rate is 7.829 and model classifies 3141 as Default Outcome and 2439 as NON Default Outcome correctly.

Best Pruned tree:

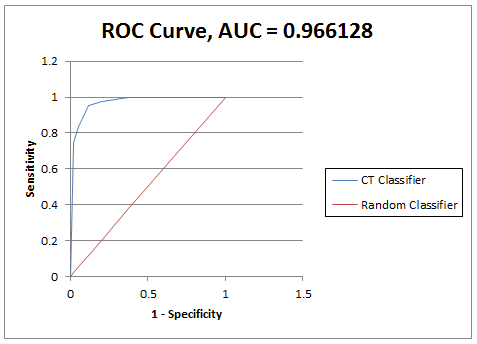
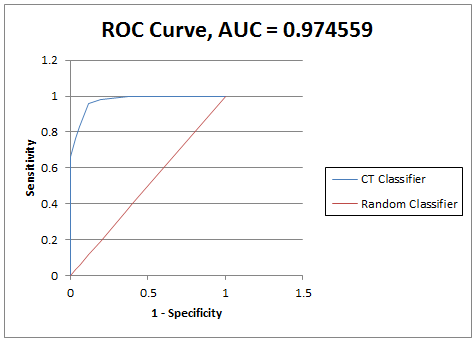




**Lift Chart:**

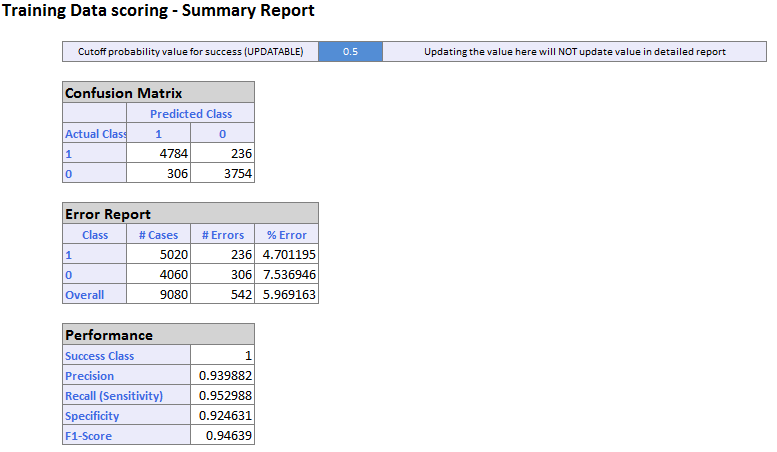


**ROC Curve:**

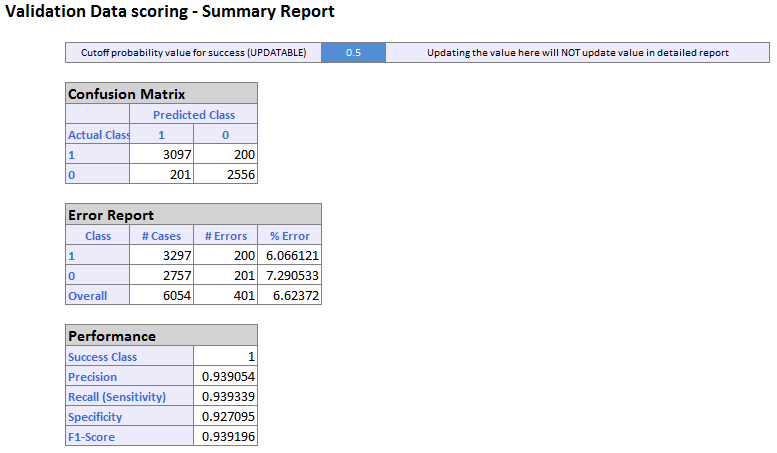


**RANDOM TREE:**

The cutoff probability value for success was kept at 0.5.

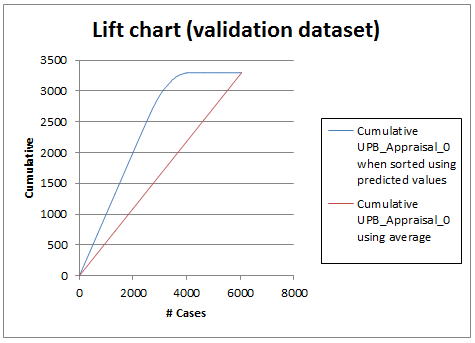
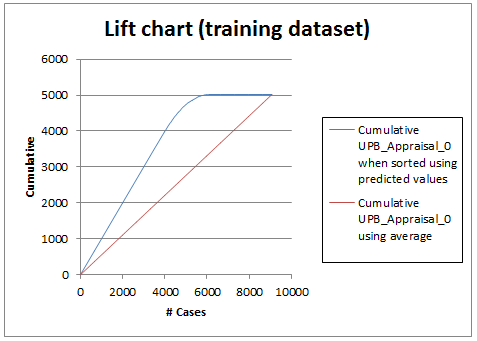


The overall % error rate 5.96 and model classifies 4784 as Default Outcome and 3754 as NON Default Outcome correctly.

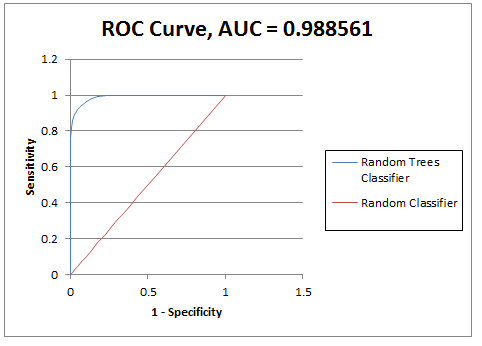
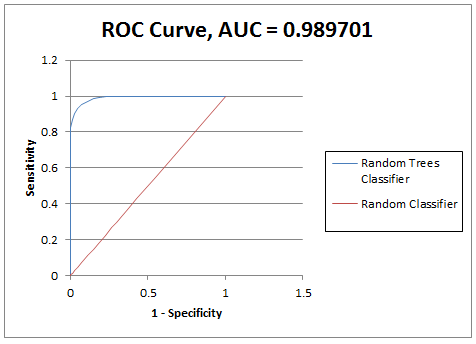


The overall % error rate is 6.623 and model classifies 3097 as Default Outcome and 2556 as NON Default Outcome correctly.

**Lift Chart:**



**ROC Curve:**



**Comparison between the models:**

Based on overall percentage error for validation datasets and maximum area under the curve (ROC), Random Forest turns out to be the best model amongst all models.

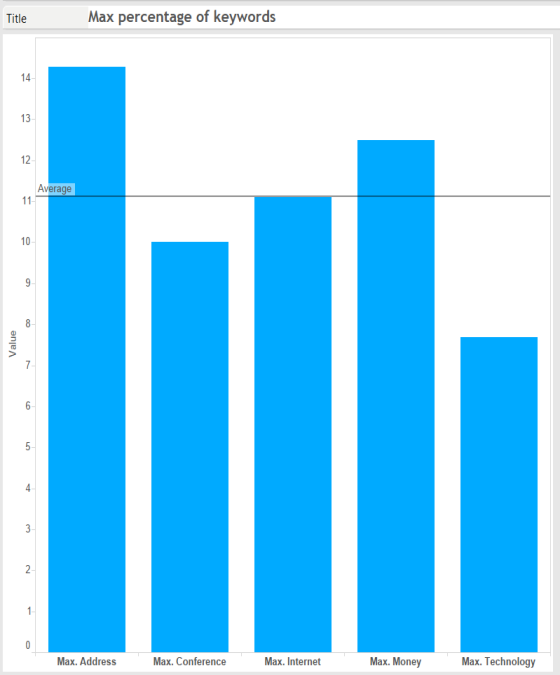
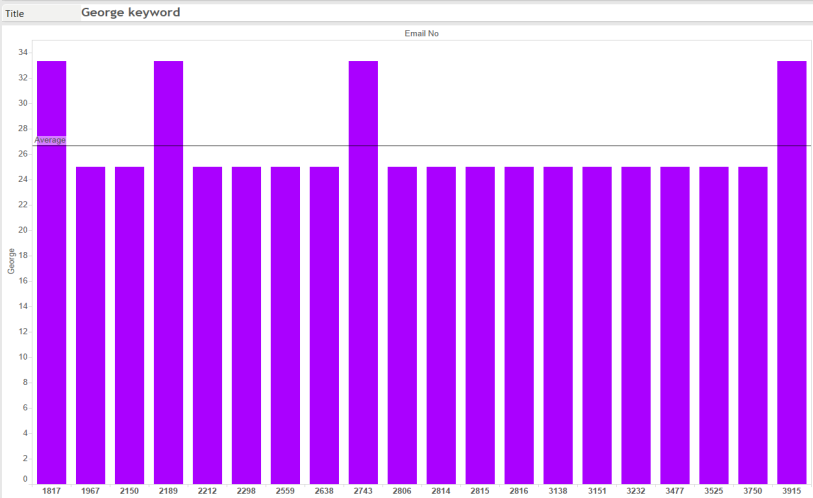
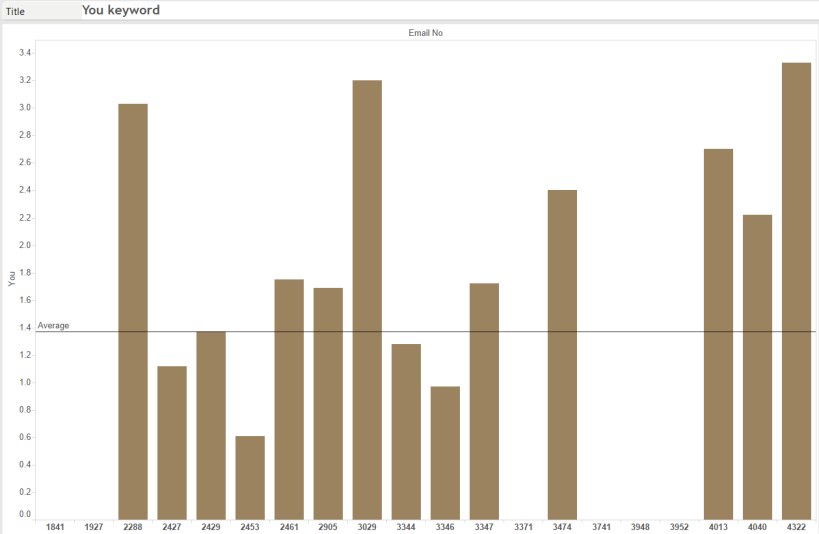
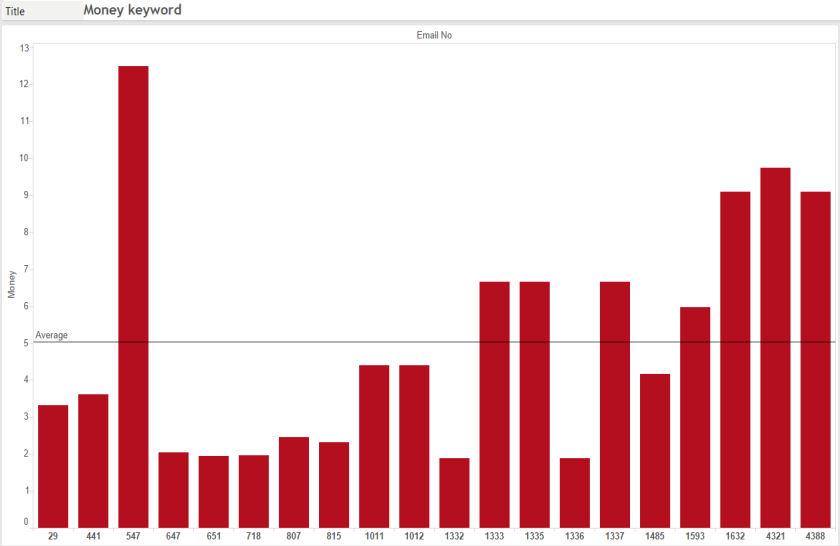
Considering detecting the Default Outcome, random forest model is useful because it gives minimum error rate and Outcome are classified in a better way as Default and Non-default. Logistic Regression gives 0% error rate but that model is ignored because it is too over-fitted a model.

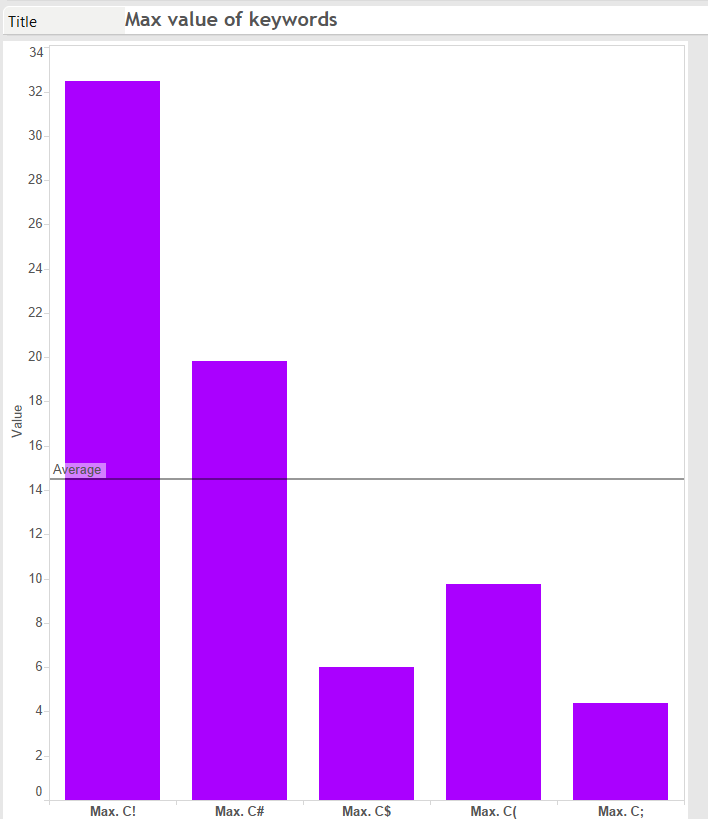
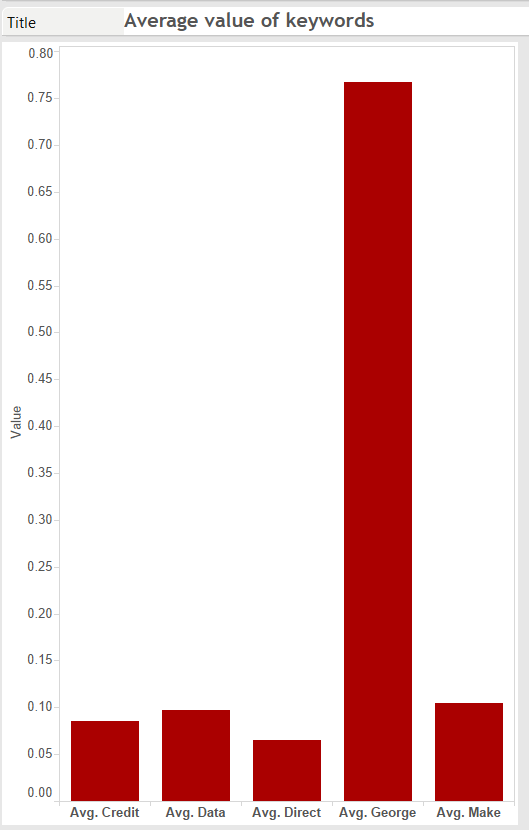
**Question 3:**

**Cleaning data:**

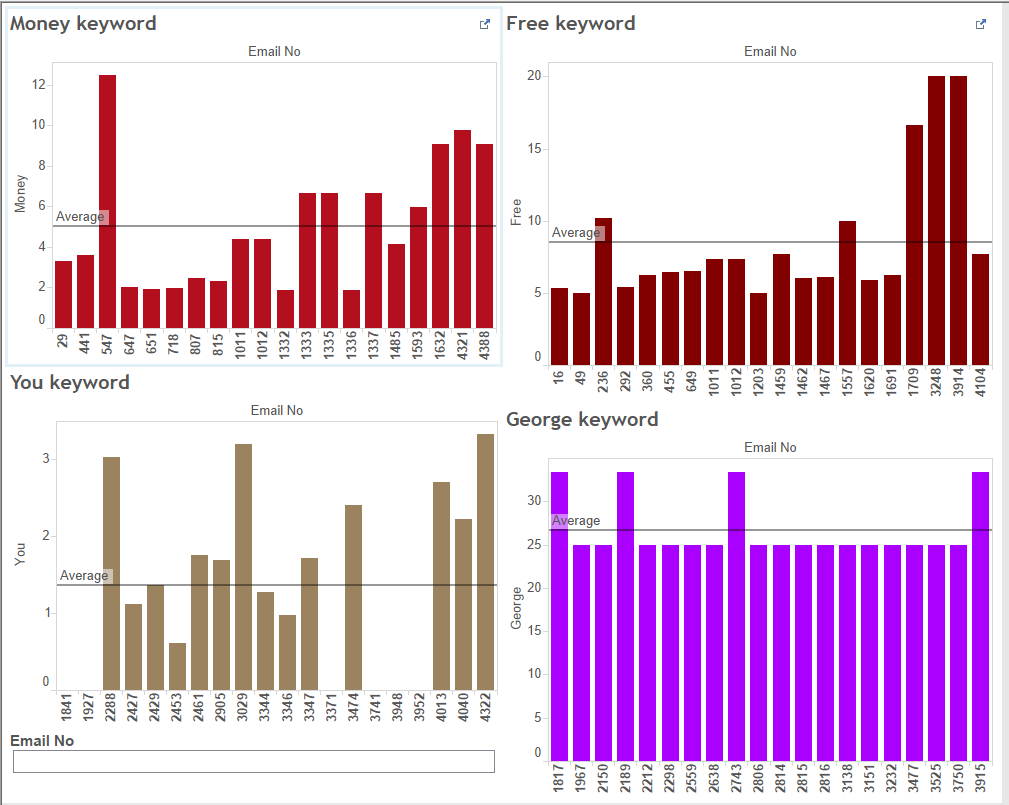
Imported the data set into Wrangler, default datatype for columns was integer. The datatype was changed to decimal to correct the mismatched values in the columns.

**Exploratory analysis:**

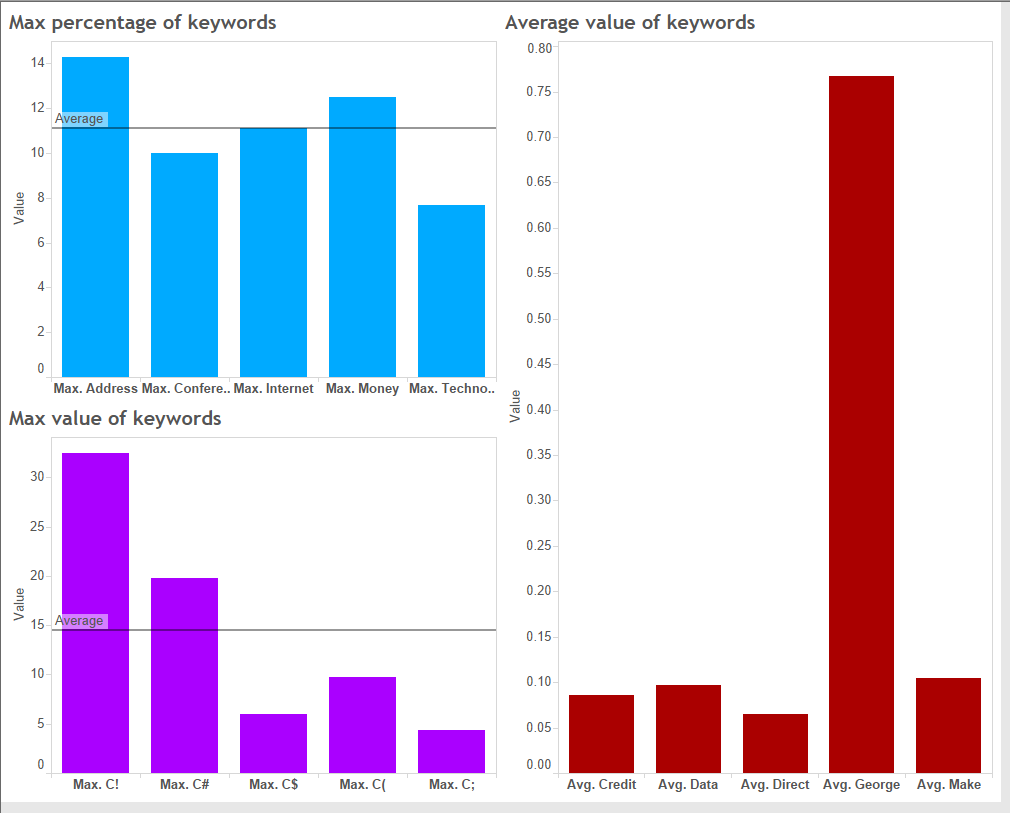




**Dashboard 1:**



**Dashboard 2:**



**Creating dummies:**

The only categorical variable is SPAM. Created dummy variables from the categorical variable SPAM.

