**Homework-4**

Group IX (Rajeev Motwani)

Ashvin Khairnar

Dimple Bapna

Soumyajeet Patra

Balaji Kothandaraman Kalanidhi

206-941-8514(Ashvin Khairnar)

425-233-7801 (Dimple Bapna)

206-218-3144(Soumyajeet Patra)

206-915-5863 (Balaji Kothandaraman kalanidhi)

**Percentage of Effort Contributed by Student 1:** 25

**Percentage of Effort Contributed by Student 2:** 25

**Percentage of Effort Contributed by Student 3:** 25

**Percentage of Effort Contributed by Student 4:** 25

**Signature of Student 1:** Ashvin Khairnar

**Signature of Student 2:** Dimple Bapna

**Signature of Student 3:**  Soumyajeet Patra

**Signature of Student 4:** Balaji Kothandaraman Kalanidhi

**Submission Date:** 04-18-2020

library(tidyverse)

## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --

## v ggplot2 3.3.0 v purrr 0.3.3  
## v tibble 2.1.3 v dplyr 0.8.5  
## v tidyr 1.0.2 v stringr 1.4.0  
## v readr 1.3.1 v forcats 0.5.0

## -- Conflicts ------------------------------------------ tidyverse\_conflicts() --  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()

library(ggplot2)  
library(gains)  
library(rpart)  
library(rpart.plot)  
library(caret)

## Loading required package: lattice

##   
## Attaching package: 'caret'

## The following object is masked from 'package:purrr':  
##   
## lift

library(glmulti)

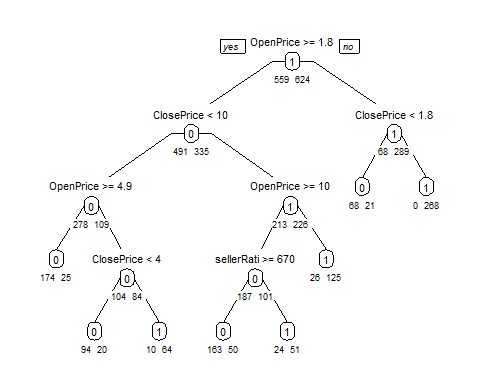
## Loading required package: rJava

eBayAuctions.df <- read.csv("C:\\Users\\kkbal\\OneDrive\\Desktop\\Neu\\data mining\\Assignment-4\\eBayAuctions.csv")  
# Converting "Duration" into categorical variable  
eBayAuctions.df$Duration <- as.factor(eBayAuctions.df$Duration)  
  
# Splitting the data into training(60%) and validation(40%) sets  
set.seed(100)  
train.index <- sample(c(1:dim(eBayAuctions.df)[1]),   
 dim(eBayAuctions.df)[1] \* 0.6)  
train.df <- eBayAuctions.df[train.index,]  
validation.df <- eBayAuctions.df[-train.index,]

Problem 9.1. A

#classification tree  
class.tree <- rpart(formula = Competitive. ~.,  
 data = train.df,  
 control = rpart.control(maxdepth = 7, minbucket = 30),  
 method = "class")

prp(class.tree, type =1, extra = 1, under = TRUE, split.font =1, varlen = -10)



# argument xval refers to the number of folds to use in rpart's built-in  
# cross-validation procedure  
# argument cp sets the smallest value for the complexity parameter.  
cv.ct <- rpart(Competitive. ~ .,   
 data = train.df,   
 control = rpart.control(maxdepth = 7, minbucket = 50),  
 method = "class",  
 cp = 0.00001, minsplit = 5, xval = 5)  
# use printcp() to print the table.  
printcp(cv.ct)

##   
## Classification tree:  
## rpart(formula = Competitive. ~ ., data = train.df, method = "class",   
## control = rpart.control(maxdepth = 7, minbucket = 50), cp = 1e-05,   
## minsplit = 5, xval = 5)  
##   
## Variables actually used in tree construction:  
## [1] ClosePrice OpenPrice sellerRating  
##   
## Root node error: 559/1183 = 0.47253  
##   
## n= 1183   
##   
## CP nsplit rel error xerror xstd  
## 1 0.279070 0 1.00000 1.00000 0.030718  
## 2 0.088551 1 0.72093 0.74061 0.029347  
## 3 0.084079 3 0.54383 0.62075 0.028013  
## 4 0.048301 4 0.45975 0.52594 0.026590  
## 5 0.010000 7 0.31485 0.42039 0.024549

Since the xerror is continuously declining, we do not need to prune the tree.  It is already a pruned tree.

RULES:

IF (OpenPrice < 1.8) AND (ClosePrice >= 1.8)  
THEN CLASS = 1  
IF (OpenPrice < 1.8) AND (ClosePrice < 1.8)  
THEN CLASS = 0  
IF (ClosePrice < 10) AND (OpenPrice >= 4.9)  
THEN CLASS = 0  
IF (OpenPrice >= 1.8) AND (ClosePrice < 10) AND (OpenPrice < 4.9)  
THEN CLASS = 0  
IF (OpenPrice >= 1.8) AND (ClosePrice < 10) AND (OpenPrice < 4.9) AND (ClosePrice >= 4)  
THEN CLASS = 1  
IF (ClosePrice >= 10) AND (OpenPrice >= 10) AND (sellerRating >= 670)  
THEN CLASS = 0  
IF (ClosePrice >= 10) AND (OpenPrice >= 10) AND (sellerRating < 670)  
THEN CLASS = 1  
IF (OpenPrice >= 1.8) AND (ClosePrice >= 10) AND (OpenPrice < 10)  
THEN CLASS = 1

Since the predictors OpenPrice, ClosePrice and SellerRating are significant, as we can see in the decision tree, we can remove Currency, and Duration

Problem 9.1 B IS this model practical for predicting the outcome of a new auction?

According to us, this is not a practical model to predict the outcome of new auction, because it is based on ClosePrice, and the closePrice for a new auction is never known before the auction starts.

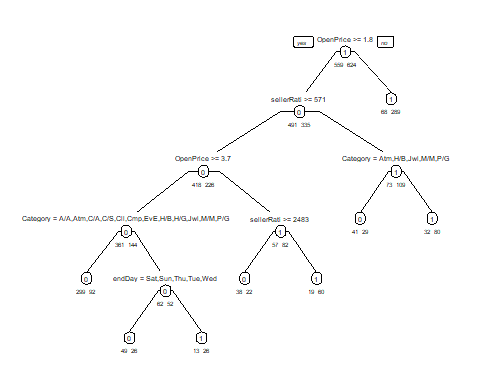
Problem 9.1 C Describe the interesting and uninteresting information that these rules provide.

Interesting –> Auctions with low sellerRating are competitive compared to those with high sellerRating  
Uninteresting –> Auctions with low close prices, compared to their open prices are not competitive. It is obvious, because it might not be bidded for, or bidded only once.

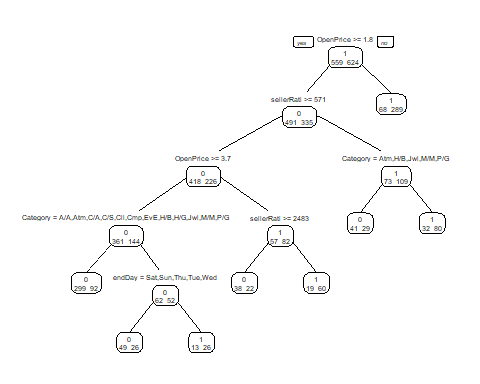
Problem 9.1 D Use only the predictors that can be used to predict the outcome of a new auction – Since we want to predict the outcome of the new auction, we won’t have the closePrice prior hand. So we will try to prepare a model without closePrice

#classification tree  
class.tree.forPrediction <- rpart(formula = Competitive. ~.-ClosePrice,  
 data = train.df,  
 control = rpart.control(maxdepth = 7, minbucket = 30),  
 method = "class")

prp(class.tree.forPrediction, type =1, extra = 1, under = TRUE,   
 split.font =1, varlen = -10)



