# Predicting the Competitiveness of Auctions using Classification Trees

Carrington Body

**Necessary R Packages:**

```
#install.packages("tidyverse")
#install.packages("dplyr")
#install.packages("caret")
#install.packages("rpart")
#install.packages("rpart.plot")
#install.packages("randomForest")
#install.packages("stargazer")
#install.packages("forecast")
#install.packages("ggplot2")
library(ggplot2)
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.2     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v lubridate 1.9.2     v tibble    3.2.1
## v purrr     1.0.1     v tidyr     1.3.0
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become error
```

```
library(dplyr)
library(caret)
```

```
## Loading required package: lattice
##
## Attaching package: 'caret'
##
## The following object is masked from 'package:purrr':
##
##     lift
```

```
library(rpart)
library(rpart.plot)
```

**Synopsis:** eBay.com is a commercial website where customers can purchase various items online. Each item has an auction, which is an event where at least one person places a bid on that item, in hopes of winning and receiving it. In this case, a competitive auction is defined as an auction with at least two bids placed on the item auctioned. Our goal, with the available data, is to accurately predict whether an auction is competitive or not. The file (eBayAuctions.csv) contains information on 1972 auctions that transacted on eBay.com during May-June 2004. The data include variables that describe the item (auction category), the seller rating (number of eBay ratings), and the auction terms that the seller selected (auction duration, opening price, currency). In addition, we have the day-of-week of auction close and the price at which the auction closed. Non-competitive auctions will be classified as 0, while competitive auctions will be classified as 1.

**Reading in the file for use** Here, we are examining the data and the types of each variable.

```
setwd("~/RDataMiningAnalytics")
eBay <- read.csv("eBayAuctions.csv", header = TRUE)
eBayAuctions <- as.data.frame(eBay)
View(eBayAuctions)
```

```
str(eBayAuctions)
```

```
## 'data.frame':    1972 obs. of  8 variables:
##  $ Category    : chr  "Music/Movie/Game" "Music/Movie/Game" "Music/Movie/Game" "Music/Movie/Game" ..
##  $ currency    : chr  "US" "US" "US" "US" ...
##  $ sellerRating: int  3249 3249 3249 3249 3249 3249 3249 3249 3249 3249 ...
##  $ Duration    : int  5 5 5 5 5 5 5 5 5 5 ...
##  $ endDay      : chr  "Mon" "Mon" "Mon" "Mon" ...
##  $ ClosePrice  : num  0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01 ...
##  $ OpenPrice   : num  0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01 ...
##  $ Competitive : int  0 0 0 0 0 0 0 0 0 0 ...
```

```
attach(eBayAuctions)
```

**Converting categorical variables** Now, we must keep in mind that Duration and Competitive are categorical because they are classifying a type of the attribute that represents each auction, instead of quantifying. Let's verify the types of both variables once we convert.

```
Duration <- as.factor(eBayAuctions$Duration)
Competitive <- as.factor(eBayAuctions$Competitive)
paste("Duration type:", class(Duration))
```

```
## [1] "Duration type: factor"
```

```
paste("Competitive type:", class(Competitive))
```