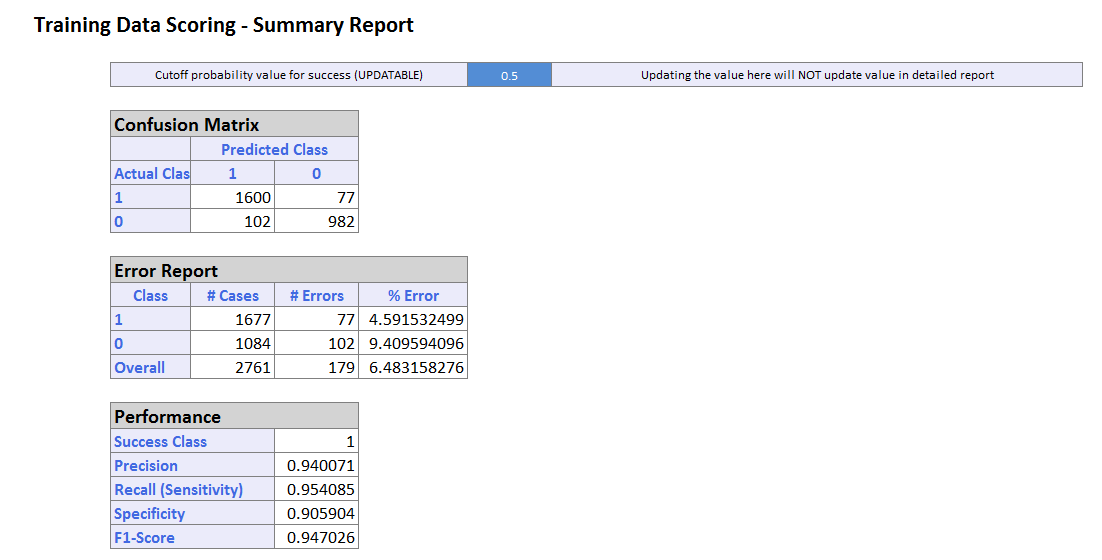
**Partitioning the data:**

Partitioned data into 60% and 40%.

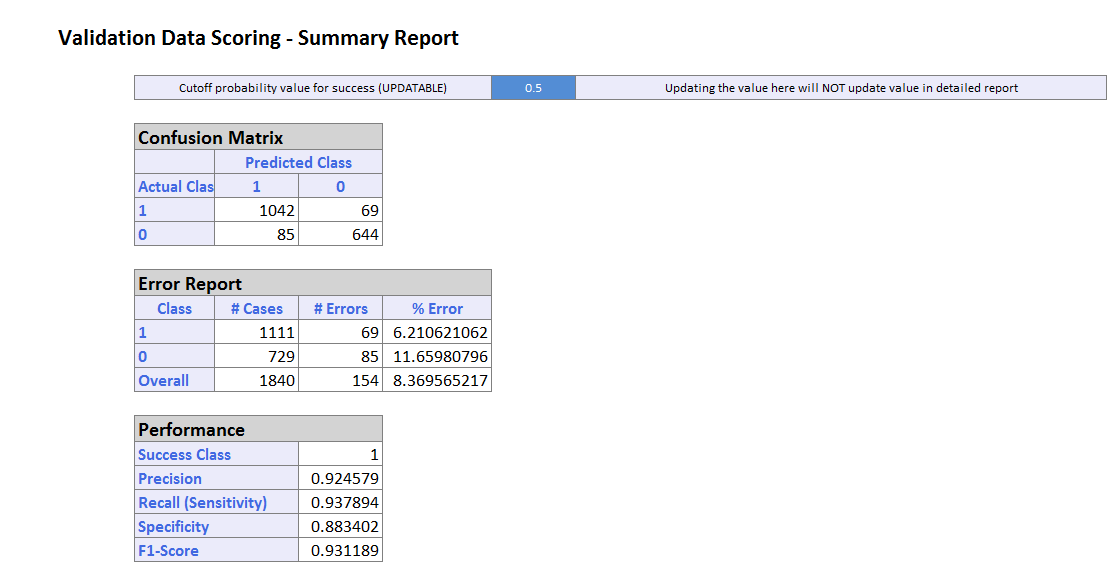
60% partition is used as training data and 40% partition is used as validation data.

**Logistic Regression:**

The cutoff probability value for success was kept at 0.5.

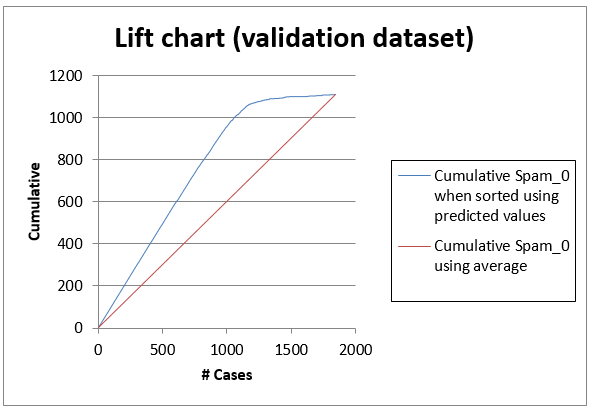
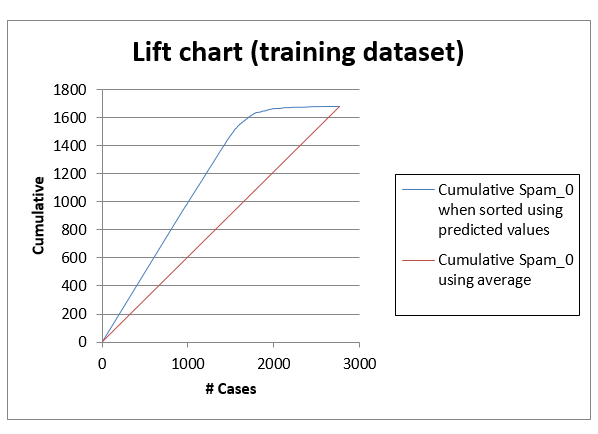


The overall % error rate is 6.48 and model classifies 1600 SPAM emails and 982 NON SPAM emails correctly.



The overall error rate 8.36% and the model classifies 1042 SPAM emails and 644 NON SPAM emails

**Lift Chart:**

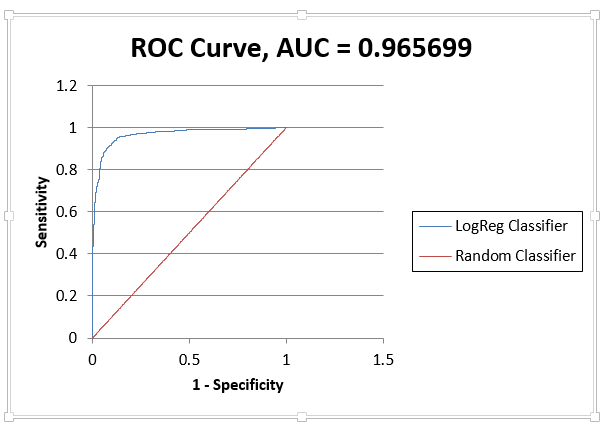
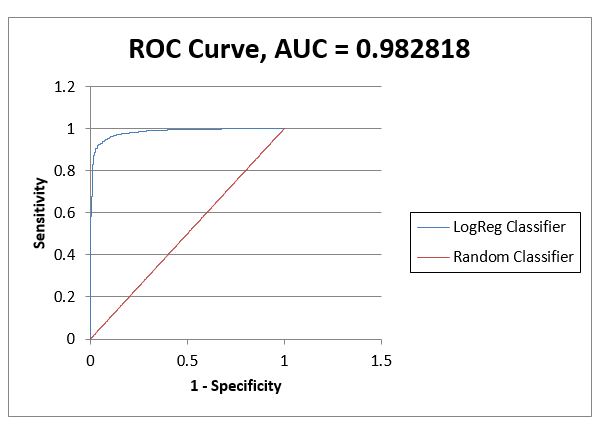


A lift chart is the graphical way to assess the predictive performance of a model. It compares the model with the baseline model that has no predictors.

Lift chart is based on 1800 validation records. It can be said that model’s predictive performance is better than baseline model.

**ROC Curve:**

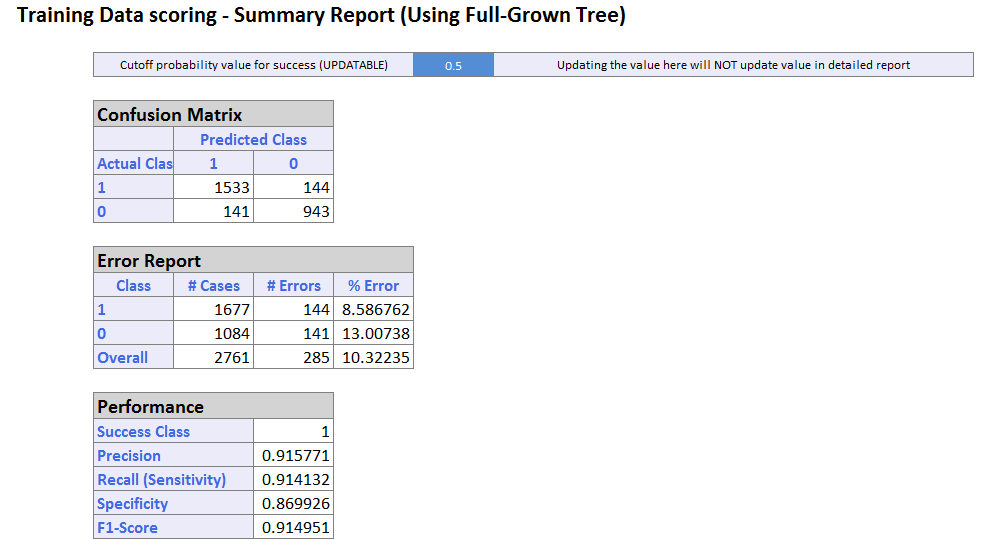
**Training dataset and Validation dataset:**



The ROC (Receiver Operating Characteristic) curve plots the sensitivity and 1-specificity. Better performance are given by the curves that are near the Y axis. Area under the Curve is 0.96 as shown in the figure.

**CART**

The cutoff probability value for success was kept at 0.5.

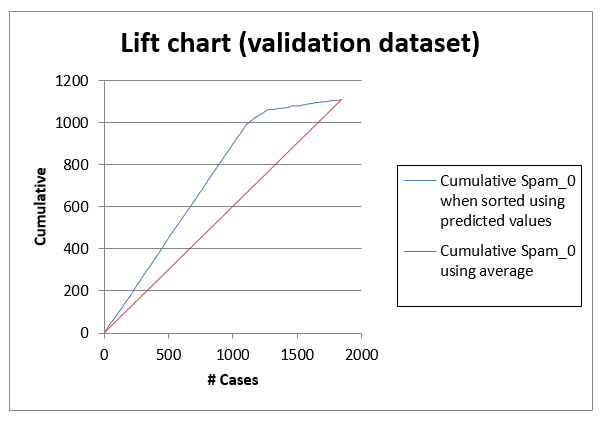
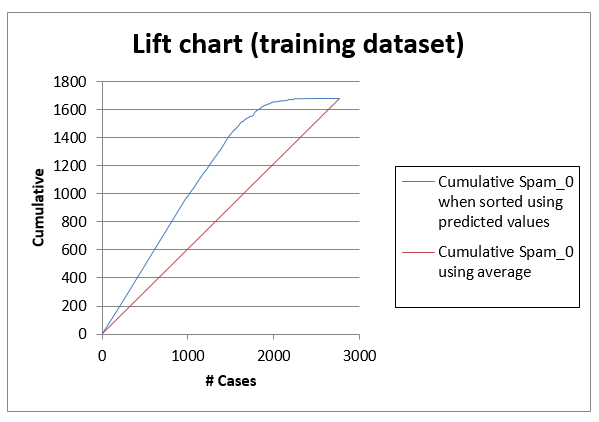


The overall % error rate is 10.32 and model classifies 1533 SPAM emails and 943 NON SPAM emails correctly.

The overall % error rate is 12.39 and model classifies 993 SPAM emails and 619 NON SPAM emails correctly.

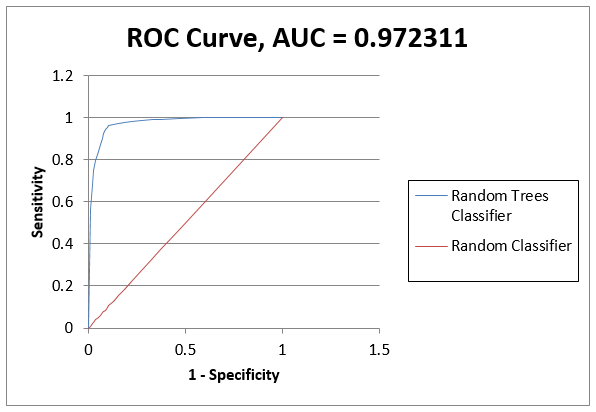
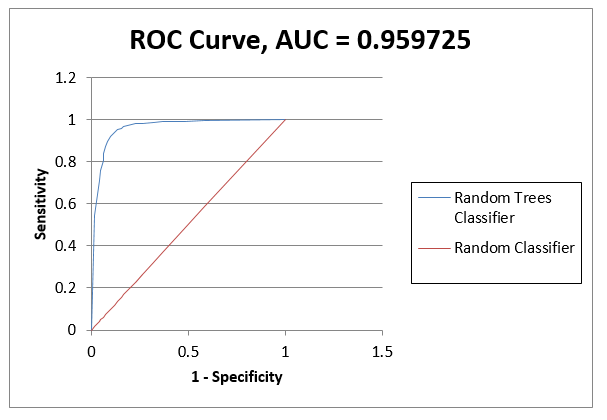
**Lift Chart:**

**Training dataset and Validation dataset:**



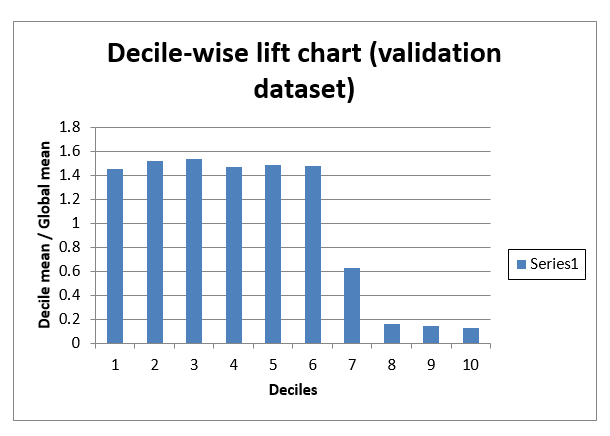
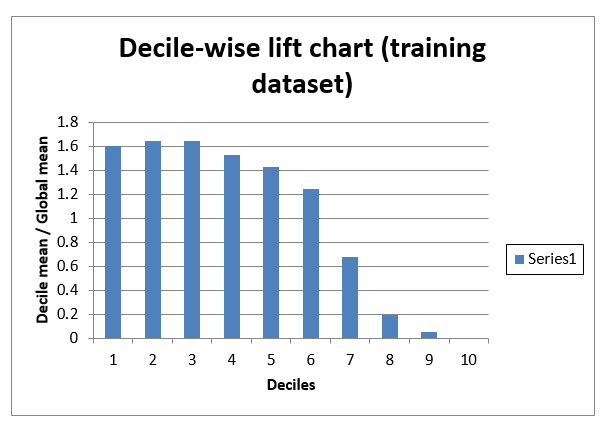
**ROC Curve:**

**Training dataset and Validation dataset:**

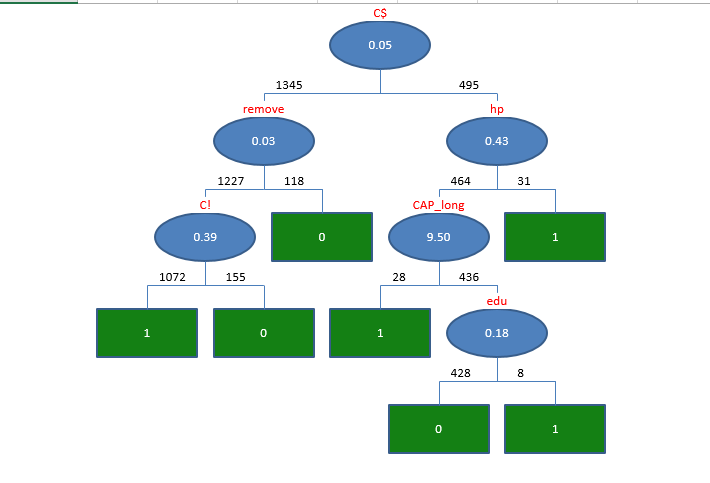
****

**Decile chart:**

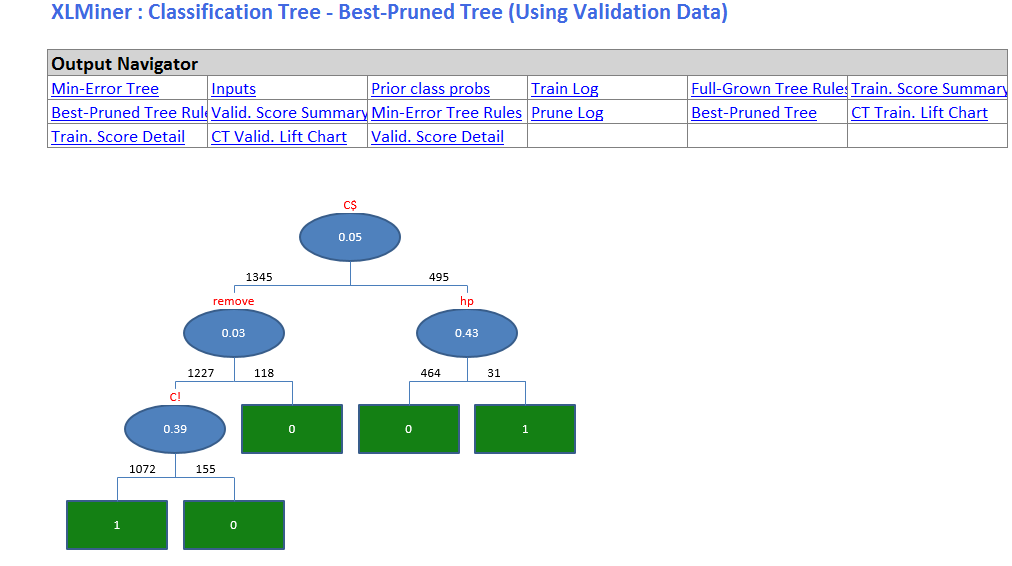
**Training dataset and Validation dataset:**