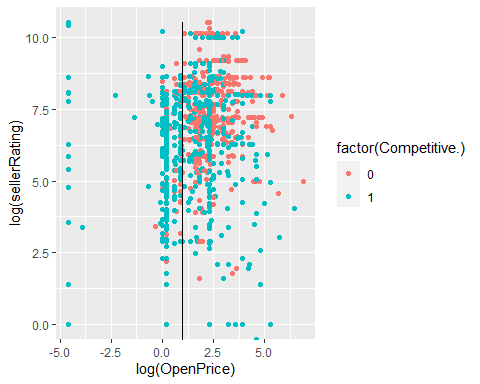
pruned.ct <- prune(class.tree.forPrediction,  
 cp = class.tree.forPrediction$cptable[  
 which.min(class.tree.forPrediction$cptable[,"xerror"]),"CP"])  
prp(pruned.ct, type = 1, extra = 1, split.font = 1, varlen = -10)



RULES IF (OpenPrice < 1.8) THEN CLASS = 1  
IF (OpenPrice >= 1.8) AND (SellerRating < 571) AND (Category != Atm,H/B,Jwl,M/M,P/G)  
THEN CLASS = 1  
IF (OpenPrice >= 1.8) AND (SellerRating < 571) AND (Category = Atm,H/B,Jwl,M/M,P/G)  
THEN CLASS = 0  
IF (SellerRating >= 571) AND (OpenPrice >= 3.7)  
THEN CLASS = 0  
IF (OpenPrice >= 1.8) AND (OpenPrice < 3.7) AND (SellerRating >= 2483)  
THEN CLASS = 0  
IF (OpenPrice >= 1.8) AND (SellerRating >= 571) AND (OpenPrice < 3.7) AND (SellerRating < 2483)  
THEN CLASS = 1

Problem 9.1 E

# Two best predictors: SellerRating and OpenPrice  
# Scatter Plot  
ggplot(eBayAuctions.df,  
 aes(x = log(OpenPrice), y = log(sellerRating))) +  
 geom\_point(aes(color = factor(Competitive.))) +  
 geom\_line(aes(x = log(2.6)))



This splitting seems to do a pretty good job of separating the two classes.

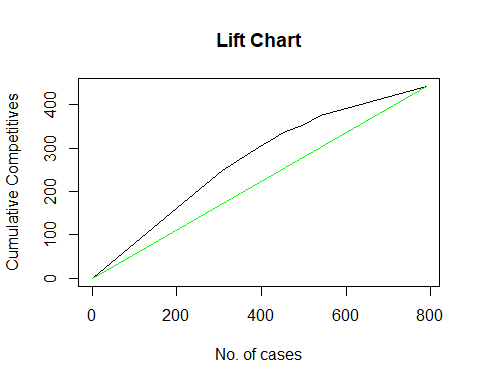
Problem 9.1. F

predictions\_class <- predict(pruned.ct,   
 validation.df,   
 type = 'class')  
# Confusion Matrix  
cm = table(validation.df$Competitive., predictions\_class)  
confusionMatrix(cm)

## Confusion Matrix and Statistics  
##   
## predictions\_class  
## 0 1  
## 0 243 104  
## 1 124 318  
##   
## Accuracy : 0.711   
## 95% CI : (0.678, 0.7425)  
## No Information Rate : 0.5349   
## P-Value [Acc > NIR] : <2e-16   
##   
## Kappa : 0.4172   
##   
## Mcnemar's Test P-Value : 0.2083   
##   
## Sensitivity : 0.6621   
## Specificity : 0.7536   
## Pos Pred Value : 0.7003   
## Neg Pred Value : 0.7195   
## Prevalence : 0.4651   
## Detection Rate : 0.3080   
## Detection Prevalence : 0.4398   
## Balanced Accuracy : 0.7078   
##   
## 'Positive' Class : 0   
##

predictions\_prob <- predict(pruned.ct,  
 validation.df,  
 type = 'prob')  
validation.gain.df <- data.frame(actual = validation.df$Competitive.,   
 prob = predictions\_prob[,2])

# Lift Chart  
# And then a lift chart  
# install.packages("gains")  
validation.gain.df$actual <- as.numeric(validation.gain.df$actual)  
gain <- suppressWarnings(gains(validation.gain.df$actual,  
 validation.gain.df$prob,  
 groups = dim(validation.gain.df)[1]))  
plot(c(0, gain$cume.pct.of.total \* sum(validation.gain.df$actual)) ~   
 c(0, gain$cume.obs),  
 xlab = "No. of cases",  
 ylab = "Cumulative Competitives",  
 main = "Lift Chart",  
 type = "l")  
lines(c(0, sum(validation.gain.df$actual)) ~ c(0, dim(validation.gain.df)[1]),   
 col = "green")



The accuracy of the model is only 70%. It does not fit very well.

Problem 9.1 G From th last tree, it is clear that the lower Open Price can attract more bidders. So the competitiveness of the auction basically depends on the seller.  
The first rule, the start node of the decision tree says that if the OpenPrice < 1.8, then it will lead to competitive auction. So to gain more bids, an OpenPrice of less than 1.8 is recommended.

Problem 9.2

flights\_delay.df <- read.csv("C:\\Users\\kkbal\\OneDrive\\Desktop\\Neu\\data mining\\Assignment-4\\FlightDelays.csv")

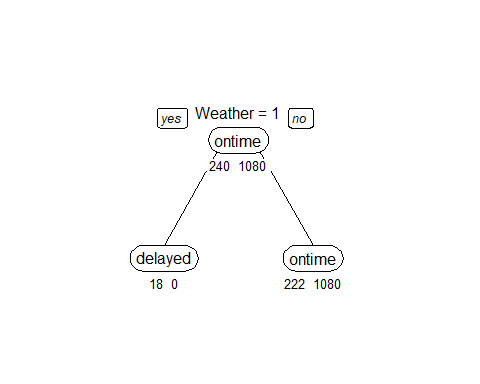
head(flights\_delay.df)

## CRS\_DEP\_TIME CARRIER DEP\_TIME DEST DISTANCE FL\_DATE FL\_NUM ORIGIN  
## 1 1455 OH 1455 JFK 184 1/1/2004 5935 BWI  
## 2 1640 DH 1640 JFK 213 1/1/2004 6155 DCA  
## 3 1245 DH 1245 LGA 229 1/1/2004 7208 IAD  
## 4 1715 DH 1709 LGA 229 1/1/2004 7215 IAD  
## 5 1039 DH 1035 LGA 229 1/1/2004 7792 IAD  
## 6 840 DH 839 JFK 228 1/1/2004 7800 IAD  
## Weather DAY\_WEEK DAY\_OF\_MONTH TAIL\_NUM Flight.Status  
## 1 0 4 1 N940CA ontime  
## 2 0 4 1 N405FJ ontime  
## 3 0 4 1 N695BR ontime  
## 4 0 4 1 N662BR ontime  
## 5 0 4 1 N698BR ontime  
## 6 0 4 1 N687BR ontime

# Converting DAY\_WEEK into a categorical variable  
flights\_delay.df$DAY\_WEEK <- as.factor(flights\_delay.df$DAY\_WEEK)

# Binning the scheduled DEPT\_TIME into 8 bins  
flights\_delay.df$DEP\_TIME <- cut(flights\_delay.df$DEP\_TIME, breaks = seq(600, 2200, by = 200), labels = 0:7)  
  
flights\_delay.df <- flights\_delay.df[, -11]  
# Split data  
set.seed(92)  
train.index <- sample(c(1:dim(flights\_delay.df)[1]), dim(flights\_delay.df)[1]\*0.6)  
train <- flights\_delay.df[train.index, ]  
valid <- flights\_delay.df[-train.index, ]

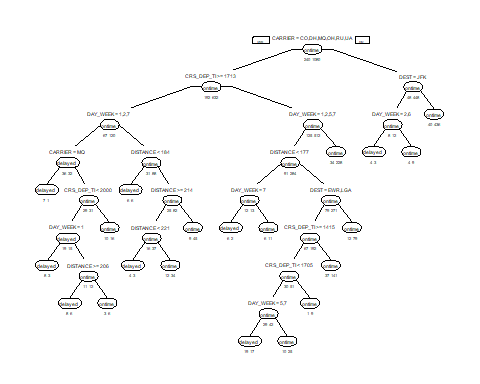
train <- train[, -c(3,6,7,11)]  
class.tree <- rpart(Flight.Status ~ ., data = train, method = "class")  
pruned.ct <- prune(class.tree, maxdepth = 8, cp = 0.001)  
prp(pruned.ct, type = 1, extra = 1, under = TRUE, split.font = 1, varlen = -10)