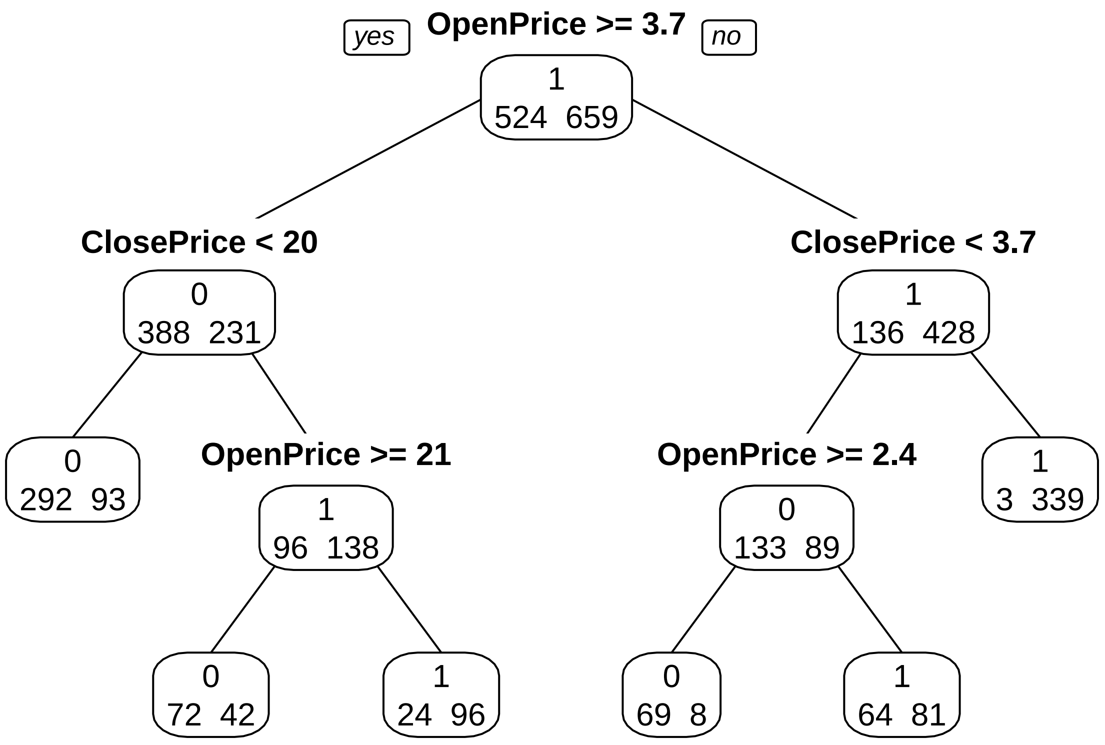
```
## [1] "Competitive type: factor"
```

**Partitioning the Data** We want to split the data in training and validation sets by a 60:40 ratio.

```
set.seed(1)
train.index <- sample(c(1:dim(eBayAuctions)[1]), 0.60*dim(eBayAuctions)[1])
valid.index <- setdiff(c(1:dim(eBayAuctions)[1]), train.index)
eBay_train.df <- eBayAuctions[train.index, ]
eBay_valid.df <- eBayAuctions[valid.index, ]
```

**Fitting a Classification Tree** Here, we'll fit our first tree using all variables. Using all variables from the start gives us a baseline for our model and the best chance to optimize it.

```r
class.tree <- rpart(Competitive ~ ., data = eBay_train.df,method = "class", minbucket = 50,
                    control = rpart.control(maxdepth = 3))
#set terminal node to 50 to reduce overfitting and easy interpretation
#set max depth to 3 so I could analyze more than just 2 levels of classification (too basic).
prp(class.tree, type = 1,  extra = 1, split.font = 2, varlen = 15)
```



**Rules of Full Classification Tree**

```r
class.tree
```

```
## n= 1183
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
##  1) root 1183 524 1 (0.44294167 0.55705833)
##    2) OpenPrice>=3.685 619 231 0 (0.62681745 0.37318255)
##      4) ClosePrice< 20.195 385  93 0 (0.75844156 0.24155844) *
##      5) ClosePrice>=20.195 234  96 1 (0.41025641 0.58974359)
##       10) OpenPrice>=20.5 114  42 0 (0.63157895 0.36842105) *
##       11) OpenPrice< 20.5 120  24 1 (0.20000000 0.80000000) *
##    3) OpenPrice< 3.685 564 136 1 (0.24113475 0.75886525)
##      6) ClosePrice< 3.685 222  89 0 (0.59909910 0.40090090)
##        12) OpenPrice>=2.445 77   8 0 (0.89610390 0.10389610) *
##        13) OpenPrice< 2.445 145  64 1 (0.44137931 0.55862069) *
##      7) ClosePrice>=3.685 342   3 1 (0.00877193 0.99122807) *
```