**Minimum Error Tree:**

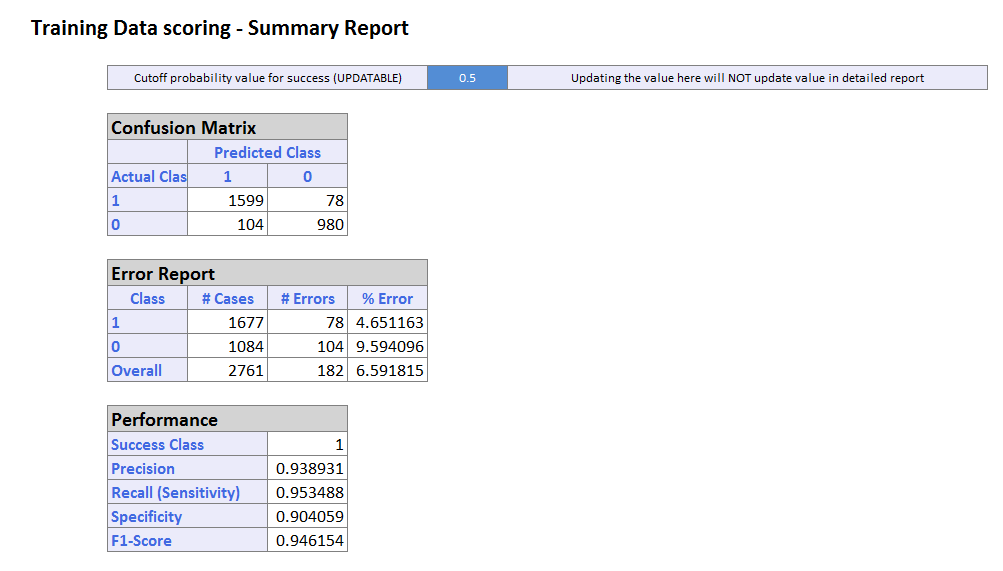


**Pruned tree:**

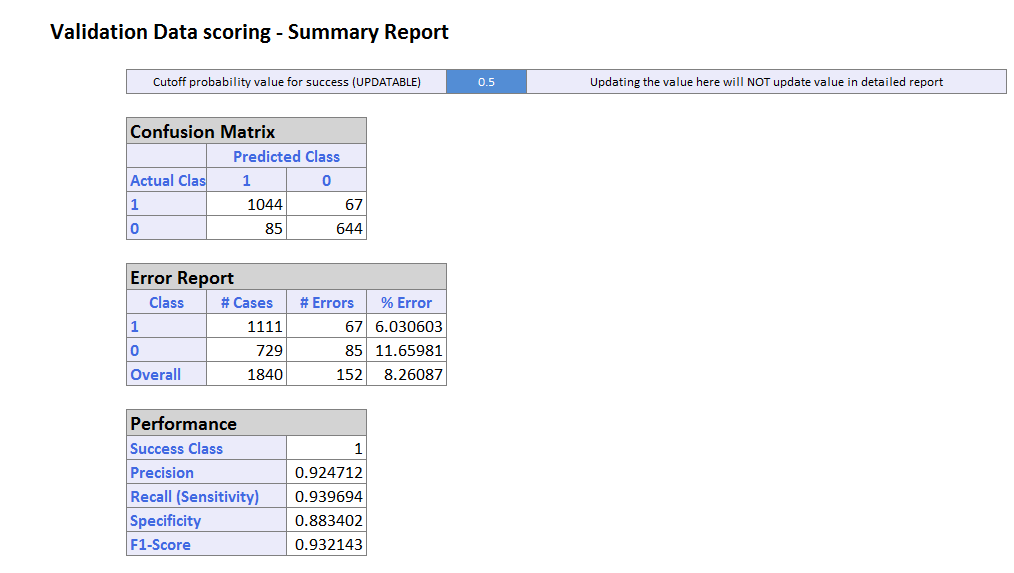


**Random Forest:**

The cutoff probability value for success was kept at 0.5.



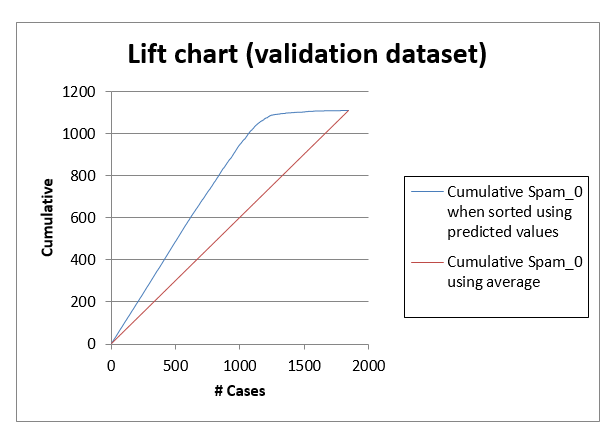
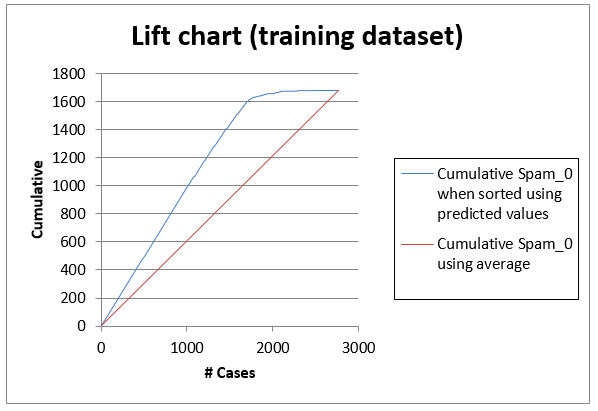
The overall % error rate is 6.59 and model classifies 1599 SPAM emails and 980 NON SPAM emails correctly.



The overall % error rate is 8.26 and model classifies 1044 SPAM emails and 644 NON SPAM emails correctly.

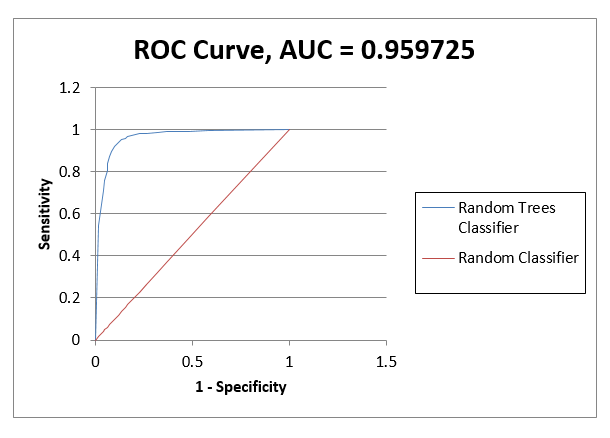
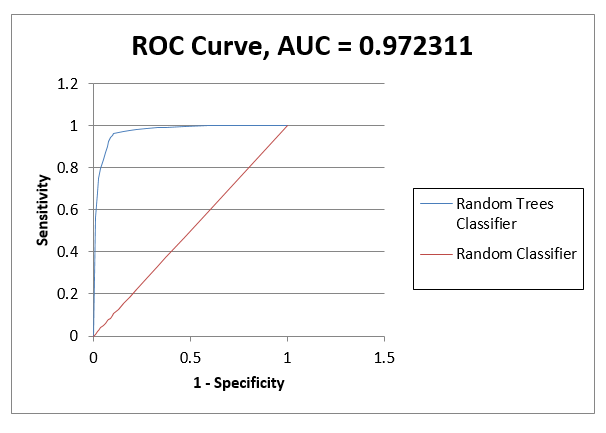
**Lift Chart:**

**Training dataset and Validation dataset:**



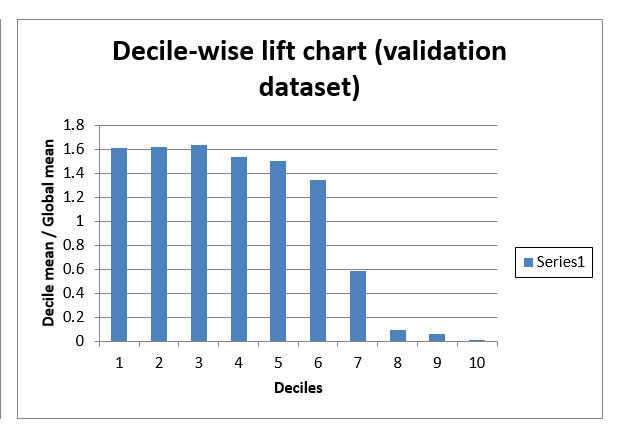
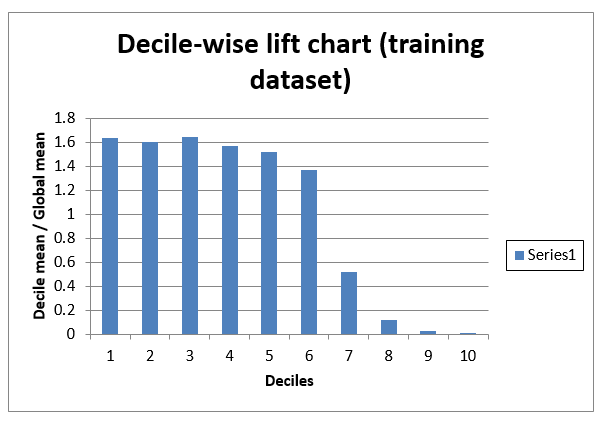
**ROC Curve:**

**Training dataset and Validation dataset:**



Decile **chart:**

**Training dataset and Validation dataset:**



**Comparison between the models:**

Based on overall percentage error for validation datasets and maximum area under the curve (ROC), Random Forest turns out to be the best model amongst all models.

Considering detecting the SPAM mails, random forest model is useful because it gives minimum error rate and emails are classified in a better way as SPAM and NON SPAM.

**Problem 4**

The blog feedback dataset contains following attributes:

1-50 : Average, standard deviation, min, max and median of the Attributes 51…60

51: Total number of comments before basetime

52: Number of comments in the last 24 hours before the basetime

53: Let T1 denote the datetime 48 hours before basetime, Let T2 denote the datetime 24 hours before basetime. This attribute is the number of comments in the time period between T1 and T2

54: Number of comments in the first 24 hours after the publication of the blog post, but before basetime

55: The difference of Attribute 52 and Attribute 53

56…60: The same features as the attributes 51…55, but features 56…60 refer to the number of links (trackbacks),

61: The length of time between the publication of the blog post and basetime

62: The length of the blog post

63…262: The 200 bag of words features for 200 frequent words of the text of the blog post

263…269: binary indicator features (0 or 1) for the weekday (Monday…Sunday) of the basetime

270…276: binary indicator features (0 or 1) for the weekday (Monday…Sunday) of the date of publication of the blog post

277: Number of parent pages: we consider a blog post P as a parent of blog post B, if B is a reply (trackback) to blog post P.

278…280: Minimum, maximum, average number of comments that the parents received

281: The target: the number of comments in the next 24 hours (relative to basetime)

Setting the work directory and reading the training data csv.

setwd("C:/Users/Mushtaq/Downloads/ADS/Week 4/R-tutorial")

blogtrain<-read.csv("blogData\_train.csv",header = T)

Lets create the test data:

blogtest1<-read.csv("blogData\_test-2012.02.01.00\_00.csv",header=T)

blogtest2<-read.csv("blogData\_test-2012.02.02.00\_00.csv",header=T)

names(blogtest2)<-names(blogtest1)

blogtest3<-read.csv("blogData\_test-2012.02.03.00\_00.csv",header=T)

names(blogtest3)<-names(blogtest1)

blogtest4<-read.csv("blogData\_test-2012.02.04.00\_00.csv",header=T)

names(blogtest4)<-names(blogtest1)

blogtest5<-read.csv("blogData\_test-2012.02.05.00\_00.csv",header=T)

names(blogtest5)<-names(blogtest1)

blogtest6<-read.csv("blogData\_test-2012.02.06.00\_00.csv",header=T)

names(blogtest6)<-names(blogtest1)

blogtest7<-read.csv("blogData\_test-2012.02.07.00\_00.csv",header=T)

names(blogtest7)<-names(blogtest1)

blogtest8<-read.csv("blogData\_test-2012.02.08.00\_00.csv",header=T)

names(blogtest8)<-names(blogtest1)

blogtest9<-read.csv("blogData\_test-2012.02.09.00\_00.csv",header=T)

names(blogtest9)<-names(blogtest1)

blogtest10<-read.csv("blogData\_test-2012.02.10.00\_00.csv",header=T)

names(blogtest10)<-names(blogtest1)

blogtest11<-read.csv("blogData\_test-2012.02.11.00\_00.csv",header=T)

names(blogtest11)<-names(blogtest1)

blogtest12<-read.csv("blogData\_test-2012.02.12.00\_00.csv",header=T)

names(blogtest12)<-names(blogtest1)

blogtest13<-read.csv("blogData\_test-2012.02.13.00\_00.csv",header=T)

names(blogtest13)<-names(blogtest1)

blogtest14<-read.csv("blogData\_test-2012.02.14.00\_00.csv",header=T)

names(blogtest14)<-names(blogtest1)

blogtest15<-read.csv("blogData\_test-2012.02.15.00\_00.csv",header=T)

names(blogtest15)<-names(blogtest1)

blogtest16<-read.csv("blogData\_test-2012.02.16.00\_00.csv",header=T)

names(blogtest16)<-names(blogtest1)

blogtest17<-read.csv("blogData\_test-2012.02.17.00\_00.csv",header=T)

names(blogtest17)<-names(blogtest1)

blogtest18<-read.csv("blogData\_test-2012.02.18.00\_00.csv",header=T)

names(blogtest18)<-names(blogtest1)

blogtest19<-read.csv("blogData\_test-2012.02.19.00\_00.csv",header=T)

names(blogtest19)<-names(blogtest1)

blogtest20<-read.csv("blogData\_test-2012.02.20.00\_00.csv",header=T)

names(blogtest20)<-names(blogtest1)

blogtest21<-read.csv("blogData\_test-2012.02.21.00\_00.csv",header=T)

names(blogtest21)<-names(blogtest1)

blogtest22<-read.csv("blogData\_test-2012.02.22.00\_00.csv",header=T)

names(blogtest22)<-names(blogtest1)

blogtest23<-read.csv("blogData\_test-2012.02.23.00\_00.csv",header=T)

names(blogtest23)<-names(blogtest1)

blogtest24<-read.csv("blogData\_test-2012.02.24.00\_00.csv",header=T)

names(blogtest24)<-names(blogtest1)

blogtest25<-read.csv("blogData\_test-2012.02.25.00\_00.csv",header=T)

names(blogtest25)<-names(blogtest1)

blogtest26<-read.csv("blogData\_test-2012.02.26.00\_00.csv",header=T)

names(blogtest26)<-names(blogtest1)

blogtest27<-read.csv("blogData\_test-2012.02.27.00\_00.csv",header=T)

names(blogtest27)<-names(blogtest1)

blogtest28<-read.csv("blogData\_test-2012.02.28.00\_00.csv",header=T)

names(blogtest28)<-names(blogtest1)

blogtest29<-read.csv("blogData\_test-2012.02.29.00\_00.csv",header=T)

names(blogtest29)<-names(blogtest1)

blogtest30<-read.csv("blogData\_test-2012.03.01.00\_00.csv",header=T)

names(blogtest30)<-names(blogtest1)

blogtest31<-read.csv("blogData\_test-2012.03.02.00\_00.csv",header=T)

names(blogtest31)<-names(blogtest1)

blogtest32<-read.csv("blogData\_test-2012.03.03.00\_00.csv",header=T)

names(blogtest32)<-names(blogtest1)

blogtest33<-read.csv("blogData\_test-2012.03.04.00\_00.csv",header=T)

names(blogtest33)<-names(blogtest1)

blogtest34<-read.csv("blogData\_test-2012.03.05.00\_00.csv",header=T)

names(blogtest34)<-names(blogtest1)

blogtest35<-read.csv("blogData\_test-2012.03.06.00\_00.csv",header=T)

names(blogtest35)<-names(blogtest1)

blogtest36<-read.csv("blogData\_test-2012.03.07.00\_00.csv",header=T)

names(blogtest36)<-names(blogtest1)

blogtest37<-read.csv("blogData\_test-2012.03.08.00\_00.csv",header=T)

names(blogtest37)<-names(blogtest1)

blogtest38<-read.csv("blogData\_test-2012.03.09.00\_00.csv",header=T)

names(blogtest38)<-names(blogtest1)

blogtest39<-read.csv("blogData\_test-2012.03.10.00\_00.csv",header=T)

names(blogtest39)<-names(blogtest1)

blogtest40<-read.csv("blogData\_test-2012.03.11.00\_00.csv",header=T)

names(blogtest40)<-names(blogtest1)

blogtest41<-read.csv("blogData\_test-2012.03.12.00\_00.csv",header=T)

names(blogtest41)<-names(blogtest1)

blogtest42<-read.csv("blogData\_test-2012.03.13.00\_00.csv",header=T)

names(blogtest42)<-names(blogtest1)

blogtest43<-read.csv("blogData\_test-2012.03.14.00\_00.csv",header=T)

names(blogtest43)<-names(blogtest1)

blogtest44<-read.csv("blogData\_test-2012.03.15.00\_00.csv",header=T)

names(blogtest44)<-names(blogtest1)

blogtest45<-read.csv("blogData\_test-2012.03.16.00\_00.csv",header=T)

names(blogtest45)<-names(blogtest1)

blogtest46<-read.csv("blogData\_test-2012.03.17.00\_00.csv",header=T)

names(blogtest46)<-names(blogtest1)

blogtest47<-read.csv("blogData\_test-2012.03.18.00\_00.csv",header=T)

names(blogtest47)<-names(blogtest1)

blogtest48<-read.csv("blogData\_test-2012.03.19.00\_00.csv",header=T)

names(blogtest48)<-names(blogtest1)

blogtest49<-read.csv("blogData\_test-2012.03.20.00\_00.csv",header=T)

names(blogtest49)<-names(blogtest1)

blogtest50<-read.csv("blogData\_test-2012.03.21.00\_00.csv",header=T)

names(blogtest50)<-names(blogtest1)

blogtest51<-read.csv("blogData\_test-2012.03.22.00\_00.csv",header=T)

names(blogtest51)<-names(blogtest1)

blogtest52<-read.csv("blogData\_test-2012.03.23.00\_00.csv",header=T)

names(blogtest52)<-names(blogtest1)

blogtest53<-read.csv("blogData\_test-2012.03.24.00\_00.csv",header=T)

names(blogtest53)<-names(blogtest1)

blogtest54<-read.csv("blogData\_test-2012.03.25.00\_00.csv",header=T)

names(blogtest54)<-names(blogtest1)

blogtest55<-read.csv("blogData\_test-2012.03.26.01\_00.csv",header=T)

names(blogtest55)<-names(blogtest1)

blogtest56<-read.csv("blogData\_test-2012.03.27.01\_00.csv",header=T)

names(blogtest56)<-names(blogtest1)

blogtest57<-read.csv("blogData\_test-2012.03.28.01\_00.csv",header=T)

names(blogtest57)<-names(blogtest1)

blogtest58<-read.csv("blogData\_test-2012.03.29.01\_00.csv",header=T)

names(blogtest58)<-names(blogtest1)

blogtest59<-read.csv("blogData\_test-2012.03.30.01\_00.csv",header=T)

names(blogtest59)<-names(blogtest1)

blogtest60<-read.csv("blogData\_test-2012.03.31.01\_00.csv",header=T)

names(blogtest60)<-names(blogtest1)

Combining all test csv files into 1.

blogtest<-rbind(blogtest1,blogtest2,blogtest3,blogtest4,blogtest5,blogtest6,blogtest7,blogtest8,blogtest9,blogtest10,blogtest11,blogtest12,blogtest13,blogtest14,blogtest15,blogtest16,blogtest17,blogtest18,blogtest19,blogtest20,blogtest21,blogtest22,blogtest23,blogtest24,blogtest25,blogtest26,blogtest27,blogtest28,blogtest29,blogtest30,blogtest31,blogtest32,blogtest33,blogtest34,blogtest35,blogtest36,blogtest37,blogtest38,blogtest39,blogtest40,blogtest41,blogtest42,blogtest43,blogtest44,blogtest45,blogtest46,blogtest47,blogtest48,blogtest49,blogtest50,blogtest51,blogtest52,blogtest53,blogtest54,blogtest55,blogtest56,blogtest57,blogtest58,blogtest59,blogtest60)

names(blogtest)<-names(blogtrain)

The target is the number of feedbacks that the blog-entry will receive in the next H hours. Most regression algorithms assume that the instances are vectors. Furthermore, it is assumed that the value of the target is known for some (sufficiently enough) instances, and based on this information, we want to predict the value of the target for those cases where it is unknown. First, using the cases where the target is known, a prediction model, regressor, is constructed. Then, the regressor is used to predict the value of the target for the instances with unknown valued target.