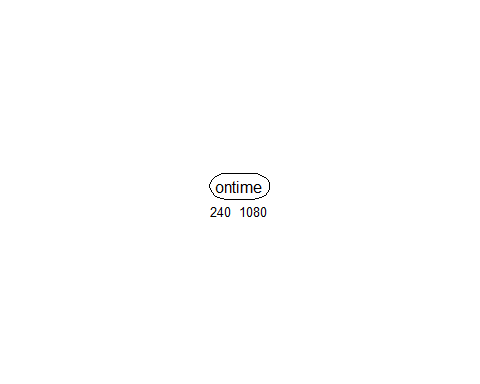
If (Weather >= 0.5) then class=delayed. If (Weather < 0.5) and (CRS\_DEP\_TIME = {4,6,7}) and (DAY\_WEEK = 7) and (CARRIER = {CO,DH,MQ,RU}) then class=delayed. If (Weather < 0.5) and (CRS\_DEP\_TIME = {4,6,7}) and (DAY\_WEEK = 7) and (CARRIER != {CO,DH,MQ,RU}) then class=ontime. If (Weather < 0.5) and (CRS\_DEP\_TIME = {4,6,7}) and (DAY\_WEEK != 7) then class=ontime. If (Weather < 0.5) and (CRS\_DEP\_TIME != {4,6,7}) and (ORIGIN = BWI) and (DAY\_WEEK = {2,7}) then class=delayed. If (Weather < 0.5) and (CRS\_DEP\_TIME != {4,6,7}) and (ORIGIN = BWI) and (DAY\_WEEK != {2,7}) then class=ontime. If (Weather < 0.5) and (CRS\_DEP\_TIME != {4,6,7}) and (ORIGIN != BWI) then class=ontime.

1. We cannot use this tree, because we do not have weather and carrier.

# Delete Weather  
train <- train[,-6]  
class\_tree2 <- rpart(Flight.Status ~ ., data = train, method = "class", maxdepth = 8, cp = 0.001)  
pruned\_ct2 <- prune(class\_tree2, cp = class\_tree2$cptable[which.min(class\_tree2$cptable[,"xerror"]),"CP"])  
prp(class\_tree2, type = 1, extra = 1, under = TRUE, split.font = 1, varlen = -10)



prp(pruned\_ct2, type = 1, extra = 1, under = TRUE, split.font = 1, varlen = -10)



i. The new observations will be classified as ontime.

Naive Rule.

print(class\_tree2$variable.importance)

## DAY\_WEEK CARRIER DISTANCE CRS\_DEP\_TIME DEST   
## 15.651391 15.501562 14.544986 14.477638 12.932566   
## ORIGIN   
## 6.320693

CRS\_DEP\_TIME, DISTANCE, CARRIER.

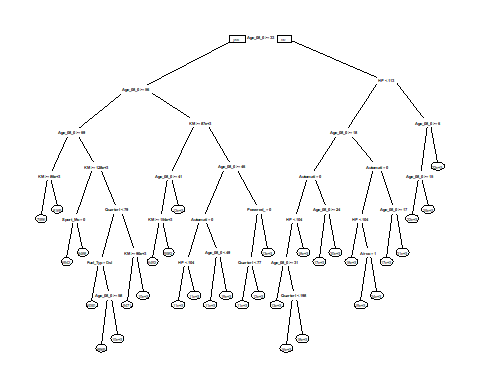
1. The pruned tree results in a single node as even if we add splits, it does not reduce the classification error in validation set.
2. The unpruned tree will cause overfitting But as the pruned tree has a lower error rate, which will avoid overfitting.
3. In our classification tree, there are only a few predictors considered in the tree. And all the splits are based on single predictor rather than combination of predictors, which might ignore the relationship between predictors. In addition, the different pre-processing of data in logistic regression might lead to the improvement. The departure time in the logistic regression is broken down into 16 bins, whereas in the classification tree it uses 8 bins Finally, this dataset is not very large, so a model-based method like logistic regression is likely to have more accuracy than a data-driven method like classification tree.

Problem 9.3

Corolla.df <- read.csv("C:\\Users\\kkbal\\OneDrive\\Desktop\\Neu\\data mining\\Assignment-4\\ToyotaCorolla.csv")  
set.seed(93)  
train.index <- sample(c(1:dim(Corolla.df)[1]), dim(Corolla.df)[1]\*0.6)  
training <- Corolla.df[train.index, ]  
validation <- Corolla.df[-train.index, ]

a

RT <- rpart(Price ~ Age\_08\_04 + KM + Fuel\_Type + HP + Automatic + Doors + Quarterly\_Tax + Mfr\_Guarantee + Guarantee\_Period + Airco + Automatic\_airco + CD\_Player + Powered\_Windows + Sport\_Model + Tow\_Bar,   
 method="anova", data = training,  
 minbucket = 1, maxdepth = 30, cp = 0.001, xval = 5)  
prp(RT)



print(RT$variable.importance)

## Age\_08\_04 KM Automatic\_airco HP   
## 9373061176 2863993167 2803395513 1225187858   
## Quarterly\_Tax Guarantee\_Period CD\_Player Fuel\_Type   
## 1206952131 381096442 299141754 84936083   
## Doors Airco Powered\_Windows Mfr\_Guarantee   
## 81993486 67787321 29972951 24366499   
## Sport\_Model Automatic   
## 20128494 2230532

Age\_08\_04, KM, Automatic\_airco, Quarterly\_Tax.

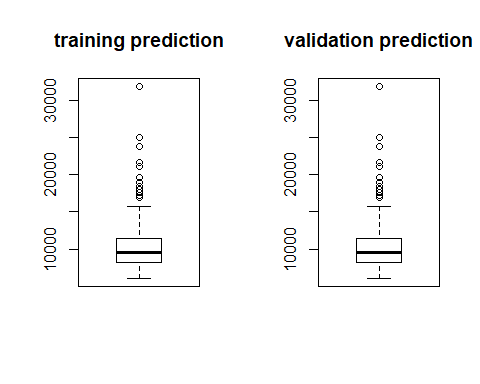
t <- predict(RT, training[,c(4,7,8,9,12,14,17,19,21,25,26,28,30,34,39)])  
v <- predict(RT, validation[,c(4,7,8,9,12,14,17,19,21,25,26,28,30,34,39)])  
t\_R <- sqrt(sum((training[, 3] - as.array(t))^2)/nrow(as.array(t)))  
v\_R <- sqrt(sum((validation[, 3] - as.array(v))^2)/nrow(as.array(v)))  
t\_R

## [1] 972.677

v\_R

## [1] 1292.571

par(mfrow = c(1, 2))  
boxplot(t, main = "training prediction")  
boxplot(v, main = "validation prediction")

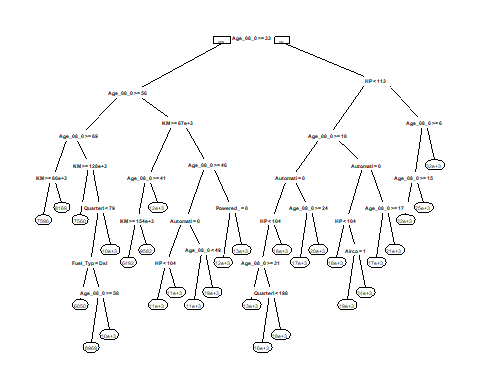


par(mfrow = c(1, 1))

Training set has RMSE score of 972.677 while Validation set has a RMSE score of 1292.571 which is 32.89% higher than that of training set This might be a result of overfitting

1. The Regression tree that we created only contains a few rules to generate prediction for new data where the prediction is just a mean of all prices , which will be same as the corresponding actual price in training data.