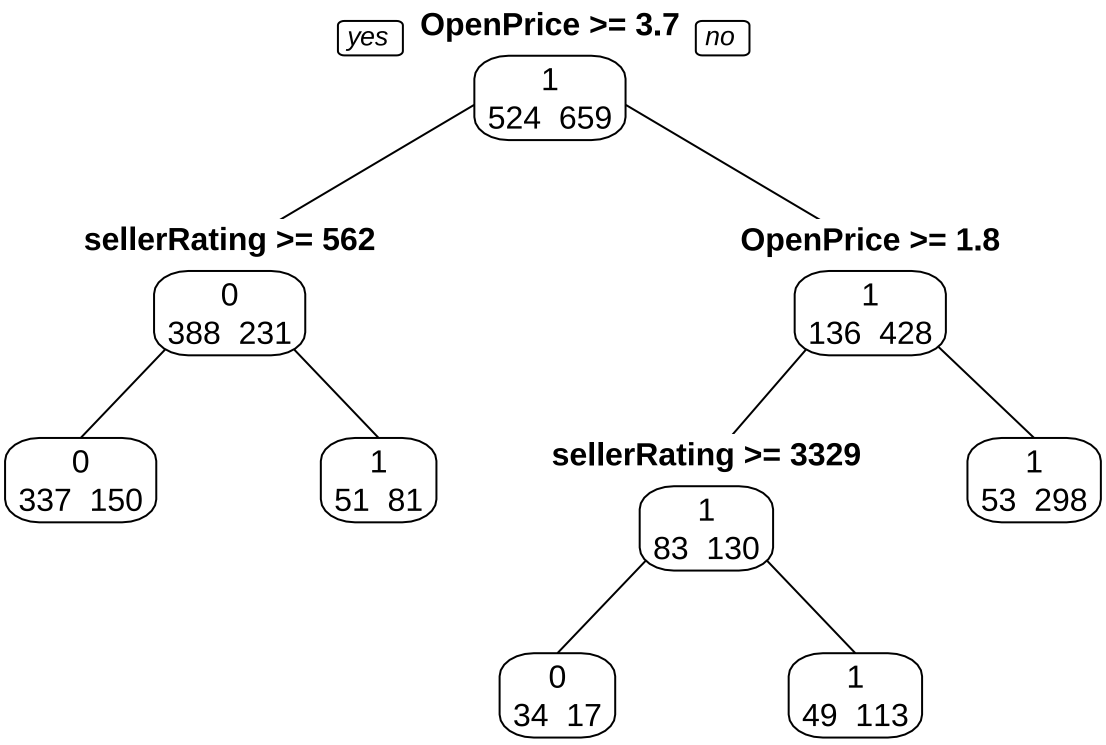
```
# If (OpenPrice >= 3.7) AND (ClosePrice < 20), THEN Class = 0 (noncompetitive).
# If (OpenPrice >= 3.7) AND (ClosePrice > 20), THEN Class = 1 (competitive).
# If (OpenPrice < 3.7) AND (ClosePrice > 3.7), THEN Class = 1 (competitive).
# If (OpenPrice < 3.7) AND (ClosePrice < 3.7), THEN Class = 0 (noncompetitive).
# For an auction to be competitive, if the item's opening price was less than $3.70, then then its
#closing price would have to be greater than $3.70, which makes sense because a higher bid
#(meaning at least two bids total) would've had to been placed for it to be classified as competitive.
#We can also see this instance with the other competitive rule in which if an item's opening price was
#at least $3.70 or better, then its closing price would have to be better than $20. If we go down to
#the lowest depth of the tree, we can see that the attribute opening prices was measured for less than
#$21 or more; if the opening price was less than $21, we can see that the auction would be considered
#competitive had its closing price been greater $20, otherwise, no. The two noncompetitive rules show
#that the opening and closing prices of items did not undergo much change, indicating that more than tw
#bids were not placed on these items.
```