Now we will extract some features from the document and build models from those features:

1. Basic features: Number of links and comments
2. Textual features: The most discriminative bag of words features
3. Weekday features: Binary indicator features that describe on which day of the week the main text of the document was published and for which day of the week the prediction has to be calculated

We will create three models as follows: 1. Using only basic features. 2. Using basic and weekday features. 3. Using basic and textual features.

Applying multi linear Regression,

Taking the columns of only basic features,

blogtrain\_basic<-blogtrain[,c(51:60,281)]

blogtest\_basic<-blogtest[,c(51:60,281)]

Creating the regression model using lm()

regression\_basic<-lm(blogtrain\_basic$X1.0.2~.,data=blogtrain\_basic)

regression\_basic

##

## Call:

## lm(formula = blogtrain\_basic$X1.0.2 ~ ., data = blogtrain\_basic)

##

## Coefficients:

## (Intercept) X2.0.1 X2.0.2 X0.0.14 X2.0.3

## 2.13989 0.08007 0.29936 -0.01698 -0.08807

## X2.0.4 X0.0.15 X0.0.16 X0.0.17 X0.0.18

## NA -2.99623 1.12627 0.22766 3.21444

## X0.0.19

## NA

summary(regression\_basic)

##

## Call:

## lm(formula = blogtrain\_basic$X1.0.2 ~ ., data = blogtrain\_basic)

##

## Residuals:

## Min 1Q Median 3Q Max

## -288.85 -2.51 -2.14 -1.60 1421.28

##

## Coefficients: (2 not defined because of singularities)

## Estimate Std. Error t value Pr(>|t|)

## (Intercept) 2.139890 0.156311 13.690 < 2e-16 \*\*\*

## X2.0.1 0.080070 0.007243 11.055 < 2e-16 \*\*\*

## X2.0.2 0.299357 0.004620 64.793 < 2e-16 \*\*\*

## X0.0.14 -0.016978 0.005273 -3.220 0.001282 \*\*

## X2.0.3 -0.088074 0.007404 -11.896 < 2e-16 \*\*\*

## X2.0.4 NA NA NA NA

## X0.0.15 -2.996227 0.596173 -5.026 5.03e-07 \*\*\*

## X0.0.16 1.126266 0.301542 3.735 0.000188 \*\*\*

## X0.0.17 0.227656 0.317478 0.717 0.473329

## X0.0.18 3.214440 0.586787 5.478 4.32e-08 \*\*\*

## X0.0.19 NA NA NA NA

## ---

## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

##

## Residual standard error: 33.19 on 52387 degrees of freedom

## Multiple R-squared: 0.2255, Adjusted R-squared: 0.2254

## F-statistic: 1906 on 8 and 52387 DF, p-value: < 2.2e-16

Time to predict the model using test data,

predictions\_basic=predict.lm(regression\_basic,blogtest\_basic)

## Warning in predict.lm(regression\_basic, blogtest\_basic): prediction from a

## rank-deficient fit may be misleading

summary(predictions\_basic)

## Min. 1st Qu. Median Mean 3rd Qu. Max.

## -19.010 2.140 2.140 6.040 3.305 381.500

Now lets build the model by using basic and weekday features.

blogtrain\_basicweekday<-blogtrain[,c(51:60,270:276,281)]

blogtest\_basicweekday<-blogtest[,c(51:60,270:276,281)]

Creating the regression model using lm()

regression\_basicweekday<-lm(blogtrain\_basicweekday$X1.0.2~.,data=blogtrain\_basicweekday)

summary(regression\_basicweekday)

##

## Call:

## lm(formula = blogtrain\_basicweekday$X1.0.2 ~ ., data = blogtrain\_basicweekday)

##

## Residuals:

## Min 1Q Median 3Q Max

## -288.66 -2.81 -2.11 -1.60 1420.70

##

## Coefficients: (3 not defined because of singularities)

## Estimate Std. Error t value Pr(>|t|)

## (Intercept) 2.358680 0.488838 4.825 1.40e-06 \*\*\*

## X2.0.1 0.079686 0.007244 11.001 < 2e-16 \*\*\*

## X2.0.2 0.299144 0.004621 64.737 < 2e-16 \*\*\*

## X0.0.14 -0.017121 0.005273 -3.247 0.001167 \*\*

## X2.0.3 -0.087358 0.007408 -11.792 < 2e-16 \*\*\*

## X2.0.4 NA NA NA NA

## X0.0.15 -3.014114 0.596276 -5.055 4.32e-07 \*\*\*

## X0.0.16 1.129878 0.301546 3.747 0.000179 \*\*\*

## X0.0.17 0.234097 0.317485 0.737 0.460914

## X0.0.18 3.228334 0.586902 5.501 3.80e-08 \*\*\*

## X0.0.19 NA NA NA NA

## X0.0.227 -0.482932 0.607259 -0.795 0.426463

## X0.0.228 -0.174627 0.601699 -0.290 0.771647

## X0.0.229 -0.630579 0.599466 -1.052 0.292850

## X1.0.1 -0.676346 0.605175 -1.118 0.263741

## X0.0.230 0.361581 0.610383 0.592 0.553596

## X0.0.231 0.472711 0.674555 0.701 0.483447

## X0.0.232 NA NA NA NA

## ---

## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

##

## Residual standard error: 33.19 on 52381 degrees of freedom

## Multiple R-squared: 0.2256, Adjusted R-squared: 0.2254

## F-statistic: 1090 on 14 and 52381 DF, p-value: < 2.2e-16

Time to predict the model using test data,

predictions\_basicweekday=predict.lm(regression\_basicweekday,blogtest\_basicweekday)

## Warning in predict.lm(regression\_basicweekday, blogtest\_basicweekday):

## prediction from a rank-deficient fit may be misleading

summary(predictions\_basicweekday)

## Min. 1st Qu. Median Mean 3rd Qu. Max.

## -18.470 1.876 2.359 6.016 3.354 381.200

Now lets build the model by using basic and textual features.

blogtrain\_basictextual<-blogtrain[,c(51:60,63:262,281)]

blogtest\_basictextual<-blogtest[,c(51:60,63:262,281)]

Creating the regression model using lm()

regression\_basictextual<-lm(blogtrain\_basictextual$X1.0.2~.,data=blogtrain\_basictextual)

summary(regression\_basictextual)