prune\_rt <- prune(RT,  
 cp = RT$cptable[which.min(RT$cptable[,"xerror"]),"CP"])  
prp(prune\_rt)



validation\_P <- predict(prune\_rt, validation[,c(4,7,8,9,12,14,17,19,21,25,26,28,30,34,39)])  
valid\_R <- sqrt(sum((validation[, 3] - as.array(validation\_P))^2)/nrow(as.array(validation\_P)))  
valid\_R

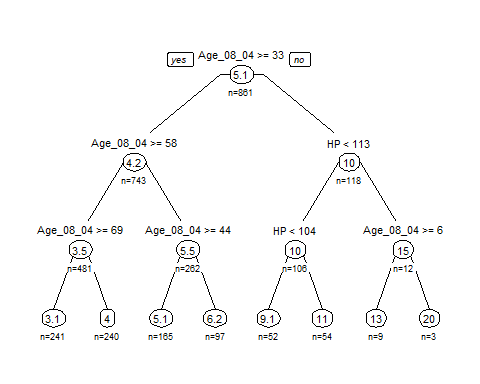
## [1] 1286.394

After Pruning, validation set has 1302.968 of RMSE, which is smaller than before.

summary(Corolla.df$Price)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 4350 8450 9900 10731 11950 32500

Corolla.df$Binned\_Price <- cut(Corolla.df$Price, breaks = seq(4300, 32500, by = 1410))  
set.seed(931)  
train.index <- sample(c(1:dim(Corolla.df)[1]), dim(Corolla.df)[1]\*0.6)  
training2 <- Corolla.df[train.index, ]  
validation2 <- Corolla.df[-train.index, ]  
ctree <- rpart(Binned\_Price ~ Age\_08\_04 + KM + Fuel\_Type + HP + Automatic + Doors + Quarterly\_Tax + Mfr\_Guarantee + Guarantee\_Period + Airco + Automatic\_airco + CD\_Player + Powered\_Windows + Sport\_Model + Tow\_Bar,   
 method="anova", data = training2, minbucket = 1)  
prp(ctree, type = 1, extra = 1, under = TRUE, split.font = 1, varlen = -10)



1. The two trees are different as the choice of a split depends on the ordering of observation values and not on these values they are sensitive to changes in data causing huge splits

new.data <- data.frame(Age\_08\_04 = 77,  
 KM = 117000,  
 Fuel\_Type = "Petrol",  
 HP = 110,  
 Automatic = 0,  
 Doors = 5,  
 Quarterly\_Tax = 100,  
 Mfr\_Guarantee = 0,  
 Guarantee\_Period = 3,  
 Airco = 1,  
 Automatic\_airco = 0,  
 CD\_Player = 0,  
 Powered\_Windows = 0,  
 Sport\_Model = 0,  
 Tow\_Bar = 1)  
R.pred <- predict(RT, new.data)  
Class.pred <- predict(ctree, new.data)  
R.pred

## 1   
## 7596.209

Class.pred \* 1410 + 4300

## 1   
## 8617.759

validation\_p2 <- predict(ctree, validation2[,c(4,7,8,9,12,14,17,19,21,25,26,28,30,34,39)])  
validation\_p2 <- validation\_p2 \* 1410 + 4300  
validation2\_R <- sqrt(sum((validation2[, 3] - as.array(validation\_p2))^2)/nrow(as.array(validation\_p2)))  
validation2\_R

## [1] 1508.533

The prediction from Regression Tree is 7596.209 and Class Tree is 8617.759 and have a difference of 1000+. The accuracy of Regression Tree is higher than Class Tree, because Class Tree has 20 bins but the Regression Tree provides accurate actual numbers.

Problem 10.1 [15 points] Financial Condition of Banks. The file Banks.csv includes data on a sample of 20 banks. The “Financial Condition” column records the judgment of an expert on the financial condition of each bank. This outcome variable takes one of two possible values—weak or strong—according to the financial condition of the bank. The predictors are two ratios used in the financial analysis of banks: TotLns&Lses/Assets is the ratio of total loans and leases to total assets and TotExp/Assets is the ratio of total expenses to total assets. The target is to use the two ratios for classifying the financial condition of a new bank. Run a logistic regression model (on the entire dataset) that models the status of a bank as a function of the two financial measures provided. Specify the success class as weak (this is similar to creating a dummy that is 1 for financially weak banks and 0 otherwise), and use the default cutoff value of 0.5.

financial<-read\_csv('C:\\Users\\kkbal\\OneDrive\\Desktop\\Neu\\data mining\\Assignment-4\\Banks.csv')

## Parsed with column specification:  
## cols(  
## Obs = col\_double(),  
## `Financial Condition` = col\_double(),  
## `TotCap/Assets` = col\_double(),  
## `TotExp/Assets` = col\_double(),  
## `TotLns&Lses/Assets` = col\_double()  
## )

head(financial)

## # A tibble: 6 x 5  
## Obs `Financial Condit~ `TotCap/Assets` `TotExp/Assets` `TotLns&Lses/As~  
## <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 1 1 9.7 0.12 0.65  
## 2 2 1 1 0.11 0.62  
## 3 3 1 6.9 0.09 1.02  
## 4 4 1 5.8 0.1 0.67  
## 5 5 1 4.3 0.11 0.69  
## 6 6 1 9.1 0.13 0.74

Run a logistic regression model (on the entire dataset) that models the status of a bank as a function of the two financial measures provided. Specify the success class as weak (this is similar to creating a dummy that is 1 for financially weak banks and 0 otherwise), and use the defaultcutoff value of 0.5.

colnames(financial)

## [1] "Obs" "Financial Condition" "TotCap/Assets"   
## [4] "TotExp/Assets" "TotLns&Lses/Assets"

financial.log<-glm(`Financial Condition`~ `TotExp/Assets`+`TotLns&Lses/Assets`, data=financial,family = 'binomial')  
financial.log

##   
## Call: glm(formula = `Financial Condition` ~ `TotExp/Assets` + `TotLns&Lses/Assets`,   
## family = "binomial", data = financial)  
##   
## Coefficients:  
## (Intercept) `TotExp/Assets` `TotLns&Lses/Assets`   
## -14.721 89.834 8.371   
##   
## Degrees of Freedom: 19 Total (i.e. Null); 17 Residual  
## Null Deviance: 27.73   
## Residual Deviance: 13.15 AIC: 19.15

1. Write the estimated equation that associates the financial condition of a bank with its two predictors in three formats:
2. The logit as a function of the predictors

The logit function of financial.condition=weak

logit<- -14.721+ 89.834\* (TotExp/Assets)+ 8.371\*(TotLns&Lses/Assets) logit

1. The odds as a function of the predictors

The odds function of financial.condition=weak

odds<- e^(-14.721+ 89.834\* (TotExp/Assets)+ 8.371\*(TotLns&Lses/Assets))

odds

1. The probability as a function of the predictors

The Probability of financial.condition=weak

probability<-1/(1+e^(-14.721+ 89.834\* (TotExp/Assets)+ 8.371\*(TotLns&Lses/Assets)) probability

1. Consider a new bank whose total loans and leases/assets ratio = 0.6 and total expenses/assets ratio = 0.11. From your logistic regression model, estimate the following four quantities for this bank (use R to do all the intermediate calculations; show your final answers to four decimal places): the logit, the odds, the probability of being financially weak, and the classification of the bank (use cutoff = 0.5).

new.bank<-data.frame('TotLns&Lses/Assets'=0.6,'TotExp/Assets'=0.11)  
new.bank

## TotLns.Lses.Assets TotExp.Assets  
## 1 0.6 0.11

### the logit function  
  
logit<- round(-14.721+ 89.834\* (new.bank$TotExp.Assets)+ 8.371\*(new.bank$TotLns.Lses.Assets),4)  
logit

## [1] 0.1833

### the odds function  
  
odds<-round(exp(logit),4)  
odds

## [1] 1.2012

### the probability of being financially weak  
  
probability<-round(1/(1+exp(logit)),4)  
probability

## [1] 0.4543

### the classification of the new bank  
  
classfication<- ifelse(probability>0.5,1,0)  
classfication

## [1] 0

1. The cutoff value of 0.5 is used in conjunction with the probability of being financially weak. Compute the threshold that should be used if we want to make a classification based on the odds of being financially weak, and the threshold for the corresponding logit.

solution

If the cut off value is 0.5 based on odds then the threshold of odds is equal to 1 and the corresponding logit is equal to 0.