### Practical model?

```
#This model is not practical because we want to use it to be able to predict the competitiveness of
#new auctions. With new auctions, we do not know the closing price, so this variable, at least, needs
#to be eliminated from the model. Since competitiveness is based on the number of bids placed, the
#most important variables would be opening price and sellerRating. The day in which the auction ended
#on should be not significant to competitiveness [but it could be in some cases], nor should how long
#the auction was open should be because an auction could close with just one bid on it [however
#an auction that is open for longer gives customers more time to make bids on it]. Neither the
#category of the item or the currency of the auction should be significant.
```

**Fitting New Tree** Here, we are fitting a new classification tree for the training set that only include the variables opening price and seller rating. Opening price is useful because it is an important auction term the seller selects. The other auction terms are duration and currency which should not matter. The seller rating also has some importance here because it gives us an idea of how popular they are on eBay.com and the outreach they have with auctions. A seller with more ratings may have more competitive auctions than those with less ratings.

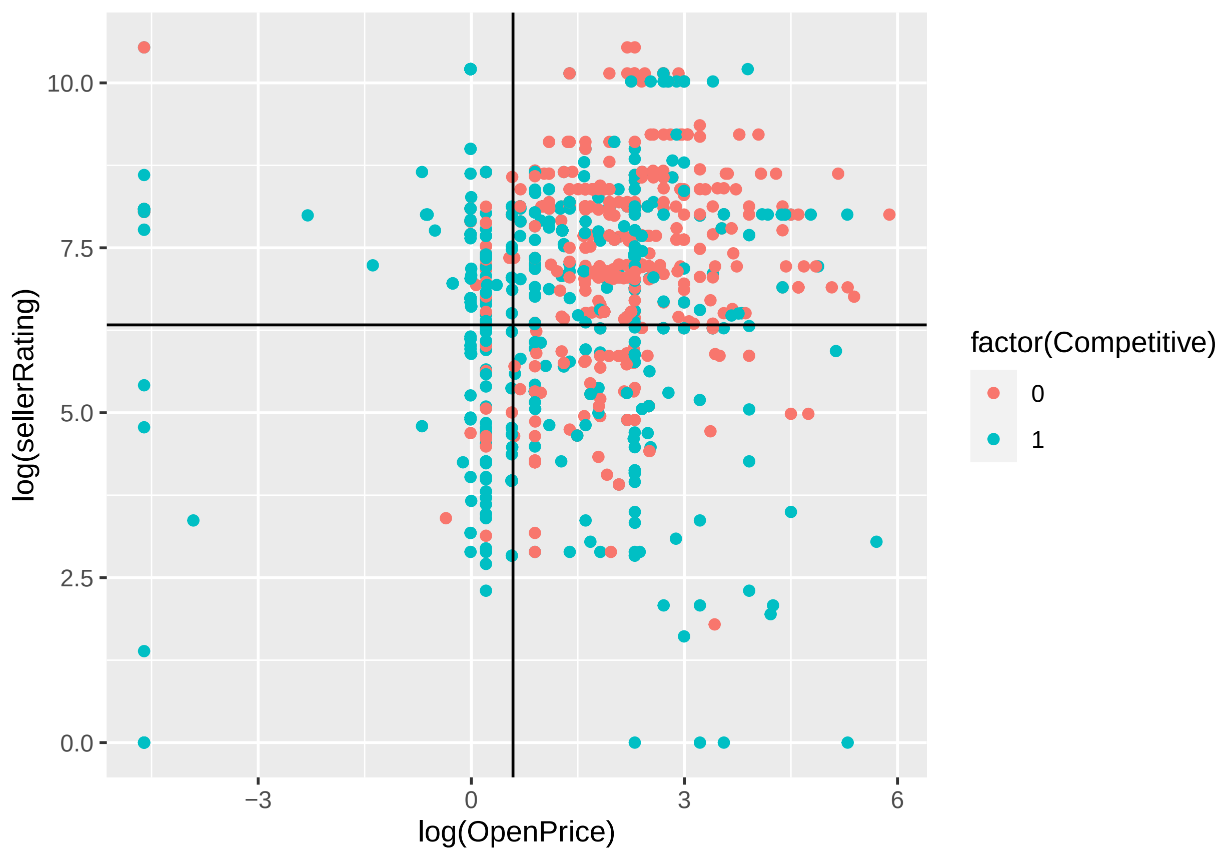
```
class.tree2 <- rpart(Competitive ~ sellerRating + Duration + endDay + OpenPrice, data = eBay_train.df,
                     method = "class", minbucket = 50, control = rpart.control(maxdepth = 3))
prp(class.tree2, type = 1,  extra = 1, split.font = 2, varlen = 15)
```



```
#Here, I included Duration and endDay just to see if our tree would be any different if we did not
#included include these two terms. The tree still selected Opening price and seller rating as the two
#most important variables.
```