##

## Call:

## lm(formula = blogtrain\_basictextual$X1.0.2 ~ ., data = blogtrain\_basictextual)

##

## Residuals:

## Min 1Q Median 3Q Max

## -282.48 -3.97 -1.28 -0.03 1422.58

##

## Coefficients: (24 not defined because of singularities)

## Estimate Std. Error t value Pr(>|t|)

## (Intercept) 0.847466 0.257653 3.289 0.00101 \*\*

## X2.0.1 0.069047 0.007252 9.522 < 2e-16 \*\*\*

## X2.0.2 0.296531 0.004633 64.003 < 2e-16 \*\*\*

## X0.0.14 -0.016684 0.005281 -3.159 0.00158 \*\*

## X2.0.3 -0.079922 0.007438 -10.745 < 2e-16 \*\*\*

## X2.0.4 NA NA NA NA

## X0.0.15 -3.021667 0.595823 -5.071 3.96e-07 \*\*\*

## X0.0.16 1.063574 0.300987 3.534 0.00041 \*\*\*

## X0.0.17 0.189247 0.316930 0.597 0.55043

## X0.0.18 3.263229 0.586404 5.565 2.64e-08 \*\*\*

## X0.0.19 NA NA NA NA

## X0.0.21 1.634112 3.727680 0.438 0.66112

## X0.0.22 0.254961 0.432977 0.589 0.55596

## X0.0.23 91.005797 78.572981 1.158 0.24677

## X0.0.24 -1.773220 9.628151 -0.184 0.85388

## X0.0.25 0.083152 0.404283 0.206 0.83704

## X0.0.26 -0.690242 0.449794 -1.535 0.12489

## X0.0.27 1.114260 0.403991 2.758 0.00582 \*\*

## X0.0.28 -2.114230 3.018516 -0.700 0.48367

## X0.0.29 -2.472069 2.567957 -0.963 0.33572

## X0.0.30 1.125333 0.922655 1.220 0.22260

## X0.0.31 NA NA NA NA

## X0.0.32 -2.666658 6.273503 -0.425 0.67079

## X0.0.33 -0.032282 2.250852 -0.014 0.98856

## X0.0.34 NA NA NA NA

## X0.0.35 2.137198 1.278763 1.671 0.09467 .

## X0.0.36 -1.331027 3.992618 -0.333 0.73885

## X0.0.37 -0.068728 0.416024 -0.165 0.86879

## X0.0.38 NA NA NA NA

## X0.0.39 NA NA NA NA

## X0.0.40 -2.349633 2.645408 -0.888 0.37444

## X0.0.41 -3.175498 3.894631 -0.815 0.41487

## X0.0.42 2.886833 13.591126 0.212 0.83179

## X0.0.43 17.084271 9.808063 1.742 0.08154 .

## X0.0.44 -0.614141 1.038096 -0.592 0.55412

## X0.0.45 -16.593124 14.067201 -1.180 0.23818

## X0.0.46 2.547900 1.380803 1.845 0.06501 .

## X0.0.47 -1.560294 0.629263 -2.480 0.01316 \*

## X0.0.48 -1.223018 19.153603 -0.064 0.94909

## X0.0.49 -4.037526 9.776702 -0.413 0.67963

## X0.0.50 27.407495 11.065273 2.477 0.01326 \*

## X0.0.51 -2.680592 2.145608 -1.249 0.21155

## X0.0.52 NA NA NA NA

## X0.0.53 -1.955333 13.565093 -0.144 0.88539

## X0.0.54 0.027526 0.543952 0.051 0.95964

## X0.0.55 NA NA NA NA

## X0.0.56 10.426064 8.107211 1.286 0.19844

## X0.0.57 -3.407066 8.676028 -0.393 0.69454

## X0.0.58 -2.820556 4.863682 -0.580 0.56197

## X0.0.59 0.143375 0.495200 0.290 0.77218

## X0.0.60 2.374018 0.575715 4.124 3.74e-05 \*\*\*

## X0.0.61 0.153296 0.873826 0.175 0.86074

## X0.0.62 -0.617930 0.729507 -0.847 0.39697

## X0.0.63 1.712855 2.727239 0.628 0.52997

## X0.0.64 -5.503326 4.261663 -1.291 0.19659

## X0.0.65 -4.604709 2.798933 -1.645 0.09994 .

## X0.0.66 -1.721541 1.183777 -1.454 0.14588

## X0.0.67 -2.466834 19.230565 -0.128 0.89793

## X0.0.68 -0.884397 0.949867 -0.931 0.35182

## X0.0.69 -2.124992 13.545456 -0.157 0.87534

## X0.0.70 -4.718432 7.867139 -0.600 0.54867

## X0.0.71 -6.453293 9.777555 -0.660 0.50925

## X0.0.72 -0.978981 0.456613 -2.144 0.03204 \*

## X0.0.73 0.100316 0.611783 0.164 0.86975

## X0.0.74 -0.074516 1.041480 -0.072 0.94296

## X0.0.75 -2.804185 3.725117 -0.753 0.45159

## X0.0.76 -0.048611 1.365339 -0.036 0.97160

## X0.0.77 -2.013840 0.703618 -2.862 0.00421 \*\*

## X0.0.78 -0.148625 0.442138 -0.336 0.73676

## X0.0.79 0.293380 0.803951 0.365 0.71517

## X0.0.80 -0.816797 0.402981 -2.027 0.04268 \*

## X0.0.81 -3.215536 7.696830 -0.418 0.67611

## X0.0.82 22.634374 19.743769 1.146 0.25163

## X0.0.83 7.316953 1.474151 4.964 6.95e-07 \*\*\*

## X0.0.84 8.584089 7.598044 1.130 0.25858

## X0.0.85 -3.248480 1.991073 -1.632 0.10279

## X0.0.86 0.180822 1.334370 0.136 0.89221

## X0.0.87 0.224165 19.127505 0.012 0.99065

## X0.0.88 NA NA NA NA

## X0.0.89 -2.243392 2.145862 -1.045 0.29582

## X0.0.90 NA NA NA NA

## X0.0.91 0.235560 6.615317 0.036 0.97159

## X0.0.92 0.839249 0.784453 1.070 0.28469

## X0.0.93 -0.508768 0.924780 -0.550 0.58222

## X0.0.94 0.529696 0.722997 0.733 0.46378

## X0.0.95 3.644237 2.430729 1.499 0.13382

## X0.0.96 2.265292 0.878289 2.579 0.00991 \*\*

## X0.0.97 0.227358 0.404013 0.563 0.57361

## X0.0.98 -0.396488 0.578616 -0.685 0.49320

## X0.0.99 -0.027882 0.583320 -0.048 0.96188

## X0.0.100 -2.026450 2.996871 -0.676 0.49892

## X0.0.101 -0.111257 0.575560 -0.193 0.84672

## X0.0.102 -1.266460 1.675217 -0.756 0.44965

## X0.0.103 -2.905214 2.650662 -1.096 0.27307

## X0.0.104 -3.555265 2.672780 -1.330 0.18347

## X0.0.105 -3.047814 2.091392 -1.457 0.14504

## X0.0.106 -10.292553 11.378679 -0.905 0.36571

## X0.0.107 NA NA NA NA

## X0.0.108 -1.988575 3.146963 -0.632 0.52745

## X0.0.109 0.559923 0.384870 1.455 0.14572

## X0.0.110 -3.746630 3.287103 -1.140 0.25438

## X0.0.111 -1.071550 0.712518 -1.504 0.13262

## X0.0.112 8.345845 0.846661 9.857 < 2e-16 \*\*\*

## X0.0.113 -8.210874 19.144145 -0.429 0.66800

## X0.0.114 NA NA NA NA

## X0.0.115 -1.331595 1.613187 -0.825 0.40912

## X0.0.116 0.787977 0.395376 1.993 0.04627 \*

## X0.0.117 0.470369 0.521865 0.901 0.36742

## X0.0.118 -3.082727 4.554453 -0.677 0.49850

## X0.0.119 NA NA NA NA

## X0.0.120 -1.371580 0.892809 -1.536 0.12448

## X0.0.121 0.305643 0.935719 0.327 0.74394

## X0.0.122 -1.298528 1.990649 -0.652 0.51420

## X0.0.123 -5.526628 4.214885 -1.311 0.18979

## X0.0.124 NA NA NA NA

## X0.0.125 -4.128919 4.255301 -0.970 0.33190

## X0.0.126 -5.180522 10.540813 -0.491 0.62309

## X0.0.127 NA NA NA NA

## X0.0.128 -1.084212 0.439125 -2.469 0.01355 \*

## X0.0.129 11.787163 5.389462 2.187 0.02874 \*

## X0.0.130 NA NA NA NA

## X0.0.131 -3.876353 9.628152 -0.403 0.68724

## X0.0.132 -1.305413 1.251473 -1.043 0.29691

## X0.0.133 2.375786 2.869448 0.828 0.40770

## X0.0.134 -4.713524 4.526261 -1.041 0.29771

## X0.0.135 -2.614264 2.446024 -1.069 0.28517

## X0.0.136 -4.509309 7.836970 -0.575 0.56503

## X0.0.137 NA NA NA NA

## X0.0.138 0.410619 0.538753 0.762 0.44597

## X0.0.139 -0.972983 0.753665 -1.291 0.19671

## X0.0.140 -1.135507 1.495399 -0.759 0.44766

## X0.0.141 -0.157578 0.594603 -0.265 0.79100

## X0.0.142 2.602058 0.512454 5.078 3.83e-07 \*\*\*

## X0.0.143 1.842455 8.647188 0.213 0.83127

## X0.0.144 -1.630391 1.263717 -1.290 0.19700

## X0.0.145 -1.354175 1.086279 -1.247 0.21254

## X0.0.146 -2.491832 1.965856 -1.268 0.20496

## X0.0.147 -0.054989 1.237245 -0.044 0.96455

## X0.0.148 NA NA NA NA

## X0.0.149 1.351434 0.509246 2.654 0.00796 \*\*

## X0.0.150 -1.047553 1.176250 -0.891 0.37315

## X0.0.151 0.522369 0.475811 1.098 0.27228

## X0.0.152 0.922745 0.452164 2.041 0.04128 \*

## X0.0.153 4.331742 1.051065 4.121 3.77e-05 \*\*\*

## X0.0.154 0.805796 1.149698 0.701 0.48338

## X0.0.155 -0.494108 0.593075 -0.833 0.40478

## X0.0.156 NA NA NA NA

## X0.0.157 3.043259 9.715558 0.313 0.75410

## X0.0.158 NA NA NA NA

## X0.0.159 -3.234341 2.411269 -1.341 0.17981

## X0.0.160 0.449904 0.441576 1.019 0.30827

## X0.0.161 -3.886129 5.382312 -0.722 0.47029

## X0.0.162 -2.769249 4.119496 -0.672 0.50144

## X0.0.163 1.031387 1.464316 0.704 0.48122

## X0.0.164 2.941547 2.303257 1.277 0.20156

## X0.0.165 1.692388 0.709554 2.385 0.01708 \*

## X0.0.166 1.722219 1.370116 1.257 0.20876

## X0.0.167 -2.778269 4.383498 -0.634 0.52621

## X0.0.168 0.996222 0.398155 2.502 0.01235 \*

## X0.0.169 -1.652474 2.963244 -0.558 0.57708

## X0.0.170 NA NA NA NA

## X0.0.171 -0.741194 0.410733 -1.805 0.07115 .

## X0.0.172 4.079053 7.250672 0.563 0.57373

## X0.0.173 1.535999 2.343600 0.655 0.51221

## X0.0.174 -1.019412 0.625632 -1.629 0.10323

## X0.0.175 -5.777633 6.442594 -0.897 0.36984

## X0.0.176 4.241928 3.210602 1.321 0.18643

## X0.0.177 4.032759 0.910521 4.429 9.48e-06 \*\*\*

## X0.0.178 -2.133954 3.567160 -0.598 0.54969

## X0.0.179 1.529311 0.753262 2.030 0.04234 \*

## X0.0.180 -3.828366 4.027636 -0.951 0.34185

## X0.0.181 -4.936244 19.948654 -0.247 0.80456

## X0.0.182 -2.904884 9.708709 -0.299 0.76479

## X0.0.183 -1.161862 1.847412 -0.629 0.52941

## X0.0.184 -0.457575 0.414406 -1.104 0.26952

## X0.0.185 1.576523 2.031672 0.776 0.43777

## X0.0.186 -1.799460 0.467737 -3.847 0.00012 \*\*\*

## X0.0.187 -3.463498 3.802995 -0.911 0.36244

## X0.0.188 -0.370631 0.487494 -0.760 0.44709

## X0.0.189 -3.229403 2.287928 -1.411 0.15810

## X0.0.190 1.190554 0.395009 3.014 0.00258 \*\*

## X0.0.191 0.153285 0.446654 0.343 0.73146

## X0.0.192 -1.226597 0.766750 -1.600 0.10966

## X0.0.193 -0.584177 4.078497 -0.143 0.88611

## X0.0.194 NA NA NA NA

## X0.0.195 -1.706313 2.087091 -0.818 0.41361

## X0.0.196 -3.493958 3.164329 -1.104 0.26952

## X0.0.197 -3.065526 5.066545 -0.605 0.54515

## X0.0.198 -2.344982 3.499284 -0.670 0.50278

## X0.0.199 1.807940 0.917461 1.971 0.04878 \*

## X0.0.200 -1.653323 1.277912 -1.294 0.19575

## X0.0.201 NA NA NA NA

## X0.0.202 -6.637217 19.371897 -0.343 0.73189

## X0.0.203 -0.667033 1.054266 -0.633 0.52693

## X0.0.204 0.357972 0.568618 0.630 0.52899

## X0.0.205 -0.142610 0.480832 -0.297 0.76678

## X0.0.206 0.707324 0.412222 1.716 0.08619 .

## X0.0.207 -1.459573 1.759229 -0.830 0.40673

## X0.0.208 NA NA NA NA

## X0.0.209 -2.748764 1.220132 -2.253 0.02427 \*

## X0.0.210 4.552534 1.859707 2.448 0.01437 \*

## X0.0.211 -16.134111 13.617969 -1.185 0.23612

## X0.0.212 0.374160 0.946792 0.395 0.69271

## X0.0.213 -0.611701 1.330699 -0.460 0.64574

## X0.0.214 -3.411174 11.182840 -0.305 0.76034

## X0.0.215 0.475560 0.781941 0.608 0.54307

## X0.0.216 -1.969372 3.127245 -0.630 0.52886

## X0.0.217 0.278107 2.549547 0.109 0.91314

## X0.0.218 0.624875 2.905938 0.215 0.82974

## X0.0.219 1.563589 1.623596 0.963 0.33553

## X0.0.220 -8.435505 9.585418 -0.880 0.37884

## ---

## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

##

## Residual standard error: 33.08 on 52209 degrees of freedom

## Multiple R-squared: 0.2333, Adjusted R-squared: 0.2306

## F-statistic: 85.4 on 186 and 52209 DF, p-value: < 2.2e-16

Time to predict the model using test data,

predictions\_basictextual=predict.lm(regression\_basictextual,blogtest\_basictextual)

## Warning in predict.lm(regression\_basictextual, blogtest\_basictextual):

## prediction from a rank-deficient fit may be misleading

summary(predictions\_basictextual)

## Min. 1st Qu. Median Mean 3rd Qu. Max.

## -28.7200 0.8475 1.7870 5.8950 5.1320 376.5000

Lets continue building the model using CART algorithm. Importing all the reuired libraries

**library**(rpart)

**library**(rpart.plot)

## Warning: package 'rpart.plot' was built under R version 3.2.4

**library**(ROCR)

## Warning: package 'ROCR' was built under R version 3.2.4

## Loading required package: gplots

## Warning: package 'gplots' was built under R version 3.2.4

##

## Attaching package: 'gplots'

## The following object is masked from 'package:stats':

##

## lowess

**library**(party)

## Warning: package 'party' was built under R version 3.2.4

## Loading required package: grid

## Loading required package: mvtnorm

## Loading required package: modeltools

## Loading required package: stats4

## Loading required package: strucchange

## Warning: package 'strucchange' was built under R version 3.2.4

## Loading required package: zoo

##

## Attaching package: 'zoo'

## The following objects are masked from 'package:base':

##

## as.Date, as.Date.numeric

## Loading required package: sandwich

Lets create CART model using basic features only

blogtrain\_basic<-blogtrain[,c(51:60,281)]

blogtest\_basic<-blogtest[,c(51:60,281)]

Creating the CART model using rpart

tree\_basic<-rpart(blogtrain\_basic$X1.0.2~.,data=blogtrain\_basic, method = "anova",control=rpart.control(cp=0.001))

printcp(tree\_basic)

##

## Regression tree:

## rpart(formula = blogtrain\_basic$X1.0.2 ~ ., data = blogtrain\_basic,

## method = "anova", control = rpart.control(cp = 0.001))

##

## Variables actually used in tree construction:

## [1] X0.0.14 X0.0.15 X0.0.16 X0.0.18 X0.0.19 X2.0.1 X2.0.2 X2.0.3 X2.0.4

##

## Root node error: 74495814/52396 = 1421.8

##

## n= 52396

##

## CP nsplit rel error xerror xstd

## 1 0.1601367 0 1.00000 1.00002 0.066871

## 2 0.0356080 1 0.83986 0.84826 0.061039

## 3 0.0208948 2 0.80426 0.80975 0.059376

## 4 0.0089661 3 0.78336 0.79373 0.058969

## 5 0.0048979 4 0.77439 0.78757 0.059284

## 6 0.0045895 5 0.76950 0.78643 0.059365

## 7 0.0043841 6 0.76491 0.78946 0.059430

## 8 0.0035830 7 0.76052 0.78468 0.059286

## 9 0.0024613 11 0.74619 0.78365 0.059307

## 10 0.0024404 13 0.74127 0.78696 0.059280

## 11 0.0023383 15 0.73639 0.78715 0.059274

## 12 0.0021951 16 0.73405 0.78913 0.059329

## 13 0.0020115 17 0.73185 0.78757 0.059237

## 14 0.0017869 19 0.72783 0.78578 0.059151

## 15 0.0017494 24 0.71852 0.78507 0.059155

## 16 0.0012456 25 0.71677 0.78380 0.059163

## 17 0.0012077 29 0.71179 0.79028 0.059260

## 18 0.0011379 31 0.70937 0.79199 0.059199

## 19 0.0011267 32 0.70823 0.79224 0.059137

## 20 0.0010000 34 0.70598 0.79065 0.058687

Calculating the best complexity parameter of the model

bestcp\_basic <-tree\_basic$cptable[which.min(tree\_basic$cptable[,"xerror"]),"CP"]

bestcp\_basic

## [1] 0.002461266

Pruning the tree to avoid overfitting

tree.pruned\_basic <- prune(tree\_basic, cp=bestcp\_basic)

summary(tree.pruned\_basic)

## Call:

## rpart(formula = blogtrain\_basic$X1.0.2 ~ ., data = blogtrain\_basic,

## method = "anova", control = rpart.control(cp = 0.001))

## n= 52396

##

## CP nsplit rel error xerror xstd

## 1 0.160136685 0 1.0000000 1.0000151 0.06687132

## 2 0.035607988 1 0.8398633 0.8482630 0.06103885

## 3 0.020894750 2 0.8042553 0.8097497 0.05937646

## 4 0.008966101 3 0.7833606 0.7937338 0.05896866

## 5 0.004897880 4 0.7743945 0.7875707 0.05928377

## 6 0.004589499 5 0.7694966 0.7864310 0.05936519

## 7 0.004384053 6 0.7649071 0.7894595 0.05942969

## 8 0.003582984 7 0.7605230 0.7846781 0.05928645

## 9 0.002461266 11 0.7461911 0.7836451 0.05930677

##

## Variable importance

## X2.0.2 X2.0.4 X2.0.3 X2.0.1 X0.0.16 X0.0.19 X0.0.18 X0.0.15

## 42 36 6 6 3 3 2 2

##

## Node number 1: 52396 observations, complexity param=0.1601367

## mean=6.764829, MSE=1421.784

## left son=2 (51595 obs) right son=3 (801 obs)

## Primary splits:

## X2.0.2 < 221.5 to the left, improve=0.16013670, (0 missing)

## X2.0.4 < 185.5 to the left, improve=0.12456960, (0 missing)

## X2.0.3 < 279.5 to the left, improve=0.07712658, (0 missing)

## X2.0.1 < 352.5 to the left, improve=0.07535665, (0 missing)

## X0.0.16 < 2.5 to the left, improve=0.04953726, (0 missing)

## Surrogate splits:

## X2.0.4 < 221.5 to the left, agree=0.996, adj=0.760, (0 split)

## X2.0.1 < 818.5 to the left, agree=0.986, adj=0.055, (0 split)

## X2.0.3 < 788 to the left, agree=0.985, adj=0.029, (0 split)

## X0.0.16 < 9.5 to the left, agree=0.985, adj=0.019, (0 split)

## X0.0.19 < 10.5 to the left, agree=0.985, adj=0.017, (0 split)

##

## Node number 2: 51595 observations, complexity param=0.03560799

## mean=4.884756, MSE=843.6107

## left son=4 (45388 obs) right son=5 (6207 obs)

## Primary splits:

## X2.0.4 < 7.5 to the left, improve=0.06094381, (0 missing)

## X2.0.2 < 20.5 to the left, improve=0.05892590, (0 missing)

## X2.0.3 < 14.5 to the left, improve=0.02121635, (0 missing)

## X2.0.1 < 14.5 to the left, improve=0.02009537, (0 missing)

## X0.0.19 < 0.5 to the left, improve=0.01385694, (0 missing)

## Surrogate splits:

## X2.0.2 < 14.5 to the left, agree=0.938, adj=0.483, (0 split)

## X0.0.19 < 1.5 to the left, agree=0.891, adj=0.092, (0 split)

## X0.0.16 < 1.5 to the left, agree=0.888, adj=0.068, (0 split)

##

## Node number 3: 801 observations, complexity param=0.02089475

## mean=127.8664, MSE=23770.55

## left son=6 (575 obs) right son=7 (226 obs)

## Primary splits:

## X2.0.2 < 427.5 to the left, improve=0.08175181, (0 missing)

## X2.0.4 < 421 to the left, improve=0.07196278, (0 missing)

## X2.0.3 < 430 to the left, improve=0.06611954, (0 missing)

## X2.0.1 < 430.5 to the left, improve=0.05480402, (0 missing)

## X0.0.18 < 1.5 to the left, improve=0.04451982, (0 missing)

## Surrogate splits:

## X2.0.4 < 427.5 to the left, agree=0.939, adj=0.783, (0 split)

## X2.0.3 < 427.5 to the left, agree=0.843, adj=0.442, (0 split)

## X2.0.1 < 427.5 to the left, agree=0.813, adj=0.336, (0 split)

## X0.0.16 < 7.5 to the left, agree=0.757, adj=0.137, (0 split)

## X0.0.19 < 7.5 to the left, agree=0.757, adj=0.137, (0 split)

##

## Node number 4: 45388 observations, complexity param=0.00489788

## mean=2.233167, MSE=336.933

## left son=8 (45220 obs) right son=9 (168 obs)

## Primary splits:

## X2.0.2 < 107.5 to the left, improve=0.023859180, (0 missing)

## X2.0.1 < 400.5 to the left, improve=0.011253080, (0 missing)

## X0.0.14 < 187.5 to the left, improve=0.009131713, (0 missing)

## X2.0.4 < 1.5 to the left, improve=0.009126476, (0 missing)

## X2.0.3 < 343.5 to the left, improve=0.007968743, (0 missing)

## Surrogate splits:

## X0.0.14 < 655.5 to the left, agree=0.996, adj=0.018, (0 split)

##

## Node number 5: 6207 observations, complexity param=0.004589499

## mean=24.27421, MSE=4121.272

## left son=10 (3562 obs) right son=11 (2645 obs)

## Primary splits:

## X2.0.4 < 42.5 to the left, improve=0.013365470, (0 missing)

## X2.0.2 < 103.5 to the left, improve=0.011946390, (0 missing)

## X2.0.3 < 42.5 to the left, improve=0.011160650, (0 missing)

## X2.0.1 < 42.5 to the left, improve=0.009678477, (0 missing)

## X0.0.14 < 0.5 to the right, improve=0.007272552, (0 missing)

## Surrogate splits:

## X2.0.2 < 42.5 to the left, agree=0.973, adj=0.936, (0 split)

## X2.0.3 < 42.5 to the left, agree=0.955, adj=0.895, (0 split)

## X2.0.1 < 42.5 to the left, agree=0.951, adj=0.885, (0 split)

## X0.0.19 < 2.5 to the left, agree=0.616, adj=0.099, (0 split)

## X0.0.16 < 2.5 to the left, agree=0.614, adj=0.095, (0 split)

##

## Node number 6: 575 observations, complexity param=0.003582984

## mean=100.2296, MSE=16396.05

## left son=12 (184 obs) right son=13 (391 obs)

## Primary splits:

## X2.0.3 < 276.5 to the left, improve=0.02100116, (0 missing)

## X0.0.18 < 0.5 to the left, improve=0.01977664, (0 missing)

## X0.0.15 < 0.5 to the left, improve=0.01941526, (0 missing)

## X2.0.1 < 353.5 to the left, improve=0.01664092, (0 missing)

## X2.0.2 < 354.5 to the left, improve=0.01289905, (0 missing)

## Surrogate splits:

## X2.0.1 < 276.5 to the left, agree=0.951, adj=0.848, (0 split)

## X2.0.2 < 276.5 to the left, agree=0.880, adj=0.625, (0 split)

## X2.0.4 < 276.5 to the left, agree=0.718, adj=0.120, (0 split)

##

## Node number 7: 226 observations, complexity param=0.008966101

## mean=198.1814, MSE=35645.6

## left son=14 (61 obs) right son=15 (165 obs)

## Primary splits:

## X0.0.18 < 2.5 to the left, improve=0.08291271, (0 missing)

## X0.0.15 < 2.5 to the left, improve=0.07893757, (0 missing)

## X0.0.16 < 2.5 to the left, improve=0.05969577, (0 missing)

## X2.0.2 < 666.5 to the left, improve=0.04538495, (0 missing)

## X0.0.19 < 2.5 to the left, improve=0.04426379, (0 missing)

## Surrogate splits:

## X0.0.15 < 2.5 to the left, agree=0.996, adj=0.984, (0 split)

## X0.0.16 < 2.5 to the left, agree=0.889, adj=0.590, (0 split)

## X0.0.19 < 2.5 to the left, agree=0.850, adj=0.443, (0 split)

## X2.0.3 < 433.5 to the left, agree=0.743, adj=0.049, (0 split)

## X2.0.1 < 433.5 to the left, agree=0.739, adj=0.033, (0 split)

##

## Node number 8: 45220 observations

## mean=2.060349, MSE=311.8869

##

## Node number 9: 168 observations

## mean=48.75, MSE=4906.652

##

## Node number 10: 3562 observations

## mean=17.87872, MSE=3641.125

##

## Node number 11: 2645 observations

## mean=32.88696, MSE=4638.619

##

## Node number 12: 184 observations

## mean=73.17935, MSE=13359.72

##

## Node number 13: 391 observations, complexity param=0.003582984

## mean=112.9591, MSE=17318.54

## left son=26 (106 obs) right son=27 (285 obs)

## Primary splits:

## X0.0.18 < 0.5 to the left, improve=0.04192850, (0 missing)

## X0.0.15 < 0.5 to the left, improve=0.04103568, (0 missing)

## X0.0.19 < 0.5 to the left, improve=0.02890781, (0 missing)

## X0.0.16 < 0.5 to the left, improve=0.02322103, (0 missing)

## X2.0.1 < 288.5 to the right, improve=0.02304655, (0 missing)

## Surrogate splits:

## X0.0.15 < 0.5 to the left, agree=0.997, adj=0.991, (0 split)

## X0.0.16 < 0.5 to the left, agree=0.847, adj=0.434, (0 split)

##

## Node number 14: 61 observations, complexity param=0.004384053

## mean=108.7705, MSE=39642.87

## left son=28 (54 obs) right son=29 (7 obs)

## Primary splits:

## X2.0.4 < 702.5 to the left, improve=0.13505570, (0 missing)

## X2.0.2 < 694 to the left, improve=0.07957718, (0 missing)

## X2.0.3 < 722 to the left, improve=0.06185540, (0 missing)

## X0.0.14 < 5.5 to the right, improve=0.05540664, (0 missing)

## X2.0.1 < 764.5 to the right, improve=0.03749789, (0 missing)

## Surrogate splits:

## X2.0.2 < 694 to the left, agree=0.967, adj=0.714, (0 split)

## X2.0.3 < 722 to the left, agree=0.951, adj=0.571, (0 split)

## X2.0.1 < 934 to the left, agree=0.902, adj=0.143, (0 split)

##

## Node number 15: 165 observations

## mean=231.2364, MSE=30119.72

##

## Node number 26: 106 observations

## mean=68.77358, MSE=8636.892

##

## Node number 27: 285 observations, complexity param=0.003582984

## mean=129.393, MSE=19551.28

## left son=54 (263 obs) right son=55 (22 obs)

## Primary splits:

## X2.0.1 < 291.5 to the right, improve=0.04152467, (0 missing)

## X2.0.3 < 290 to the right, improve=0.03675561, (0 missing)

## X0.0.14 < 0.5 to the right, improve=0.02339020, (0 missing)

## X2.0.2 < 288.5 to the right, improve=0.01363084, (0 missing)

## X0.0.17 < 1.5 to the right, improve=0.01008088, (0 missing)

## Surrogate splits:

## X2.0.3 < 288.5 to the right, agree=0.986, adj=0.818, (0 split)

##

## Node number 28: 54 observations

## mean=82.42593, MSE=22460.02

##

## Node number 29: 7 observations

## mean=312, MSE=125540

##

## Node number 54: 263 observations

## mean=121.1521, MSE=16487.41

##

## Node number 55: 22 observations, complexity param=0.003582984

## mean=227.9091, MSE=45661.17

## left son=110 (13 obs) right son=111 (9 obs)

## Primary splits:

## X2.0.4 < 282.5 to the left, improve=0.35277120, (0 missing)

## X2.0.2 < 284 to the left, improve=0.25578780, (0 missing)

## X2.0.3 < 284.5 to the left, improve=0.25578780, (0 missing)

## X2.0.1 < 282.5 to the left, improve=0.15519920, (0 missing)

## X0.0.18 < 2.5 to the right, improve=0.06292435, (0 missing)

## Surrogate splits:

## X2.0.2 < 282.5 to the left, agree=0.955, adj=0.889, (0 split)

## X2.0.3 < 284.5 to the left, agree=0.955, adj=0.889, (0 split)

## X2.0.1 < 282.5 to the left, agree=0.909, adj=0.778, (0 split)

##

## Node number 110: 13 observations

## mean=122.3077, MSE=17903.14

##

## Node number 111: 9 observations

## mean=380.4444, MSE=46381.14

Predict the model using test data

blogtrain\_basic<-blogtrain\_basic[-c(7565:52396),]

pred\_basic<-predict(tree.pruned\_basic,blogtest\_basic,type = "vector")