1. Interpret the estimated coefficient for the total loans & leases to total assets ratio (TotLns&Lses/Assets) in terms of the odds of being financially weak.

Solution

The co-efficient of total loans & leases to total assets ratio has postive value so the odds of bank to be financial weak will also have postive effect. If this increases there is more chance for bank to be classified as financially weak.(if co-efficient is multiplied with a postive observation and taking in account of logit function the resultant value indicates how much impact that this observation affects the classification of bank being financially weak).

1. When a bank that is in poor financial condition is misclassified as financially strong, the misclassification cost is much higher than when a financially strong bank is misclassified as weak. To minimize the expected cost of misclassification, should the cutoff value for classification (which is currently at 0.5) be increased or decreased?

solution

In this scenario to minimize the cost misclassification we should decrease the cut off value.

Problem 10.2 [15 points] Identifying Good System Administrators. A management consultant is studying the roles played by experience and training in a system administrator’s ability to complete a set of tasks in a specified amount of time. In particular, she is interested in discriminating between administrators who are able to complete given tasks within a specified time and those who are not. Data are collected on the performance of 75 randomly selected administrators. They are stored in the file SystemAdministrators.csv.

The variable Experience measures months of full-time system administrator experience, while Training measures the number of relevant training credits. The outcome variable Completed is either Yes or No, according to whether or not the administrator completed the tasks.

system<-read\_csv('C:\\Users\\kkbal\\OneDrive\\Desktop\\Neu\\data mining\\Assignment-4\\SystemAdministrators.csv')

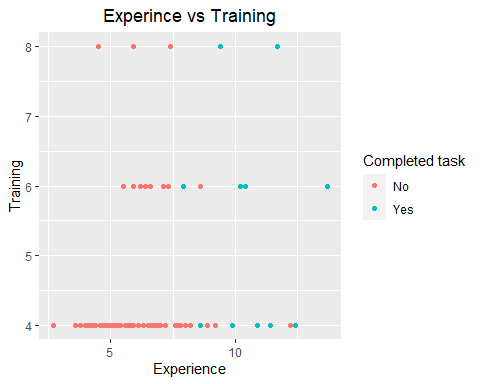
## Parsed with column specification:  
## cols(  
## Experience = col\_double(),  
## Training = col\_double(),  
## `Completed task` = col\_character()  
## )

system$`Completed task`<-as.factor(system$`Completed task`)  
head(system)

## # A tibble: 6 x 3  
## Experience Training `Completed task`  
## <dbl> <dbl> <fct>   
## 1 10.9 4 Yes   
## 2 9.9 4 Yes   
## 3 10.4 6 Yes   
## 4 13.7 6 Yes   
## 5 9.4 8 Yes   
## 6 12.4 4 Yes

1. Create a scatter plot of Experience vs. Training using color or symbol to distinguish programmers who completed the task from those who did not complete it. Which predictor(s) appear(s) potentially useful for classifying task completion?

ggplot(system,aes(Experience,Training,color=`Completed task`))+geom\_point()+ggtitle('Experince vs Training')+theme(plot.title = element\_text(hjust= 0.5))



1. Run a logistic regression model with both predictors using the entire dataset as training data. Among those who completed the task, what is the percentage of programmers incorrectly classified as failing to complete the task?

log.system<-glm(`Completed task`~.,data=system,family='binomial')  
log.pred<-predict(log.system,system)  
summary(log.system)

##   
## Call:  
## glm(formula = `Completed task` ~ ., family = "binomial", data = system)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.65306 -0.34959 -0.17479 -0.08196 2.21813   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -10.9813 2.8919 -3.797 0.000146 \*\*\*  
## Experience 1.1269 0.2909 3.874 0.000107 \*\*\*  
## Training 0.1805 0.3386 0.533 0.593970   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 75.060 on 74 degrees of freedom  
## Residual deviance: 35.713 on 72 degrees of freedom  
## AIC: 41.713  
##   
## Number of Fisher Scoring iterations: 6

confusionMatrix(factor(ifelse(log.pred>0.5,'Yes','No')) ,factor(system$`Completed task`))

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction No Yes  
## No 59 6  
## Yes 1 9  
##   
## Accuracy : 0.9067   
## 95% CI : (0.8171, 0.9616)  
## No Information Rate : 0.8   
## P-Value [Acc > NIR] : 0.01041   
##   
## Kappa : 0.6667   
##   
## Mcnemar's Test P-Value : 0.13057   
##   
## Sensitivity : 0.9833   
## Specificity : 0.6000   
## Pos Pred Value : 0.9077   
## Neg Pred Value : 0.9000   
## Prevalence : 0.8000   
## Detection Rate : 0.7867   
## Detection Prevalence : 0.8667   
## Balanced Accuracy : 0.7917   
##   
## 'Positive' Class : No   
##

Among programmers completed the task and incorrectly classified as failing to complete the task = (6/15\*100=40 percent)

1. To decrease the percentage in part (b), should the cutoff probability be increased or decreased?

Solution

To decrease the percentage, We have to decrease the threshold of the cut off probability.

1. How much experience must be accumulated by a programmer with 4 years of training before his or her estimated probability of completing the task exceeds 0.5?

solution

prob<- -10.9813 + 1.1269*(Experince)+ 0.1805*(Training)

0.5<- -10.9813+1.1269*(Experience)+ 0.1805*(4)

Experience<- (0.5+10.9813-0.722)/1.1269  
  
Experience

## [1] 9.547697

So the Experience should be greater than 9.547697 to make the probability of completing task to exceed 0.5

Competitive Auctions on eBay.com. The file eBayAuctions.csv contains information on 1972 auctions transacted on eBay.com during May–June 2004. The goal is to use these data to build a model that will distinguish competitive auctions from noncompetitive ones. A competitive auction is defined as an auction with at least two bids placed on the item being auctioned. The data include variables that describe the item (auction category), the seller (his or her eBay rating), and the auction terms that the seller selected (auction duration, opening price, currency, day of week of auction close). In addition, we have the price at which the auction closed. The goal is to predict whether or not an auction of interest will be competitive.

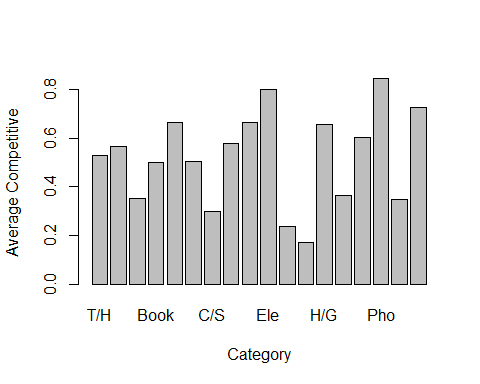
ebay\_auction\_data<-read.csv('C:\\Users\\kkbal\\OneDrive\\Desktop\\Neu\\data mining\\Assignment-4\\eBayAuctions.csv')  
ebay\_auction\_data$Duration <- factor(ebay\_auction\_data$Duration, levels = c(1,3,5,7,10),   
 labels = c("1", "3", "5", "7", "10"))  
ebay\_auction\_data$currency <- factor(ebay\_auction\_data$currency)  
ebay\_auction\_data$Category <- factor(ebay\_auction\_data$Category)  
ebay\_auction\_data$endDay <- factor(ebay\_auction\_data$endDay)  
ebay\_auction\_data$Category <- relevel(ebay\_auction\_data$Category, ref = "Toys/Hobbies")  
ebay\_auction\_data$currency <- relevel(ebay\_auction\_data$currency, ref = "US")  
ebay\_auction\_data$endDay <- relevel(ebay\_auction\_data$endDay, ref = "Wed")  
ebay\_auction\_data$Duration <- relevel(ebay\_auction\_data$Duration, ref = "7")  
head(ebay\_auction\_data)