**Tree Results w/ Scatterplot** Here, we are using a scatterplot to visualize the results of our classification tree. With the vertical and horizontal lines acting as boundaries, we can see the proper splitting of the classes (competitive and non-competitive) with respect to the two predictors (open price and seller rating). We can see that this tree is a much better model than the first one for predicting competitiveness for new auctions. We also did log transformations on the two quantitative predictors to standardize the data shown in the scatterplot.

```
ggplot(eBay_train.df, aes(x = log(OpenPrice), y = log(sellerRating))) +
  geom_point(aes(color = factor(Competitive))) + geom_hline(yintercept = log(562)) +
  geom_vline(xintercept = log(1.8))
```



```
#This splitting does seem reasonable with respect to the two predictors sellerRating and OpenPrice.
#It does seem to do a good job a separating the two classes, indicating that the second decision tree
#is a good model for a new auction.
```

**Performance on Validation Data** Here, we are testing our most recent classification tree on the validation data to see how good and accurate the model is, provided with a confusion matrix and an accuracy test.

```
predValid <- predict(class.tree2, eBay_valid.df, type = "class")
#for.matrix <- table(eBay_valid.df$Competitive, predValid)
#for.matrix
#accuracyTest <- sum(diag(for.matrix)) / sum(for.matrix)
#paste("% accuracy:", accuracyTest)
confusionMatrix(factor(predValid), factor(eBay_valid.df$Competitive))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 260 106
##          1 122 301
##
##                Accuracy : 0.711
##                  95% CI : (0.678, 0.7425)
##     No Information Rate : 0.5158
##     P-Value [Acc > NIR] : <2e-16
```

```
##
##                   Kappa : 0.4207
##
##  Mcnemar's Test P-Value : 0.3205
##
##             Sensitivity : 0.6806
##             Specificity : 0.7396
##          Pos Pred Value : 0.7104
##          Neg Pred Value : 0.7116
##              Prevalence : 0.4842
##          Detection Rate : 0.3295
##    Detection Prevalence : 0.4639
##       Balanced Accuracy : 0.7101
##
##        'Positive' Class : 0
##
```

```
#We can see that this model correctly predicted the competitiveness for roughly 71% of the auctions.
#Decent accuracy, but not great.
```

### 5-fold Cross-Validation

```
eBay_train.df$Competitive <- as.factor(eBay_train.df$Competitive)
tc <- trainControl(method = "repeatedcv", number = 5, repeats = 2)
#pruned.ct <- prune(class.tree2, cp = class.tree2$cptable[which.min(class.tree2$cptable[,"xerror"]),"CP
#prp(pruned.ct, type = 1,  extra = 1, split.font = 2, varlen = 15)
train_class.tree2 <- train(Competitive ~ sellerRating + Duration + endDay + OpenPrice,
                           data = eBay_train.df, method = "rpart", trControl = tc, maxdepth = 3,
                           minbucket = 50)
train_class.tree2
```

```
## CART
##
## 1183 samples
##    4 predictor
##    2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold, repeated 2 times)
## Summary of sample sizes: 946, 946, 946, 946, 948, 947, ...
## Resampling results across tuning parameters:
##
##   cp          Accuracy   Kappa
##   0.01622137  0.7070951  0.4103497
##   0.05725191  0.7003243  0.3977896
##   0.29961832  0.5916991  0.1050073
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was cp = 0.01622137.
```