Developing the confusion matrix

conf.matrix\_basic<-table(blogtrain\_basic$X1.0.2,pred\_basic)

rownames(conf.matrix\_basic) <- paste("Actual", rownames(conf.matrix\_basic), sep=":")

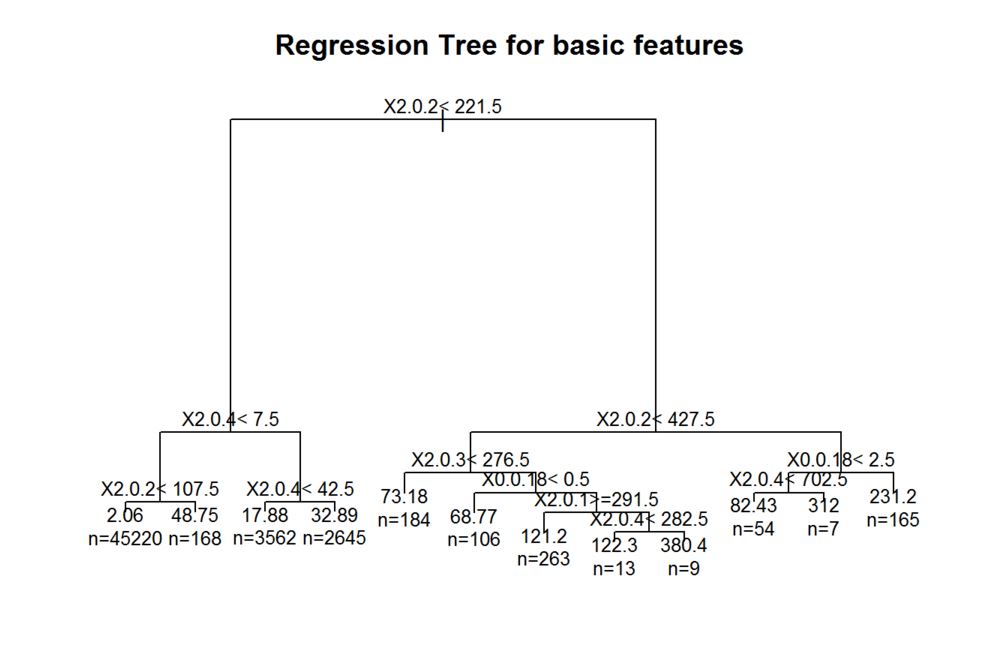
colnames(conf.matrix\_basic) <- paste("Predicted", colnames(conf.matrix\_basic), sep=":")

*#print(conf.matrix\_basic)*

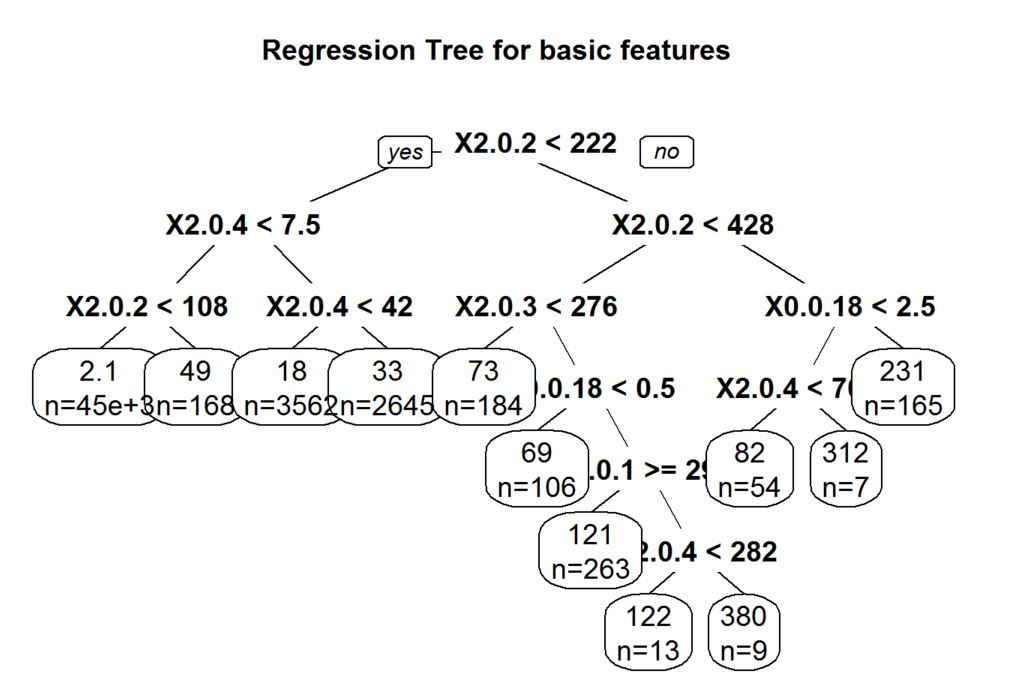
Plotting the regression tree for basic features

plot(tree.pruned\_basic, main="Regression Tree for basic features")

text(tree.pruned\_basic,cex=0.8,use.n=TRUE,xpd=TRUE)



prp(tree.pruned\_basic,faclen=0,tweak=1.5,extra=1,main="Regression Tree for basic features")



Finding out the R squared value

tmp\_basic<-printcp(tree\_basic)

##

## Regression tree:

## rpart(formula = blogtrain\_basic$X1.0.2 ~ ., data = blogtrain\_basic,

## method = "anova", control = rpart.control(cp = 0.001))

##

## Variables actually used in tree construction:

## [1] X0.0.14 X0.0.15 X0.0.16 X0.0.18 X0.0.19 X2.0.1 X2.0.2 X2.0.3 X2.0.4

##

## Root node error: 74495814/52396 = 1421.8

##

## n= 52396

##

## CP nsplit rel error xerror xstd

## 1 0.1601367 0 1.00000 1.00002 0.066871

## 2 0.0356080 1 0.83986 0.84826 0.061039

## 3 0.0208948 2 0.80426 0.80975 0.059376

## 4 0.0089661 3 0.78336 0.79373 0.058969

## 5 0.0048979 4 0.77439 0.78757 0.059284

## 6 0.0045895 5 0.76950 0.78643 0.059365

## 7 0.0043841 6 0.76491 0.78946 0.059430

## 8 0.0035830 7 0.76052 0.78468 0.059286

## 9 0.0024613 11 0.74619 0.78365 0.059307

## 10 0.0024404 13 0.74127 0.78696 0.059280

## 11 0.0023383 15 0.73639 0.78715 0.059274

## 12 0.0021951 16 0.73405 0.78913 0.059329

## 13 0.0020115 17 0.73185 0.78757 0.059237

## 14 0.0017869 19 0.72783 0.78578 0.059151

## 15 0.0017494 24 0.71852 0.78507 0.059155

## 16 0.0012456 25 0.71677 0.78380 0.059163

## 17 0.0012077 29 0.71179 0.79028 0.059260

## 18 0.0011379 31 0.70937 0.79199 0.059199

## 19 0.0011267 32 0.70823 0.79224 0.059137

## 20 0.0010000 34 0.70598 0.79065 0.058687

rsq.val\_basic<-1-tmp\_basic[,c(3,4)]

rsq.val\_basic<-rsq.val\_basic[nrow(rsq.val\_basic)]

rsq.val\_basic

## [1] 0.2940199

Lets build the model using basic and weekday features

blogtrain\_basicweekday<-blogtrain[,c(51:60,270:276,281)]

blogtest\_basicweekday<-blogtest[,c(51:60,270:276,281)]

Creating the CART model using rpart

tree\_basicweekday<-rpart(blogtrain\_basicweekday$X1.0.2~.,data=blogtrain\_basicweekday, method = "anova",control=rpart.control(cp=0.001))

printcp(tree\_basicweekday)

##

## Regression tree:

## rpart(formula = blogtrain\_basicweekday$X1.0.2 ~ ., data = blogtrain\_basicweekday,

## method = "anova", control = rpart.control(cp = 0.001))

##

## Variables actually used in tree construction:

## [1] X0.0.14 X0.0.15 X0.0.16 X0.0.18 X0.0.19 X2.0.1 X2.0.2 X2.0.3 X2.0.4

##

## Root node error: 74495814/52396 = 1421.8

##

## n= 52396

##

## CP nsplit rel error xerror xstd

## 1 0.1601367 0 1.00000 1.00002 0.066871

## 2 0.0356080 1 0.83986 0.84554 0.060959

## 3 0.0208948 2 0.80426 0.80694 0.059507

## 4 0.0089661 3 0.78336 0.79174 0.058932

## 5 0.0048979 4 0.77439 0.78235 0.059284

## 6 0.0045895 5 0.76950 0.78189 0.059400

## 7 0.0043841 6 0.76491 0.77941 0.059384

## 8 0.0035830 7 0.76052 0.78171 0.059427

## 9 0.0024613 11 0.74619 0.78005 0.059351

## 10 0.0024404 13 0.74127 0.78202 0.059316

## 11 0.0023383 15 0.73639 0.78478 0.059289

## 12 0.0021951 16 0.73405 0.78637 0.059298

## 13 0.0020115 17 0.73185 0.78618 0.059439

## 14 0.0017869 19 0.72783 0.78323 0.059312

## 15 0.0017494 24 0.71852 0.77994 0.059246

## 16 0.0012456 25 0.71677 0.78700 0.059362

## 17 0.0012077 29 0.71179 0.79016 0.059285

## 18 0.0011379 31 0.70937 0.79236 0.059343

## 19 0.0011267 32 0.70823 0.79430 0.059334

## 20 0.0010000 34 0.70598 0.79835 0.059345

Calculating the best complexity parameter of the model

bestcp\_basicweekday <-tree\_basicweekday$cptable[which.min(tree\_basicweekday$cptable[,"xerror"]),"CP"]

bestcp\_basicweekday

## [1] 0.004384053

Pruning the tree to avoid overfitting

tree.pruned\_basicweekday <- prune(tree\_basicweekday, cp=bestcp\_basicweekday)

summary(tree.pruned\_basicweekday)

## Call:

## rpart(formula = blogtrain\_basicweekday$X1.0.2 ~ ., data = blogtrain\_basicweekday,

## method = "anova", control = rpart.control(cp = 0.001))

## n= 52396

##

## CP nsplit rel error xerror xstd

## 1 0.160136685 0 1.0000000 1.0000150 0.06687054

## 2 0.035607988 1 0.8398633 0.8455374 0.06095860

## 3 0.020894750 2 0.8042553 0.8069393 0.05950731

## 4 0.008966101 3 0.7833606 0.7917356 0.05893210

## 5 0.004897880 4 0.7743945 0.7823488 0.05928438

## 6 0.004589499 5 0.7694966 0.7818918 0.05939956

## 7 0.004384053 6 0.7649071 0.7794079 0.05938418

##

## Variable importance

## X2.0.2 X2.0.4 X2.0.1 X2.0.3 X0.0.16 X0.0.19 X0.0.18 X0.0.15

## 44 38 4 4 3 3 2 2

##

## Node number 1: 52396 observations, complexity param=0.1601367

## mean=6.764829, MSE=1421.784

## left son=2 (51595 obs) right son=3 (801 obs)

## Primary splits:

## X2.0.2 < 221.5 to the left, improve=0.16013670, (0 missing)

## X2.0.4 < 185.5 to the left, improve=0.12456960, (0 missing)

## X2.0.3 < 279.5 to the left, improve=0.07712658, (0 missing)

## X2.0.1 < 352.5 to the left, improve=0.07535665, (0 missing)

## X0.0.16 < 2.5 to the left, improve=0.04953726, (0 missing)

## Surrogate splits:

## X2.0.4 < 221.5 to the left, agree=0.996, adj=0.760, (0 split)

## X2.0.1 < 818.5 to the left, agree=0.986, adj=0.055, (0 split)

## X2.0.3 < 788 to the left, agree=0.985, adj=0.029, (0 split)

## X0.0.16 < 9.5 to the left, agree=0.985, adj=0.019, (0 split)

## X0.0.19 < 10.5 to the left, agree=0.985, adj=0.017, (0 split)

##

## Node number 2: 51595 observations, complexity param=0.03560799

## mean=4.884756, MSE=843.6107

## left son=4 (45388 obs) right son=5 (6207 obs)

## Primary splits:

## X2.0.4 < 7.5 to the left, improve=0.06094381, (0 missing)

## X2.0.2 < 20.5 to the left, improve=0.05892590, (0 missing)

## X2.0.3 < 14.5 to the left, improve=0.02121635, (0 missing)

## X2.0.1 < 14.5 to the left, improve=0.02009537, (0 missing)

## X0.0.19 < 0.5 to the left, improve=0.01385694, (0 missing)

## Surrogate splits:

## X2.0.2 < 14.5 to the left, agree=0.938, adj=0.483, (0 split)

## X0.0.19 < 1.5 to the left, agree=0.891, adj=0.092, (0 split)

## X0.0.16 < 1.5 to the left, agree=0.888, adj=0.068, (0 split)

##

## Node number 3: 801 observations, complexity param=0.02089475

## mean=127.8664, MSE=23770.55

## left son=6 (575 obs) right son=7 (226 obs)

## Primary splits:

## X2.0.2 < 427.5 to the left, improve=0.08175181, (0 missing)

## X2.0.4 < 421 to the left, improve=0.07196278, (0 missing)

## X2.0.3 < 430 to the left, improve=0.06611954, (0 missing)

## X2.0.1 < 430.5 to the left, improve=0.05480402, (0 missing)

## X0.0.18 < 1.5 to the left, improve=0.04451982, (0 missing)

## Surrogate splits:

## X2.0.4 < 427.5 to the left, agree=0.939, adj=0.783, (0 split)

## X2.0.3 < 427.5 to the left, agree=0.843, adj=0.442, (0 split)

## X2.0.1 < 427.5 to the left, agree=0.813, adj=0.336, (0 split)

## X0.0.16 < 7.5 to the left, agree=0.757, adj=0.137, (0 split)

## X0.0.19 < 7.5 to the left, agree=0.757, adj=0.137, (0 split)

##

## Node number 4: 45388 observations, complexity param=0.00489788

## mean=2.233167, MSE=336.933

## left son=8 (45220 obs) right son=9 (168 obs)

## Primary splits:

## X2.0.2 < 107.5 to the left, improve=0.023859180, (0 missing)

## X2.0.1 < 400.5 to the left, improve=0.011253080, (0 missing)

## X0.0.14 < 187.5 to the left, improve=0.009131713, (0 missing)

## X2.0.4 < 1.5 to the left, improve=0.009126476, (0 missing)

## X2.0.3 < 343.5 to the left, improve=0.007968743, (0 missing)

## Surrogate splits:

## X0.0.14 < 655.5 to the left, agree=0.996, adj=0.018, (0 split)

##

## Node number 5: 6207 observations, complexity param=0.004589499

## mean=24.27421, MSE=4121.272

## left son=10 (3562 obs) right son=11 (2645 obs)

## Primary splits:

## X2.0.4 < 42.5 to the left, improve=0.013365470, (0 missing)

## X2.0.2 < 103.5 to the left, improve=0.011946390, (0 missing)

## X2.0.3 < 42.5 to the left, improve=0.011160650, (0 missing)

## X2.0.1 < 42.5 to the left, improve=0.009678477, (0 missing)

## X0.0.14 < 0.5 to the right, improve=0.007272552, (0 missing)

## Surrogate splits:

## X2.0.2 < 42.5 to the left, agree=0.973, adj=0.936, (0 split)

## X2.0.3 < 42.5 to the left, agree=0.955, adj=0.895, (0 split)

## X2.0.1 < 42.5 to the left, agree=0.951, adj=0.885, (0 split)

## X0.0.19 < 2.5 to the left, agree=0.616, adj=0.099, (0 split)

## X0.0.16 < 2.5 to the left, agree=0.614, adj=0.095, (0 split)

##

## Node number 6: 575 observations

## mean=100.2296, MSE=16396.05

##

## Node number 7: 226 observations, complexity param=0.008966101

## mean=198.1814, MSE=35645.6

## left son=14 (61 obs) right son=15 (165 obs)

## Primary splits:

## X0.0.18 < 2.5 to the left, improve=0.08291271, (0 missing)

## X0.0.15 < 2.5 to the left, improve=0.07893757, (0 missing)

## X0.0.16 < 2.5 to the left, improve=0.05969577, (0 missing)

## X2.0.2 < 666.5 to the left, improve=0.04538495, (0 missing)

## X0.0.19 < 2.5 to the left, improve=0.04426379, (0 missing)

## Surrogate splits:

## X0.0.15 < 2.5 to the left, agree=0.996, adj=0.984, (0 split)

## X0.0.16 < 2.5 to the left, agree=0.889, adj=0.590, (0 split)

## X0.0.19 < 2.5 to the left, agree=0.850, adj=0.443, (0 split)

## X2.0.3 < 433.5 to the left, agree=0.743, adj=0.049, (0 split)

## X2.0.1 < 433.5 to the left, agree=0.739, adj=0.033, (0 split)

##

## Node number 8: 45220 observations

## mean=2.060349, MSE=311.8869

##

## Node number 9: 168 observations

## mean=48.75, MSE=4906.652

##

## Node number 10: 3562 observations

## mean=17.87872, MSE=3641.125

##

## Node number 11: 2645 observations

## mean=32.88696, MSE=4638.619

##

## Node number 14: 61 observations

## mean=108.7705, MSE=39642.87

##

## Node number 15: 165 observations

## mean=231.2364, MSE=30119.72

Predict the model using test data

blogtrain\_basicweekday<-blogtrain\_basicweekday[-c(7565:52396),]

pred\_basicweekday<-predict(tree.pruned\_basicweekday,blogtest\_basicweekday,type = "vector")

*#print(tree.pruned\_basicweekday)*

Developing the confusion matrix

conf.matrix\_basicweekday<-table(blogtrain\_basicweekday$X1.0.2,pred\_basicweekday)

rownames(conf.matrix\_basicweekday) <- paste("Actual", rownames(conf.matrix\_basicweekday), sep=":")

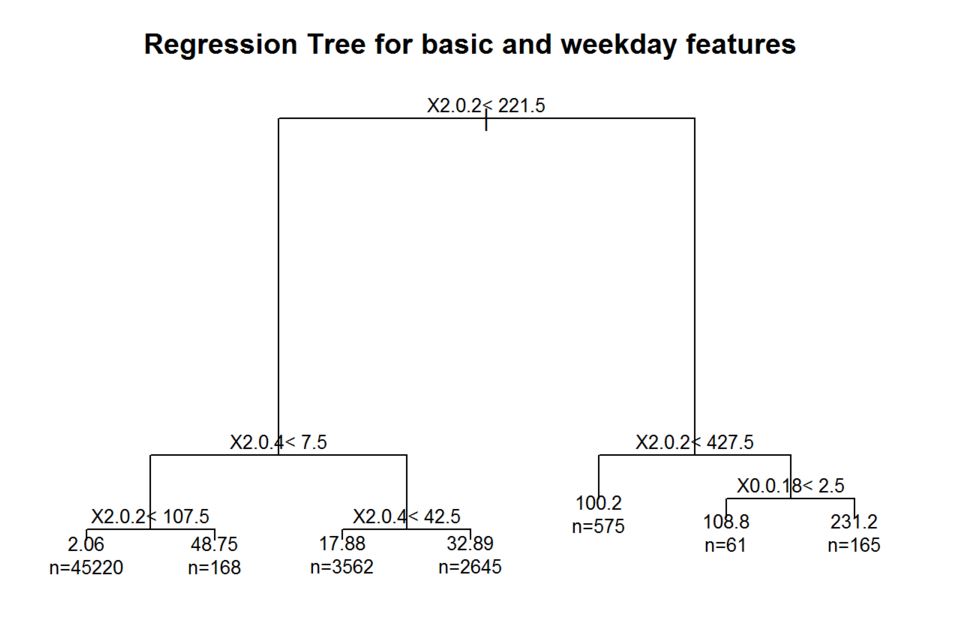
colnames(conf.matrix\_basicweekday) <- paste("Predicted", colnames(conf.matrix\_basicweekday), sep=":")