## Category currency sellerRating Duration endDay ClosePrice  
## 1 Music/Movie/Game US 3249 5 Mon 0.01  
## 2 Music/Movie/Game US 3249 5 Mon 0.01  
## 3 Music/Movie/Game US 3249 5 Mon 0.01  
## 4 Music/Movie/Game US 3249 5 Mon 0.01  
## 5 Music/Movie/Game US 3249 5 Mon 0.01  
## 6 Music/Movie/Game US 3249 5 Mon 0.01  
## OpenPrice Competitive.  
## 1 0.01 0  
## 2 0.01 0  
## 3 0.01 0  
## 4 0.01 0  
## 5 0.01 0  
## 6 0.01 0

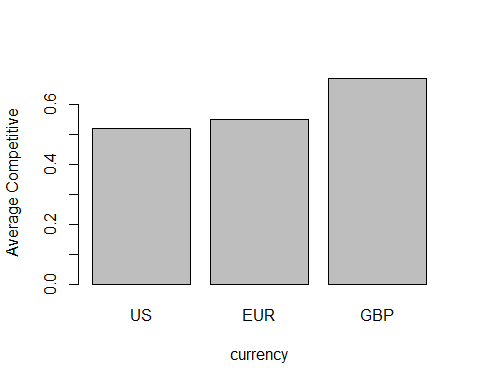
Data preprocessing. Create dummy variables for the categorical predictors. These include Category (18 categories), Currency (USD, GBP, Euro), EndDay (Monday–Sunday), and Duration (1, 3, 5, 7, or 10 days).

a

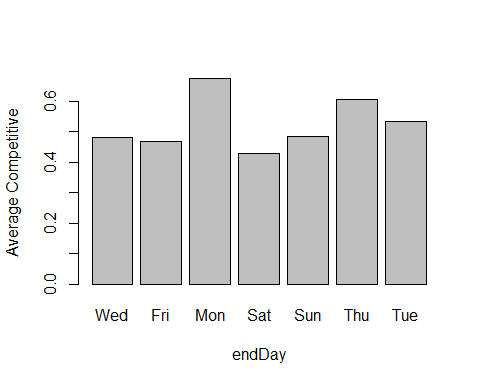
barplot(  
 aggregate(  
 ebay\_auction\_data$`Competitive.` == 1,  
 by = list(ebay\_auction\_data$Category),  
 mean,  
 rm.na = T  
 )[, 2],  
 xlab = "Category",  
 ylab = "Average Competitive",  
 names.arg = c(  
 "T/H",  
 "A/A/C",  
 "Auto",  
 "Book",  
 "B/I",  
 "C/A",  
 "C/S",  
 "Col",  
 "Com",  
 "Ele",  
 "Eve",  
 "H/B",  
 "H/G",  
 "Jew",  
 "M/M/G",  
 "Pho",  
 "P/G",  
 "SG"  
 )  
)



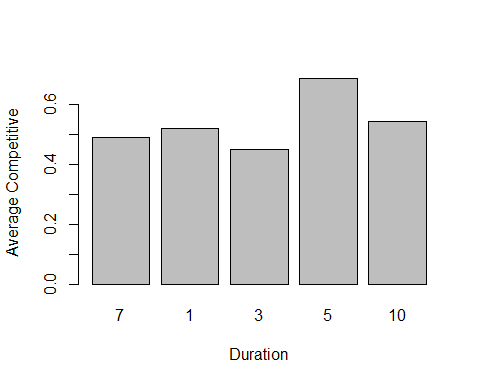
barplot(  
 aggregate(  
 ebay\_auction\_data$`Competitive.` == 1,  
 by = list(ebay\_auction\_data$currency),  
 mean,  
 rm.na = T  
 )[, 2],  
 xlab = "currency",  
 ylab = "Average Competitive",  
 names.arg = c("US", "EUR", "GBP")  
)



barplot(  
 aggregate(  
 ebay\_auction\_data$`Competitive.` == 1,  
 by = list(ebay\_auction\_data$endDay),  
 mean,  
 rm.na = T  
 )[, 2],  
 xlab = "endDay",  
 ylab = "Average Competitive",  
 names.arg = c("Wed", "Fri", "Mon", "Sat", "Sun", "Thu", "Tue")  
)



barplot(  
 aggregate(  
 ebay\_auction\_data$`Competitive.` == 1,  
 by = list(ebay\_auction\_data$Duration),  
 mean,  
 rm.na = T  
 )[, 2],  
 xlab = "Duration",  
 ylab = "Average Competitive",  
 names.arg = c("7", "1", "3", "5", "10")  
)



summarise(group\_by(ebay\_auction\_data, Category),  
 "Competitive." = mean(`Competitive.`))

## # A tibble: 18 x 2  
## Category Competitive.  
## <fct> <dbl>  
## 1 Toys/Hobbies 0.530  
## 2 Antique/Art/Craft 0.565  
## 3 Automotive 0.354  
## 4 Books 0.5   
## 5 Business/Industrial 0.667  
## 6 Clothing/Accessories 0.504  
## 7 Coins/Stamps 0.297  
## 8 Collectibles 0.577  
## 9 Computer 0.667  
## 10 Electronics 0.8   
## 11 EverythingElse 0.235  
## 12 Health/Beauty 0.172  
## 13 Home/Garden 0.657  
## 14 Jewelry 0.366  
## 15 Music/Movie/Game 0.603  
## 16 Photography 0.846  
## 17 Pottery/Glass 0.35   
## 18 SportingGoods 0.726

summarise(group\_by(ebay\_auction\_data, currency),  
 "Competitive." = mean(`Competitive.`))

## # A tibble: 3 x 2  
## currency Competitive.  
## <fct> <dbl>  
## 1 US 0.519  
## 2 EUR 0.552  
## 3 GBP 0.687

summarise(group\_by(ebay\_auction\_data, endDay),  
 "Competitive." = mean(`Competitive.`))

## # A tibble: 7 x 2  
## endDay Competitive.  
## <fct> <dbl>  
## 1 Wed 0.48   
## 2 Fri 0.467  
## 3 Mon 0.673  
## 4 Sat 0.427  
## 5 Sun 0.485  
## 6 Thu 0.604  
## 7 Tue 0.532

summarise(group\_by(ebay\_auction\_data, Duration),  
 "Competitive." = mean(`Competitive.`))

## # A tibble: 5 x 2  
## Duration Competitive.  
## <fct> <dbl>  
## 1 7 0.489  
## 2 1 0.522  
## 3 3 0.451  
## 4 5 0.687  
## 5 10 0.545

we can put bin into different categories into three buckets 0~0.4, 0.4~0.6, 0.6~0.8 as theres no difference in currencies we dont combine them. For endDay Monday, Tuesday and Thursday have similar compititiveness so we combine them for duration we combine competitiveness with more than average of 5.

Combining data and removing columns

ebay\_auction\_data$Category\_low <-  
 ebay\_auction\_data$Category %in% c(  
 "Automotive",  
 "Coins/Stamps",  
 "default",  
 "Health/Beauty",  
 "Jewelry",  
 "Pottery/Glass"  
 )  
ebay\_auction\_data$Category\_mid <-  
 ebay\_auction\_data$Category %in% c(  
 "Toys/Hobbies",  
 "Antique/Art/Craft",  
 "Books",  
 "Clothing/Accessories",  
 "Collectibles"  
 )  
ebay\_auction\_data$endDay\_Mon\_Tue\_Thu <-  
 ebay\_auction\_data$endDay %in% c("Mon", "Tue", "Thu")  
ebay\_auction\_data$Duration\_5 <- ebay\_auction\_data$Duration %in% "5"  
ebay\_auction\_data <- ebay\_auction\_data[, c(2, 3, 6, 7, 9, 10, 11, 12, 8)]

b

partitioning data with training and validation

set.seed(0)  
train.index <-  
 sample(c(1:dim(ebay\_auction\_data)[1]), dim(ebay\_auction\_data)[1] \* 0.6)  
train <- ebay\_auction\_data[train.index,]  
valid <- ebay\_auction\_data[-train.index,]

logistic regression and displaying coefficients

lm.fit <- glm(`Competitive.` ~ ., data = train, family = "binomial")

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

data.frame(summary(lm.fit)$coefficients, odds = exp(coef(lm.fit)))

## Estimate Std..Error z.value Pr...z..  
## (Intercept) -1.439054e-01 1.743291e-01 -0.8254814 4.090983e-01  
## currencyEUR -5.931289e-01 1.834553e-01 -3.2330976 1.224557e-03  
## currencyGBP 4.545776e-01 3.078941e-01 1.4764091 1.398341e-01  
## sellerRating -3.640814e-05 1.563849e-05 -2.3281103 1.990625e-02  
## ClosePrice 1.365526e-01 1.408456e-02 9.6952004 3.160165e-22  
## OpenPrice -1.534199e-01 1.546315e-02 -9.9216425 3.351943e-23  
## Category\_lowTRUE -1.114796e+00 2.252968e-01 -4.9481218 7.493301e-07  
## Category\_midTRUE -1.507573e-01 1.640926e-01 -0.9187330 3.582353e-01  
## endDay\_Mon\_Tue\_ThuTRUE 3.546795e-01 1.696413e-01 2.0907620 3.654940e-02  
## Duration\_5TRUE 4.916009e-01 2.127452e-01 2.3107492 2.084671e-02  
## odds  
## (Intercept) 0.8659696  
## currencyEUR 0.5525956  
## currencyGBP 1.5755078  
## sellerRating 0.9999636  
## ClosePrice 1.1463152  
## OpenPrice 0.8577695  
## Category\_lowTRUE 0.3279822  
## Category\_midTRUE 0.8600564  
## endDay\_Mon\_Tue\_ThuTRUE 1.4257236  
## Duration\_5TRUE 1.6349315

round(data.frame(summary(lm.fit)$coefficients, odds = exp(coef(lm.fit))), 5)

## Estimate Std..Error z.value Pr...z.. odds  
## (Intercept) -0.14391 0.17433 -0.82548 0.40910 0.86597  
## currencyEUR -0.59313 0.18346 -3.23310 0.00122 0.55260  
## currencyGBP 0.45458 0.30789 1.47641 0.13983 1.57551  
## sellerRating -0.00004 0.00002 -2.32811 0.01991 0.99996  
## ClosePrice 0.13655 0.01408 9.69520 0.00000 1.14632  
## OpenPrice -0.15342 0.01546 -9.92164 0.00000 0.85777  
## Category\_lowTRUE -1.11480 0.22530 -4.94812 0.00000 0.32798  
## Category\_midTRUE -0.15076 0.16409 -0.91873 0.35824 0.86006  
## endDay\_Mon\_Tue\_ThuTRUE 0.35468 0.16964 2.09076 0.03655 1.42572  
## Duration\_5TRUE 0.49160 0.21275 2.31075 0.02085 1.63493

pred1 <- predict(lm.fit, valid, type = "response")  
confusionMatrix(as.factor(ifelse(pred1 > 0.5, 1, 0)), as.factor(valid$`Competitive.`))

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 314 92  
## 1 62 321  
##   
## Accuracy : 0.8048   
## 95% CI : (0.7754, 0.8319)  
## No Information Rate : 0.5234   
## P-Value [Acc > NIR] : < 2e-16   
##   
## Kappa : 0.6102   
##   
## Mcnemar's Test P-Value : 0.01945   
##   
## Sensitivity : 0.8351   
## Specificity : 0.7772   
## Pos Pred Value : 0.7734   
## Neg Pred Value : 0.8381   
## Prevalence : 0.4766   
## Detection Rate : 0.3980   
## Detection Prevalence : 0.5146   
## Balanced Accuracy : 0.8062   
##   
## 'Positive' Class : 0   
##

summary(lm.fit)

##   
## Call:  
## glm(formula = Competitive. ~ ., family = "binomial", data = train)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -4.4835 -0.9109 0.0007 0.8438 2.4293   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -1.439e-01 1.743e-01 -0.825 0.40910   
## currencyEUR -5.931e-01 1.835e-01 -3.233 0.00122 \*\*   
## currencyGBP 4.546e-01 3.079e-01 1.476 0.13983   
## sellerRating -3.641e-05 1.564e-05 -2.328 0.01991 \*   
## ClosePrice 1.366e-01 1.408e-02 9.695 < 2e-16 \*\*\*  
## OpenPrice -1.534e-01 1.546e-02 -9.922 < 2e-16 \*\*\*  
## Category\_lowTRUE -1.115e+00 2.253e-01 -4.948 7.49e-07 \*\*\*  
## Category\_midTRUE -1.508e-01 1.641e-01 -0.919 0.35824   
## endDay\_Mon\_Tue\_ThuTRUE 3.547e-01 1.696e-01 2.091 0.03655 \*   
## Duration\_5TRUE 4.916e-01 2.127e-01 2.311 0.02085 \*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 1627.2 on 1182 degrees of freedom  
## Residual deviance: 1160.5 on 1173 degrees of freedom  
## AIC: 1180.5  
##   
## Number of Fisher Scoring iterations: 8

c Removing closing price

lm.fit2 <-  
 glm(`Competitive.` ~ . - ClosePrice,  
 data = train,  
 family = "binomial")  
data.frame(summary(lm.fit2)$coefficients, odds = exp(coef(lm.fit2)))

## Estimate Std..Error z.value Pr...z..  
## (Intercept) 3.234748e-01 1.478393e-01 2.1880161 2.866843e-02  
## currencyEUR -6.818093e-02 1.508840e-01 -0.4518765 6.513580e-01  
## currencyGBP 4.207008e-01 2.822627e-01 1.4904585 1.361037e-01  
## sellerRating -3.510814e-05 1.141944e-05 -3.0744179 2.109138e-03  
## OpenPrice -6.295569e-03 3.151851e-03 -1.9974198 4.577960e-02  
## Category\_lowTRUE -1.275153e+00 1.898966e-01 -6.7149861 1.880842e-11  
## Category\_midTRUE -1.818441e-01 1.423994e-01 -1.2770007 2.016020e-01  
## endDay\_Mon\_Tue\_ThuTRUE 4.800924e-01 1.456843e-01 3.2954308 9.827092e-04  
## Duration\_5TRUE 7.266363e-01 1.825475e-01 3.9805336 6.876073e-05  
## odds  
## (Intercept) 1.3819213  
## currencyEUR 0.9340915  
## currencyGBP 1.5230286  
## sellerRating 0.9999649  
## OpenPrice 0.9937242  
## Category\_lowTRUE 0.2793881  
## Category\_midTRUE 0.8337313  
## endDay\_Mon\_Tue\_ThuTRUE 1.6162238  
## Duration\_5TRUE 2.0681123

round(data.frame(summary(lm.fit2)$coefficients, odds = exp(coef(lm.fit2))), 5)

## Estimate Std..Error z.value Pr...z.. odds  
## (Intercept) 0.32347 0.14784 2.18802 0.02867 1.38192  
## currencyEUR -0.06818 0.15088 -0.45188 0.65136 0.93409  
## currencyGBP 0.42070 0.28226 1.49046 0.13610 1.52303  
## sellerRating -0.00004 0.00001 -3.07442 0.00211 0.99996  
## OpenPrice -0.00630 0.00315 -1.99742 0.04578 0.99372  
## Category\_lowTRUE -1.27515 0.18990 -6.71499 0.00000 0.27939  
## Category\_midTRUE -0.18184 0.14240 -1.27700 0.20160 0.83373  
## endDay\_Mon\_Tue\_ThuTRUE 0.48009 0.14568 3.29543 0.00098 1.61622  
## Duration\_5TRUE 0.72664 0.18255 3.98053 0.00007 2.06811

pred2 <- predict(lm.fit2, valid, type = "response")  
confusionMatrix(as.factor(ifelse(pred2 > 0.5,1,0)),as.factor(valid$`Competitive.`))

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 165 84  
## 1 211 329  
##   
## Accuracy : 0.6261   
## 95% CI : (0.5913, 0.66)  
## No Information Rate : 0.5234   
## P-Value [Acc > NIR] : 3.824e-09   
##   
## Kappa : 0.2391   
##   
## Mcnemar's Test P-Value : 2.201e-13   
##   
## Sensitivity : 0.4388   
## Specificity : 0.7966   
## Pos Pred Value : 0.6627   
## Neg Pred Value : 0.6093   
## Prevalence : 0.4766   
## Detection Rate : 0.2091   
## Detection Prevalence : 0.3156   
## Balanced Accuracy : 0.6177   
##   
## 'Positive' Class : 0   
##

summary(lm.fit2)