*#print(conf.matrix\_basic)*

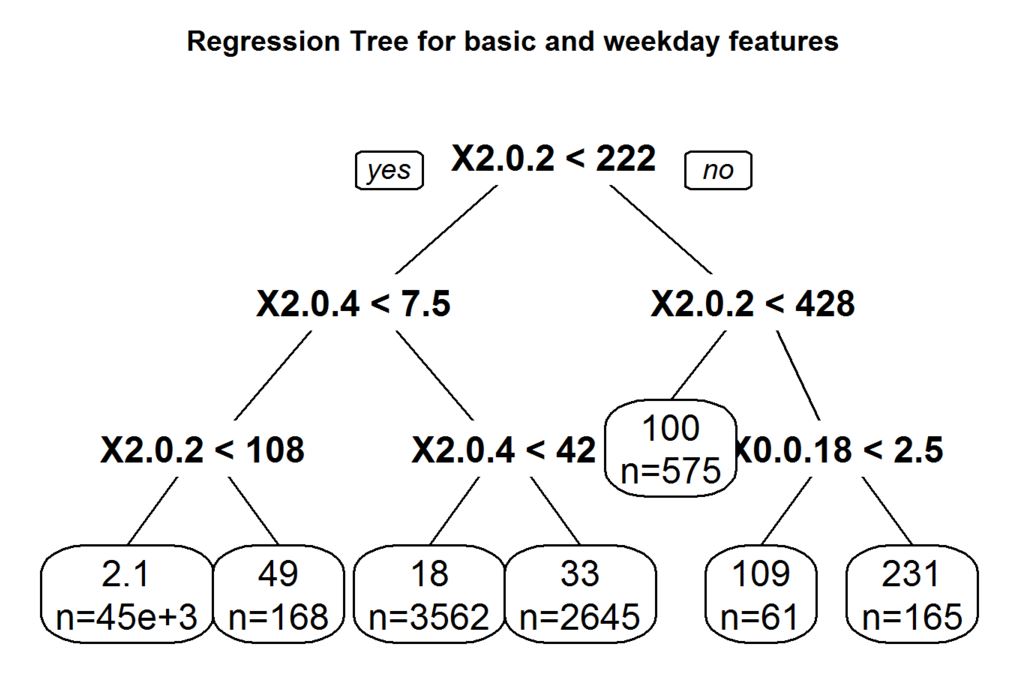
Plotting the regression tree for basic features

plot(tree.pruned\_basicweekday, main="Regression Tree for basic and weekday features")

text(tree.pruned\_basicweekday,cex=0.8,use.n=TRUE,xpd=TRUE)



prp(tree.pruned\_basicweekday,faclen=0,tweak=1.5,extra=1,main="Regression Tree for basic and weekday features")



Finding out the R squared value

tmp\_basicweekday<-printcp(tree\_basicweekday)

##

## Regression tree:

## rpart(formula = blogtrain\_basicweekday$X1.0.2 ~ ., data = blogtrain\_basicweekday,

## method = "anova", control = rpart.control(cp = 0.001))

##

## Variables actually used in tree construction:

## [1] X0.0.14 X0.0.15 X0.0.16 X0.0.18 X0.0.19 X2.0.1 X2.0.2 X2.0.3 X2.0.4

##

## Root node error: 74495814/52396 = 1421.8

##

## n= 52396

##

## CP nsplit rel error xerror xstd

## 1 0.1601367 0 1.00000 1.00002 0.066871

## 2 0.0356080 1 0.83986 0.84554 0.060959

## 3 0.0208948 2 0.80426 0.80694 0.059507

## 4 0.0089661 3 0.78336 0.79174 0.058932

## 5 0.0048979 4 0.77439 0.78235 0.059284

## 6 0.0045895 5 0.76950 0.78189 0.059400

## 7 0.0043841 6 0.76491 0.77941 0.059384

## 8 0.0035830 7 0.76052 0.78171 0.059427

## 9 0.0024613 11 0.74619 0.78005 0.059351

## 10 0.0024404 13 0.74127 0.78202 0.059316

## 11 0.0023383 15 0.73639 0.78478 0.059289

## 12 0.0021951 16 0.73405 0.78637 0.059298

## 13 0.0020115 17 0.73185 0.78618 0.059439

## 14 0.0017869 19 0.72783 0.78323 0.059312

## 15 0.0017494 24 0.71852 0.77994 0.059246

## 16 0.0012456 25 0.71677 0.78700 0.059362

## 17 0.0012077 29 0.71179 0.79016 0.059285

## 18 0.0011379 31 0.70937 0.79236 0.059343

## 19 0.0011267 32 0.70823 0.79430 0.059334

## 20 0.0010000 34 0.70598 0.79835 0.059345

rsq.val\_basicweekday<-1-tmp\_basicweekday[,c(3,4)]

rsq.val\_basicweekday<-rsq.val\_basicweekday[nrow(rsq.val\_basicweekday)]

rsq.val\_basicweekday

## [1] 0.2940199

Lets build the model using basic and textual features

blogtrain\_basictextual<-blogtrain[,c(51:60,281)]

blogtest\_basictextual<-blogtest[,c(51:60,63:262,281)]

Creating the CART model using rpart

tree\_basictextual<-rpart(blogtrain\_basictextual$X1.0.2~.,data=blogtrain\_basictextual, method = "anova",control=rpart.control(cp=0.001))

printcp(tree\_basictextual)

##

## Regression tree:

## rpart(formula = blogtrain\_basictextual$X1.0.2 ~ ., data = blogtrain\_basictextual,

## method = "anova", control = rpart.control(cp = 0.001))

##

## Variables actually used in tree construction:

## [1] X0.0.14 X0.0.15 X0.0.16 X0.0.18 X0.0.19 X2.0.1 X2.0.2 X2.0.3 X2.0.4

##

## Root node error: 74495814/52396 = 1421.8

##

## n= 52396

##

## CP nsplit rel error xerror xstd

## 1 0.1601367 0 1.00000 1.00001 0.066871

## 2 0.0356080 1 0.83986 0.84496 0.061011

## 3 0.0208948 2 0.80426 0.81139 0.059991

## 4 0.0089661 3 0.78336 0.79204 0.059210

## 5 0.0048979 4 0.77439 0.78251 0.059505

## 6 0.0045895 5 0.76950 0.78218 0.059299

## 7 0.0043841 6 0.76491 0.78033 0.059353

## 8 0.0035830 7 0.76052 0.78100 0.059393

## 9 0.0024613 11 0.74619 0.77988 0.059330

## 10 0.0024404 13 0.74127 0.78017 0.059194

## 11 0.0023383 15 0.73639 0.77882 0.059169

## 12 0.0021951 16 0.73405 0.78078 0.059356

## 13 0.0020115 17 0.73185 0.78196 0.059368

## 14 0.0017869 19 0.72783 0.78427 0.059370

## 15 0.0017494 24 0.71852 0.78563 0.059399

## 16 0.0012456 25 0.71677 0.78849 0.059026

## 17 0.0012077 29 0.71179 0.79699 0.059276

## 18 0.0011379 31 0.70937 0.79688 0.059264

## 19 0.0011267 32 0.70823 0.79721 0.059263

## 20 0.0010000 34 0.70598 0.80232 0.059384

Calculating the best complexity parameter of the model

bestcp\_basictextual <-tree\_basictextual$cptable[which.min(tree\_basictextual$cptable[,"xerror"]),"CP"]

bestcp\_basictextual

## [1] 0.002338311

Pruning the tree to avoid overfitting

tree.pruned\_basictextual <- prune(tree\_basictextual, cp=bestcp\_basictextual)

summary(tree.pruned\_basictextual)