##   
## Call:  
## glm(formula = Competitive. ~ . - ClosePrice, family = "binomial",   
## data = train)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.8454 -1.1764 0.6592 1.0693 1.9789   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 3.235e-01 1.478e-01 2.188 0.028668 \*   
## currencyEUR -6.818e-02 1.509e-01 -0.452 0.651358   
## currencyGBP 4.207e-01 2.823e-01 1.490 0.136104   
## sellerRating -3.511e-05 1.142e-05 -3.074 0.002109 \*\*   
## OpenPrice -6.296e-03 3.152e-03 -1.997 0.045780 \*   
## Category\_lowTRUE -1.275e+00 1.899e-01 -6.715 1.88e-11 \*\*\*  
## Category\_midTRUE -1.818e-01 1.424e-01 -1.277 0.201602   
## endDay\_Mon\_Tue\_ThuTRUE 4.801e-01 1.457e-01 3.295 0.000983 \*\*\*  
## Duration\_5TRUE 7.266e-01 1.825e-01 3.981 6.88e-05 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 1627.2 on 1182 degrees of freedom  
## Residual deviance: 1486.2 on 1174 degrees of freedom  
## AIC: 1504.2  
##   
## Number of Fisher Scoring iterations: 4

The orignal model with all the variables yielded higher accuracy than the model with reduced variables

d  
The coef of Closing price is 0.1366 i.e 1.14637 times auction with higher Closing price has more odds of being competitive than auction with lower Closing price keeping rest of variables same.  
The closing price doesn’t have any real significance as its impossilbe to know closing price before auction.  
Statistically, it is significant since it explains the competitiveness of auction with high accuracy.

e

if(!require(glmulti)){  
 install.packages("glmulti")  
}  
if(!require(MASS)){  
 install.packages("MASS")  
}

## Loading required package: MASS

##   
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':  
##   
## select

lm.step <- stepAIC(lm.fit2, trace = TRUE)

## Start: AIC=1504.23  
## Competitive. ~ (currency + sellerRating + ClosePrice + OpenPrice +   
## Category\_low + Category\_mid + endDay\_Mon\_Tue\_Thu + Duration\_5) -   
## ClosePrice  
##   
## Df Deviance AIC  
## - currency 2 1489.4 1503.4  
## - Category\_mid 1 1487.9 1503.9  
## <none> 1486.2 1504.2  
## - OpenPrice 1 1491.2 1507.2  
## - sellerRating 1 1495.8 1511.8  
## - endDay\_Mon\_Tue\_Thu 1 1497.1 1513.1  
## - Duration\_5 1 1502.5 1518.5  
## - Category\_low 1 1533.5 1549.5  
##   
## Step: AIC=1503.39  
## Competitive. ~ sellerRating + OpenPrice + Category\_low + Category\_mid +   
## endDay\_Mon\_Tue\_Thu + Duration\_5  
##   
## Df Deviance AIC  
## <none> 1489.4 1503.4  
## - Category\_mid 1 1491.7 1503.7  
## - OpenPrice 1 1494.3 1506.3  
## - sellerRating 1 1500.2 1512.2  
## - Duration\_5 1 1504.9 1516.9  
## - endDay\_Mon\_Tue\_Thu 1 1507.0 1519.0  
## - Category\_low 1 1537.7 1549.7

it takes long time so cant perform exhaustive search because of lack of resources.

#glmulti(lm.fit2)

From the best fit model from stepwise selection, we can see these predictors are used: Category\_mid, sellerRating, endDay\_Mon\_Tue\_Thu, Duration\_5, Category\_low.

f

lm.fit.step <-  
 glm(  
 `Competitive.` ~ Category\_mid + sellerRating + endDay\_Mon\_Tue\_Thu + Duration\_5 + Category\_low,  
 data = train,  
 family = "binomial"  
 )  
pred.valid <- predict(lm.fit.step, valid)  
confusionMatrix(as.factor(ifelse(pred.valid > 0.5, 1, 0)), as.factor(valid$`Competitive.`))

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 267 210  
## 1 109 203  
##   
## Accuracy : 0.5957   
## 95% CI : (0.5605, 0.6302)  
## No Information Rate : 0.5234   
## P-Value [Acc > NIR] : 2.644e-05   
##   
## Kappa : 0.1992   
##   
## Mcnemar's Test P-Value : 2.157e-08   
##   
## Sensitivity : 0.7101   
## Specificity : 0.4915   
## Pos Pred Value : 0.5597   
## Neg Pred Value : 0.6506   
## Prevalence : 0.4766   
## Detection Rate : 0.3384   
## Detection Prevalence : 0.6046   
## Balanced Accuracy : 0.6008   
##   
## 'Positive' Class : 0   
##

Predictive accuracy for Stepwise Selecion model is 0.5957 We are unable to do exhaustive search thus, we dont know accuracy for exhaustive search

g

# Overfit

h

hey are different because best-fitting model tries to get statitically significant variables as predictors, and the model will fit training data very well, but it might not work well on new data. However, best predictive model tries to lower the error rate on new data by evaluating the performance of the model on new data, not only just consider the predictor significance.

i

if(! require(ROCR)){  
 install.packages("ROCR")  
}

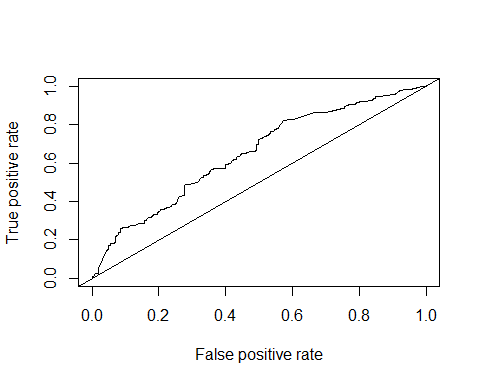
## Loading required package: ROCR

## Loading required package: gplots

##   
## Attaching package: 'gplots'

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

library(ROCR)  
pred <- prediction(pred.valid, valid$`Competitive.`)  
roc.perf <- performance(pred, measure = "tpr", x.measure = "fpr")  
plot(roc.perf)  
abline(a = 0, b = 1)



opt.cut = function(perf, pred) {  
 cut.ind = mapply(  
 FUN = function(x, y, p) {  
 d = (x - 0) ^ 2 + (y - 1) ^ 2  
 ind = which(d == min(d))  
 c(  
 sensitivity = y[[ind]],  
 specificity = 1 - x[[ind]],  
 cutoff = p[[ind]]  
 )  
 },  
 perf@x.values,  
 perf@y.values,  
 pred@cutoffs  
 )  
}  
print(opt.cut(roc.perf, pred))

## [,1]  
## sensitivity 0.5738499  
## specificity 0.6356383  
## cutoff 0.2519452

confusionMatrix(as.factor(ifelse(pred.valid > 0.2182772, 1, 0)), as.factor(valid$`Competitive.`))

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 221 165  
## 1 155 248  
##   
## Accuracy : 0.5944   
## 95% CI : (0.5592, 0.6289)  
## No Information Rate : 0.5234   
## P-Value [Acc > NIR] : 3.586e-05   
##   
## Kappa : 0.188   
##   
## Mcnemar's Test P-Value : 0.6149   
##   
## Sensitivity : 0.5878   
## Specificity : 0.6005   
## Pos Pred Value : 0.5725   
## Neg Pred Value : 0.6154   
## Prevalence : 0.4766   
## Detection Rate : 0.2801   
## Detection Prevalence : 0.4892   
## Balanced Accuracy : 0.5941   
##   
## 'Positive' Class : 0   
##

The accuracy increased from 0.5957 to 0.5944

j  
The parameters :currency, sellerRating, ClosePrice, OpenPrice, Category\_low, Category\_mid, endDay\_Mon\_Tue\_Thu, Duration\_5, Competitive. will lead to highest Competitiveness in auction setting. as we saw in b that when you fit model against all variables it gives highest competitiveness.