## Call:

## rpart(formula = blogtrain\_basictextual$X1.0.2 ~ ., data = blogtrain\_basictextual,

## method = "anova", control = rpart.control(cp = 0.001))

## n= 52396

##

## CP nsplit rel error xerror xstd

## 1 0.160136685 0 1.0000000 1.0000137 0.06687109

## 2 0.035607988 1 0.8398633 0.8449633 0.06101065

## 3 0.020894750 2 0.8042553 0.8113902 0.05999129

## 4 0.008966101 3 0.7833606 0.7920366 0.05921012

## 5 0.004897880 4 0.7743945 0.7825115 0.05950489

## 6 0.004589499 5 0.7694966 0.7821795 0.05929944

## 7 0.004384053 6 0.7649071 0.7803322 0.05935308

## 8 0.003582984 7 0.7605230 0.7810036 0.05939342

## 9 0.002461266 11 0.7461911 0.7798818 0.05932994

## 10 0.002440411 13 0.7412686 0.7801736 0.05919357

## 11 0.002338311 15 0.7363878 0.7788202 0.05916928

##

## Variable importance

## X2.0.2 X2.0.4 X2.0.1 X2.0.3 X0.0.16 X0.0.19 X0.0.18 X0.0.15 X0.0.14

## 40 36 7 6 3 2 2 2 1

##

## Node number 1: 52396 observations, complexity param=0.1601367

## mean=6.764829, MSE=1421.784

## left son=2 (51595 obs) right son=3 (801 obs)

## Primary splits:

## X2.0.2 < 221.5 to the left, improve=0.16013670, (0 missing)

## X2.0.4 < 185.5 to the left, improve=0.12456960, (0 missing)

## X2.0.3 < 279.5 to the left, improve=0.07712658, (0 missing)

## X2.0.1 < 352.5 to the left, improve=0.07535665, (0 missing)

## X0.0.16 < 2.5 to the left, improve=0.04953726, (0 missing)

## Surrogate splits:

## X2.0.4 < 221.5 to the left, agree=0.996, adj=0.760, (0 split)

## X2.0.1 < 818.5 to the left, agree=0.986, adj=0.055, (0 split)

## X2.0.3 < 788 to the left, agree=0.985, adj=0.029, (0 split)

## X0.0.16 < 9.5 to the left, agree=0.985, adj=0.019, (0 split)

## X0.0.19 < 10.5 to the left, agree=0.985, adj=0.017, (0 split)

##

## Node number 2: 51595 observations, complexity param=0.03560799

## mean=4.884756, MSE=843.6107

## left son=4 (45388 obs) right son=5 (6207 obs)

## Primary splits:

## X2.0.4 < 7.5 to the left, improve=0.06094381, (0 missing)

## X2.0.2 < 20.5 to the left, improve=0.05892590, (0 missing)

## X2.0.3 < 14.5 to the left, improve=0.02121635, (0 missing)

## X2.0.1 < 14.5 to the left, improve=0.02009537, (0 missing)

## X0.0.19 < 0.5 to the left, improve=0.01385694, (0 missing)

## Surrogate splits:

## X2.0.2 < 14.5 to the left, agree=0.938, adj=0.483, (0 split)

## X0.0.19 < 1.5 to the left, agree=0.891, adj=0.092, (0 split)

## X0.0.16 < 1.5 to the left, agree=0.888, adj=0.068, (0 split)

##

## Node number 3: 801 observations, complexity param=0.02089475

## mean=127.8664, MSE=23770.55

## left son=6 (575 obs) right son=7 (226 obs)

## Primary splits:

## X2.0.2 < 427.5 to the left, improve=0.08175181, (0 missing)

## X2.0.4 < 421 to the left, improve=0.07196278, (0 missing)

## X2.0.3 < 430 to the left, improve=0.06611954, (0 missing)

## X2.0.1 < 430.5 to the left, improve=0.05480402, (0 missing)

## X0.0.18 < 1.5 to the left, improve=0.04451982, (0 missing)

## Surrogate splits:

## X2.0.4 < 427.5 to the left, agree=0.939, adj=0.783, (0 split)

## X2.0.3 < 427.5 to the left, agree=0.843, adj=0.442, (0 split)

## X2.0.1 < 427.5 to the left, agree=0.813, adj=0.336, (0 split)

## X0.0.16 < 7.5 to the left, agree=0.757, adj=0.137, (0 split)

## X0.0.19 < 7.5 to the left, agree=0.757, adj=0.137, (0 split)

##

## Node number 4: 45388 observations, complexity param=0.00489788

## mean=2.233167, MSE=336.933

## left son=8 (45220 obs) right son=9 (168 obs)

## Primary splits:

## X2.0.2 < 107.5 to the left, improve=0.023859180, (0 missing)

## X2.0.1 < 400.5 to the left, improve=0.011253080, (0 missing)

## X0.0.14 < 187.5 to the left, improve=0.009131713, (0 missing)

## X2.0.4 < 1.5 to the left, improve=0.009126476, (0 missing)

## X2.0.3 < 343.5 to the left, improve=0.007968743, (0 missing)

## Surrogate splits:

## X0.0.14 < 655.5 to the left, agree=0.996, adj=0.018, (0 split)

##

## Node number 5: 6207 observations, complexity param=0.004589499

## mean=24.27421, MSE=4121.272

## left son=10 (3562 obs) right son=11 (2645 obs)

## Primary splits:

## X2.0.4 < 42.5 to the left, improve=0.013365470, (0 missing)

## X2.0.2 < 103.5 to the left, improve=0.011946390, (0 missing)

## X2.0.3 < 42.5 to the left, improve=0.011160650, (0 missing)

## X2.0.1 < 42.5 to the left, improve=0.009678477, (0 missing)

## X0.0.14 < 0.5 to the right, improve=0.007272552, (0 missing)

## Surrogate splits:

## X2.0.2 < 42.5 to the left, agree=0.973, adj=0.936, (0 split)

## X2.0.3 < 42.5 to the left, agree=0.955, adj=0.895, (0 split)

## X2.0.1 < 42.5 to the left, agree=0.951, adj=0.885, (0 split)

## X0.0.19 < 2.5 to the left, agree=0.616, adj=0.099, (0 split)

## X0.0.16 < 2.5 to the left, agree=0.614, adj=0.095, (0 split)

##

## Node number 6: 575 observations, complexity param=0.003582984

## mean=100.2296, MSE=16396.05

## left son=12 (184 obs) right son=13 (391 obs)

## Primary splits:

## X2.0.3 < 276.5 to the left, improve=0.02100116, (0 missing)

## X0.0.18 < 0.5 to the left, improve=0.01977664, (0 missing)

## X0.0.15 < 0.5 to the left, improve=0.01941526, (0 missing)

## X2.0.1 < 353.5 to the left, improve=0.01664092, (0 missing)

## X2.0.2 < 354.5 to the left, improve=0.01289905, (0 missing)

## Surrogate splits:

## X2.0.1 < 276.5 to the left, agree=0.951, adj=0.848, (0 split)

## X2.0.2 < 276.5 to the left, agree=0.880, adj=0.625, (0 split)

## X2.0.4 < 276.5 to the left, agree=0.718, adj=0.120, (0 split)

##

## Node number 7: 226 observations, complexity param=0.008966101

## mean=198.1814, MSE=35645.6

## left son=14 (61 obs) right son=15 (165 obs)

## Primary splits:

## X0.0.18 < 2.5 to the left, improve=0.08291271, (0 missing)

## X0.0.15 < 2.5 to the left, improve=0.07893757, (0 missing)

## X0.0.16 < 2.5 to the left, improve=0.05969577, (0 missing)

## X2.0.2 < 666.5 to the left, improve=0.04538495, (0 missing)

## X0.0.19 < 2.5 to the left, improve=0.04426379, (0 missing)

## Surrogate splits:

## X0.0.15 < 2.5 to the left, agree=0.996, adj=0.984, (0 split)

## X0.0.16 < 2.5 to the left, agree=0.889, adj=0.590, (0 split)

## X0.0.19 < 2.5 to the left, agree=0.850, adj=0.443, (0 split)

## X2.0.3 < 433.5 to the left, agree=0.743, adj=0.049, (0 split)

## X2.0.1 < 433.5 to the left, agree=0.739, adj=0.033, (0 split)

##

## Node number 8: 45220 observations

## mean=2.060349, MSE=311.8869

##

## Node number 9: 168 observations

## mean=48.75, MSE=4906.652

##

## Node number 10: 3562 observations

## mean=17.87872, MSE=3641.125

##

## Node number 11: 2645 observations

## mean=32.88696, MSE=4638.619

##

## Node number 12: 184 observations

## mean=73.17935, MSE=13359.72

##

## Node number 13: 391 observations, complexity param=0.003582984

## mean=112.9591, MSE=17318.54

## left son=26 (106 obs) right son=27 (285 obs)

## Primary splits:

## X0.0.18 < 0.5 to the left, improve=0.04192850, (0 missing)

## X0.0.15 < 0.5 to the left, improve=0.04103568, (0 missing)

## X0.0.19 < 0.5 to the left, improve=0.02890781, (0 missing)

## X0.0.16 < 0.5 to the left, improve=0.02322103, (0 missing)

## X2.0.1 < 288.5 to the right, improve=0.02304655, (0 missing)

## Surrogate splits:

## X0.0.15 < 0.5 to the left, agree=0.997, adj=0.991, (0 split)

## X0.0.16 < 0.5 to the left, agree=0.847, adj=0.434, (0 split)

##

## Node number 14: 61 observations, complexity param=0.004384053

## mean=108.7705, MSE=39642.87

## left son=28 (54 obs) right son=29 (7 obs)

## Primary splits:

## X2.0.4 < 702.5 to the left, improve=0.13505570, (0 missing)

## X2.0.2 < 694 to the left, improve=0.07957718, (0 missing)

## X2.0.3 < 722 to the left, improve=0.06185540, (0 missing)

## X0.0.14 < 5.5 to the right, improve=0.05540664, (0 missing)

## X2.0.1 < 764.5 to the right, improve=0.03749789, (0 missing)

## Surrogate splits:

## X2.0.2 < 694 to the left, agree=0.967, adj=0.714, (0 split)

## X2.0.3 < 722 to the left, agree=0.951, adj=0.571, (0 split)

## X2.0.1 < 934 to the left, agree=0.902, adj=0.143, (0 split)

##

## Node number 15: 165 observations, complexity param=0.002461266

## mean=231.2364, MSE=30119.72

## left son=30 (11 obs) right son=31 (154 obs)

## Primary splits:

## X2.0.4 < -16 to the left, improve=0.029056610, (0 missing)

## X2.0.3 < 445.5 to the left, improve=0.025428800, (0 missing)

## X2.0.2 < 666.5 to the left, improve=0.022494830, (0 missing)

## X2.0.1 < 445.5 to the left, improve=0.019468590, (0 missing)

## X0.0.14 < 558.5 to the right, improve=0.009872295, (0 missing)

## Surrogate splits:

## X0.0.14 < 558.5 to the right, agree=0.976, adj=0.636, (0 split)

## X0.0.16 < 0.5 to the left, agree=0.970, adj=0.545, (0 split)

## X2.0.1 < 1553.5 to the right, agree=0.952, adj=0.273, (0 split)

## X0.0.19 < -2.5 to the left, agree=0.939, adj=0.091, (0 split)

##

## Node number 26: 106 observations

## mean=68.77358, MSE=8636.892

##

## Node number 27: 285 observations, complexity param=0.003582984

## mean=129.393, MSE=19551.28

## left son=54 (263 obs) right son=55 (22 obs)

## Primary splits:

## X2.0.1 < 291.5 to the right, improve=0.04152467, (0 missing)

## X2.0.3 < 290 to the right, improve=0.03675561, (0 missing)

## X0.0.14 < 0.5 to the right, improve=0.02339020, (0 missing)

## X2.0.2 < 288.5 to the right, improve=0.01363084, (0 missing)

## X0.0.17 < 1.5 to the right, improve=0.01008088, (0 missing)

## Surrogate splits:

## X2.0.3 < 288.5 to the right, agree=0.986, adj=0.818, (0 split)

##

## Node number 28: 54 observations

## mean=82.42593, MSE=22460.02

##

## Node number 29: 7 observations

## mean=312, MSE=125540

##

## Node number 30: 11 observations

## mean=120.5455, MSE=22304.43

##

## Node number 31: 154 observations, complexity param=0.002461266

## mean=239.1429, MSE=29740.27

## left son=62 (137 obs) right son=63 (17 obs)

## Primary splits:

## X2.0.1 < 1021.5 to the left, improve=0.04853794, (0 missing)

## X0.0.14 < 463.5 to the left, improve=0.03582062, (0 missing)

## X2.0.3 < 445.5 to the left, improve=0.03236608, (0 missing)

## X2.0.4 < 219.5 to the right, improve=0.02494573, (0 missing)

## X0.0.17 < 1.5 to the left, improve=0.02038482, (0 missing)

## Surrogate splits:

## X0.0.14 < 426.5 to the left, agree=0.942, adj=0.471, (0 split)

## X2.0.3 < 1038 to the left, agree=0.942, adj=0.471, (0 split)

## X2.0.2 < 955.5 to the left, agree=0.929, adj=0.353, (0 split)

## X2.0.4 < 1028 to the left, agree=0.929, adj=0.353, (0 split)

## X0.0.17 < 4.5 to the left, agree=0.916, adj=0.235, (0 split)

##

## Node number 54: 263 observations

## mean=121.1521, MSE=16487.41

##

## Node number 55: 22 observations, complexity param=0.003582984

## mean=227.9091, MSE=45661.17

## left son=110 (13 obs) right son=111 (9 obs)

## Primary splits:

## X2.0.4 < 282.5 to the left, improve=0.35277120, (0 missing)

## X2.0.2 < 284 to the left, improve=0.25578780, (0 missing)

## X2.0.3 < 284.5 to the left, improve=0.25578780, (0 missing)

## X2.0.1 < 282.5 to the left, improve=0.15519920, (0 missing)

## X0.0.18 < 2.5 to the right, improve=0.06292435, (0 missing)

## Surrogate splits:

## X2.0.2 < 282.5 to the left, agree=0.955, adj=0.889, (0 split)

## X2.0.3 < 284.5 to the left, agree=0.955, adj=0.889, (0 split)

## X2.0.1 < 282.5 to the left, agree=0.909, adj=0.778, (0 split)

##

## Node number 62: 137 observations, complexity param=0.002440411

## mean=225.7591, MSE=28820.46

## left son=124 (27 obs) right son=125 (110 obs)

## Primary splits:

## X2.0.4 < 447 to the left, improve=0.03170966, (0 missing)

## X2.0.3 < 445.5 to the left, improve=0.02902210, (0 missing)

## X2.0.1 < 445.5 to the left, improve=0.02206072, (0 missing)

## X2.0.2 < 445.5 to the left, improve=0.02049825, (0 missing)

## X0.0.18 < 8.5 to the right, improve=0.01489800, (0 missing)

## Surrogate splits:

## X2.0.2 < 445.5 to the left, agree=0.898, adj=0.481, (0 split)

## X0.0.14 < 173.5 to the right, agree=0.883, adj=0.407, (0 split)

## X2.0.1 < 445.5 to the left, agree=0.876, adj=0.370, (0 split)

## X2.0.3 < 445.5 to the left, agree=0.876, adj=0.370, (0 split)

## X0.0.16 < 2.5 to the left, agree=0.847, adj=0.222, (0 split)

##

## Node number 63: 17 observations

## mean=347, MSE=24076.12

##

## Node number 110: 13 observations

## mean=122.3077, MSE=17903.14

##

## Node number 111: 9 observations

## mean=380.4444, MSE=46381.14

##

## Node number 124: 27 observations

## mean=164.7407, MSE=11699.53

##

## Node number 125: 110 observations, complexity param=0.002440411

## mean=240.7364, MSE=31884.67

## left son=250 (96 obs) right son=251 (14 obs)

## Primary splits:

## X2.0.1 < 482.5 to the right, improve=0.06797177, (0 missing)

## X2.0.2 < 481 to the right, improve=0.06797177, (0 missing)

## X2.0.4 < 466.5 to the right, improve=0.05472211, (0 missing)

## X2.0.3 < 482 to the right, improve=0.03704196, (0 missing)

## X0.0.16 < 8.5 to the right, improve=0.03339207, (0 missing)

## Surrogate splits:

## X2.0.2 < 481 to the right, agree=1.000, adj=1.000, (0 split)

## X2.0.3 < 480 to the right, agree=0.982, adj=0.857, (0 split)

## X2.0.4 < 470 to the right, agree=0.982, adj=0.857, (0 split)

##

## Node number 250: 96 observations

## mean=222.9583, MSE=27779.44

##

## Node number 251: 14 observations

## mean=362.6429, MSE=43006.37

Predict the model using test data

blogtrain\_basictextual<-blogtrain\_basictextual[-c(7565:52396),]

pred\_basictextual<-predict(tree.pruned\_basictextual,blogtest\_basictextual,type = "vector")

print(tree.pruned\_basictextual)

## n= 52396

##

## node), split, n, deviance, yval

## \* denotes terminal node

##

## 1) root 52396 74495810.0 6.764829

## 2) X2.0.2< 221.5 51595 43526090.0 4.884756

## 4) X2.0.4< 7.5 45388 15292720.0 2.233167

## 8) X2.0.2< 107.5 45220 14103530.0 2.060349 \*

## 9) X2.0.2>=107.5 168 824317.5 48.750000 \*

## 5) X2.0.4>=7.5 6207 25580730.0 24.274210

## 10) X2.0.4< 42.5 3562 12969690.0 17.878720 \*

## 11) X2.0.4>=42.5 2645 12269150.0 32.886960 \*

## 3) X2.0.2>=221.5 801 19040210.0 127.866400

## 6) X2.0.2< 427.5 575 9427730.0 100.229600

## 12) X2.0.3< 276.5 184 2458189.0 73.179350 \*

## 13) X2.0.3>=276.5 391 6771547.0 112.959100

## 26) X0.0.18< 0.5 106 915510.6 68.773580 \*

## 27) X0.0.18>=0.5 285 5572116.0 129.393000

## 54) X2.0.1>=291.5 263 4336190.0 121.152100 \*

## 55) X2.0.1< 291.5 22 1004546.0 227.909100

## 110) X2.0.4< 282.5 13 232740.8 122.307700 \*

## 111) X2.0.4>=282.5 9 417430.2 380.444400 \*

## 7) X2.0.2>=427.5 226 8055906.0 198.181400

## 14) X0.0.18< 2.5 61 2418215.0 108.770500

## 28) X2.0.4< 702.5 54 1212841.0 82.425930 \*

## 29) X2.0.4>=702.5 7 878780.0 312.000000 \*

## 15) X0.0.18>=2.5 165 4969754.0 231.236400

## 30) X2.0.4< -16 11 245348.7 120.545500 \*

## 31) X2.0.4>=-16 154 4580001.0 239.142900

## 62) X2.0.1< 1021.5 137 3948403.0 225.759100

## 124) X2.0.4< 447 27 315887.2 164.740700 \*

## 125) X2.0.4>=447 110 3507313.0 240.736400

## 250) X2.0.1>=482.5 96 2666826.0 222.958300 \*

## 251) X2.0.1< 482.5 14 602089.2 362.642900 \*

## 63) X2.0.1>=1021.5 17 409294.0 347.000000 \*

Developing the confusion matrix

conf.matrix\_basictextual<-table(blogtrain\_basictextual$X1.0.2,pred\_basictextual)

rownames(conf.matrix\_basictextual) <- paste("Actual", rownames(conf.matrix\_basictextual), sep=":")

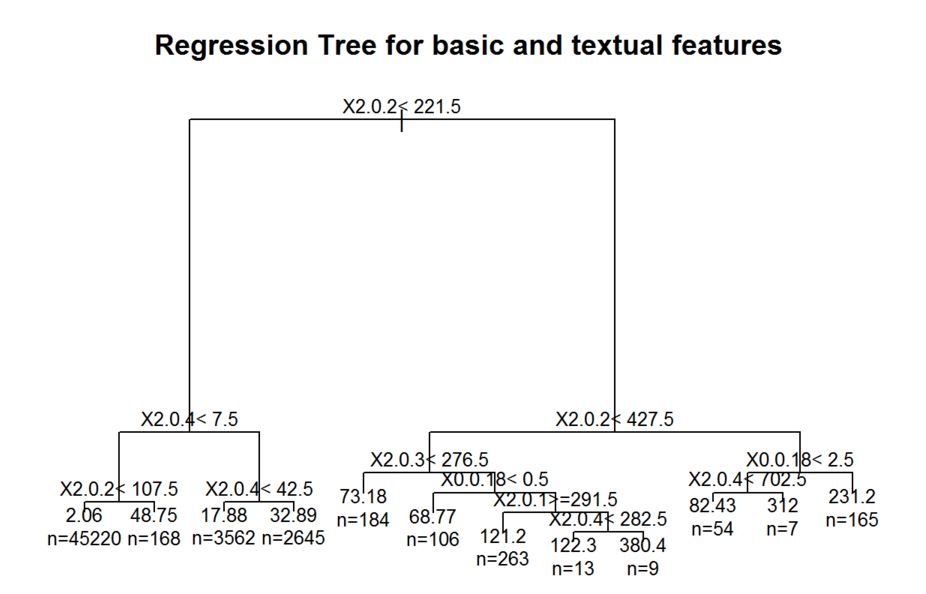
colnames(conf.matrix\_basictextual) <- paste("Predicted", colnames(conf.matrix\_basictextual), sep=":")

*##print(conf.matrix\_basic)*

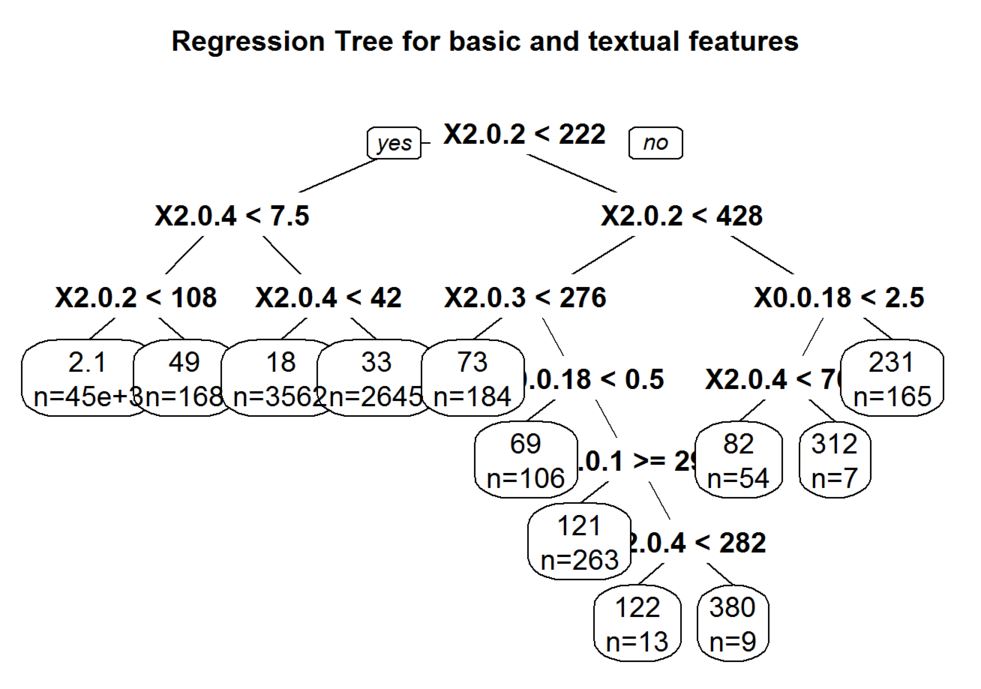
Plotting the regression tree for basic features

plot(tree.pruned\_basic, main="Regression Tree for basic and textual features")

text(tree.pruned\_basic,cex=0.8,use.n=TRUE,xpd=TRUE)



prp(tree.pruned\_basic,faclen=0,tweak=1.5,extra=1,main="Regression Tree for basic and textual features")



Finding out the R squared value

tmp\_basictextual<-printcp(tree\_basictextual)

##

## Regression tree:

## rpart(formula = blogtrain\_basictextual$X1.0.2 ~ ., data = blogtrain\_basictextual,

## method = "anova", control = rpart.control(cp = 0.001))

##

## Variables actually used in tree construction:

## [1] X0.0.14 X0.0.15 X0.0.16 X0.0.18 X0.0.19 X2.0.1 X2.0.2 X2.0.3 X2.0.4

##

## Root node error: 74495814/52396 = 1421.8

##

## n= 52396

##

## CP nsplit rel error xerror xstd

## 1 0.1601367 0 1.00000 1.00001 0.066871

## 2 0.0356080 1 0.83986 0.84496 0.061011

## 3 0.0208948 2 0.80426 0.81139 0.059991

## 4 0.0089661 3 0.78336 0.79204 0.059210

## 5 0.0048979 4 0.77439 0.78251 0.059505

## 6 0.0045895 5 0.76950 0.78218 0.059299

## 7 0.0043841 6 0.76491 0.78033 0.059353

## 8 0.0035830 7 0.76052 0.78100 0.059393

## 9 0.0024613 11 0.74619 0.77988 0.059330

## 10 0.0024404 13 0.74127 0.78017 0.059194

## 11 0.0023383 15 0.73639 0.77882 0.059169

## 12 0.0021951 16 0.73405 0.78078 0.059356

## 13 0.0020115 17 0.73185 0.78196 0.059368

## 14 0.0017869 19 0.72783 0.78427 0.059370

## 15 0.0017494 24 0.71852 0.78563 0.059399

## 16 0.0012456 25 0.71677 0.78849 0.059026

## 17 0.0012077 29 0.71179 0.79699 0.059276

## 18 0.0011379 31 0.70937 0.79688 0.059264

## 19 0.0011267 32 0.70823 0.79721 0.059263

## 20 0.0010000 34 0.70598 0.80232 0.059384

rsq.val\_basictextual<-1-tmp\_basictextual[,c(3,4)]

rsq.val\_basictextual<-rsq.val\_basictextual[nrow(rsq.val\_basictextual)]

rsq.val\_basictextual

## [1] 0.2940199

Lets build models using random forest algorithm

**library**(randomForest)

## Warning: package 'randomForest' was built under R version 3.2.4

## randomForest 4.6-12

## Type rfNews() to see new features/changes/bug fixes.

set.seed(400)

Creating the training and test data of basic features only

blogtrain\_basic<-blogtrain[,c(51:60,281)]

blogtest\_basic<-blogtest[,c(51:60,281)]

Using randomForest()

randomforest\_basic<-randomForest(blogtrain\_basic$X1.0.2~.,data=blogtrain\_basic,importance=TRUE,ntree=50)

print(randomforest\_basic)

##

## Call:

## randomForest(formula = blogtrain\_basic$X1.0.2 ~ ., data = blogtrain\_basic, importance = TRUE, ntree = 50)

## Type of random forest: regression

## Number of trees: 50

## No. of variables tried at each split: 3

##

## Mean of squared residuals: 1135.312

## % Var explained: 20.15

importance(randomforest\_basic)

## %IncMSE IncNodePurity

## X2.0.1 4.8664667 5267405.7

## X2.0.2 6.5607866 10202230.5

## X0.0.14 4.5601105 1675205.2

## X2.0.3 3.2092975 5638494.7

## X2.0.4 4.9623873 9853232.9

## X0.0.15 3.1881349 1198775.0

## X0.0.16 1.0897886 1663224.1

## X0.0.17 2.4482372 387600.7

## X0.0.18 3.6970186 1459824.3

## X0.0.19 0.2207321 1560206.5

Predicting using test data

predictions\_basic<-predict(randomforest\_basic,blogtest\_basic)

summary(predictions\_basic)

## Min. 1st Qu. Median Mean 3rd Qu. Max.

## 0.000 1.041 1.041 6.078 3.044 280.400

blogtrain\_basic<-blogtrain\_basic[-c(7565:52396),]

Creating the confirmation matrix

confmatrix\_basic<-table(blogtrain\_basic$X1.0.2,predictions\_basic)

rownames(confmatrix\_basic) <- paste("Actual", rownames(confmatrix\_basic), sep=":")

colnames(confmatrix\_basic) <- paste("Predicted", colnames(confmatrix\_basic), sep=":")

*#print(confmatrix\_basic)*

Creating the training and test data of basic and weekday features

blogtrain\_basicweekday<-blogtrain[,c(51:60,270:276,281)]

blogtest\_basicweekday<-blogtest[,c(51:60,270:276,281)]

Using randomForest()

randomforest\_basicweekday<-randomForest(blogtrain\_basicweekday$X1.0.2~.,data=blogtrain\_basicweekday,importance=TRUE,ntree=50)

print(randomforest\_basicweekday)

##

## Call:

## randomForest(formula = blogtrain\_basicweekday$X1.0.2 ~ ., data = blogtrain\_basicweekday, importance = TRUE, ntree = 50)

## Type of random forest: regression

## Number of trees: 50

## No. of variables tried at each split: 5

##

## Mean of squared residuals: 1153.45

## % Var explained: 18.87

importance(randomforest\_basicweekday)

## %IncMSE IncNodePurity

## X2.0.1 8.28360729 5474937.5

## X2.0.2 7.12132657 11258256.1

## X0.0.14 2.82422601 1509583.8

## X2.0.3 6.02552296 6119372.0

## X2.0.4 4.47786177 8423455.6

## X0.0.15 4.76204712 1197774.4

## X0.0.16 4.47502991 1544939.7

## X0.0.17 1.54761778 334115.4

## X0.0.18 3.15753880 1470783.3

## X0.0.19 1.81934863 1912855.4

## X0.0.227 0.90022649 738938.6

## X0.0.228 -0.03103034 648746.3

## X0.0.229 -0.75762494 969809.5

## X1.0.1 1.87161750 872882.6

## X0.0.230 -0.53249118 823952.4

## X0.0.231 -0.04719097 470063.9

## X0.0.232 -1.20131681 483520.1

Predicting using test data

predictions\_basicweekday<-predict(randomforest\_basicweekday,blogtest\_basicweekday)

summary(predictions\_basicweekday)

## Min. 1st Qu. Median Mean 3rd Qu. Max.

## 0.0000 0.8047 1.1030 6.1090 2.7240 316.6000

blogtrain\_basicweekday<-blogtrain\_basicweekday[-c(7565:52396),]

Creating the confusion matrix

confmatrix\_basicweekday<-table(blogtrain\_basicweekday$X1.0.2,predictions\_basicweekday)

rownames(confmatrix\_basicweekday) <- paste("Actual", rownames(confmatrix\_basicweekday), sep=":")

colnames(confmatrix\_basicweekday) <- paste("Predicted", colnames(confmatrix\_basicweekday), sep=":")

*#print(confmatrix\_basicweekday)*

Following is the table of comparison of R squared vales:

|  |  |  |
| --- | --- | --- |
|  | Regression | CART |
| Basic features | 0.2255 | 0.294 |
| Basic + Weekday Features | 0.2256 | 0.294 |
| Basic + Textual Features | 0.233 | 0.294 |

Comparison of R squared values obtained in Regression and CART. By looking at these values, we can infer that CART is the better model among them as it has R squared value closer to 1.