Working Paper ∗ November 2014 ∗ Draft 1

An Analysis on Equity and Offers for Shark Tank

Steve Bronder

Fall 2014

Duquesne University Mathematics Department

Working Paper ∗ November 2014 ∗ Draft 1

Abstract

The television show Shark Tank is an American television series in which entreprenuers pitch their company and then ask "sharks", successfull entrepreneurs, for investment in exchange for an amount of equity. The purpose of this analysis is to build a rule based model to classif companies into "Offer" and "No offer" groups. The sharks investigated will be Mark Cuban, John Daymond, Robert Herjavec, and Kevin O'leary. If any one of them make an offer the pitch is considered a success.

1 Introduction

The television show Shark Tank is an American television series in which entreprenuers pitch their company and then ask "sharks", successfull entrepreneurs, for investment in exchange for an amount of equity. The purpose of this analysis is to build a rule based model to classify companies into "Offer" and "No offer" groups. The sharks investigated will be Mark Cuban, John Daymond, Robert Herjavec, and Kevin O'leary. If any one of them make an offer the pitch is considered a success.

This paper will give results of a set of rules created from a process called C5.0 in order to predict whether a pitch receives an offer or not. Variables used in this analysis include a company's evaluation, dollar amount asked for, equity asked for, number of months profitable, revenue, and profits. The rest of this paper will go as follows. Section two will give an overview of the data gathering and summary statistics. Section three will go into detail about the C5.0 model and the algorithm used for resampling. Section four will give the rules and predictions for the model.

2 Data

Data was gathered from the CSV file located on tvquotes.net[1] and loaded into R. Data exists for four seasons of Shark Tank with each observation being a pitch from an entrepreneur[2]. The dataset contains 29 different variables, but this paper only uses seven of them. We analyze a company's evaluation, dollar amount asked for, equity asked for, number of months profitable, revenue, and profits. Each observation is classified by whether or not a shark received an offer from any of the sharks accounted for in this analysis. Some sharks were left out due to long periods of not being on the show such as Jeff Foxworthy and Kevin Harrington. The data is preprocessed to re-

[1] http://www.sharktank.tvquotes.net/

[2] There are 235 total observations

move the mean and place all variables inside of one standard deviation. This is done in order to minimize erroneous rules due to large differences in variable means and errors

```
#Read in data from working directory
sharks <- read.csv("./sharkdata3.csv",header=TRUE,na.strings="NULL")
# There is a NA row for some reason
sharks <- sharks[1:235,]
# Create factor for Offer or no Offer
sharks$offer.inv <- as.factor(ifelse(sharks$Inv.offer==1,"Offer","No.Offer"))
# Reset who offered as factor
sharks$who.offer <- as.factor(sharks$who.offer)
# Create evaluation variable
sharks$evaluation <- sharks$ask_dollar/sharks$ask_equity
```

	ask_equity	profit_term_mos	revenue
1	Min. :-1.46913	Min. :-1.2179	Min. :-0.3749
2	1st Qu.:-0.85247	1st Qu.:-0.7907	1st Qu.:-0.3663
3	Median : 0.02849	Median : 0.4911	Median :-0.3093
4	Mean : 0.00000	Mean : 0.0000	Mean : 0.0000
5	3rd Qu.: 0.46897	3rd Qu.: 0.4911	3rd Qu.:-0.1490
6	Max. : 7.07614	Max. : 9.0364	Max. : 8.2305
7		NA's :61	NA's :42

	offer.inv	evaluation	ask_dollar
1	No.Offer:165	Min. :-0.59645	Min. :-0.6508
2	Offer : 70	1st Qu.:-0.47723	1st Qu.:-0.4877
3		Median :-0.30788	Median :-0.2757
4		Mean : 0.00000	Mean : 0.0000
5		3rd Qu.: 0.06469	3rd Qu.: 0.1320
6		Max. : 7.51598	Max. : 9.1012

Table 1: Summary Statistics Shark Tank Data

3 Model: C5.0

The model we use is known as C5.0, originally developed by Ross Quinlan[3] and integrated into R by Max Kuhn[4], is a statistical classifier based off of decision trees. At each node of the decision tree, C5.0 chooses the attribute of the data that most effectively splits its set of samples into subsets enriched in one class or the other. This is done through an information statistic that represents the average number of bits[5] required to code samples into two seperate splits. Let p be defiened as the probability of the first class being correct for an arbatrary, but particular set of observations.

$$info = -[p\ log_2 p + (1-p)\ log_2(1-p)] \tag{1}$$

Suppose p is the proportion of samples in the first class such that p = .53. From equation one, the average number of bits of information to guess the true class would be .997. This means we would need quite a lot of information in order to predict the class properly. Now suppose p=.10, then the information would be .46 bits, meaning given any random value in the samples it would be easier to randomly guess a class and be correct. Using this concept of information for splitting we choose the split which seperates the data into two sets which maximizes the information gain of the split. Let n_{i+} denote the number of observations with a particular class and n_{+i} the number of observations in a split with n being the total number of observations. Then;

[3] Quinlan, J. R. C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers, 1993.

[4] M. Kuhn and K. Johnson, Applied Predictive Modeling, Springer 2013

[5] This is just a unit of measure. It's taken from the number of real bits needed to compute the huffman algorithm.

$$gain(split) = info(\text{prior to split}) - info(\text{after split}) \tag{2}$$

$$info(\text{prior to split}) = -\left[\frac{n_{1+}}{n} \times log_2\left(\frac{n_{1+}}{n}\right)\right] - \left[\frac{n_{2+}}{n} \times log_2\left(\frac{n_{2+}}{n}\right)\right] \tag{3}$$

$$info(\text{after split}) = \frac{n_{2+}}{n} info(\text{greater}) + \frac{n_{2+}}{n} info(\text{less than}) \tag{4}$$

$$info(\text{greater}) = -\left[\frac{n_{11}}{n_{1+}} \times log_2\left(\frac{n_{11}}{n_{+1}}\right)\right] - \left[\frac{n_{12}}{n_{+1}} \times log_2\left(\frac{n_{12}}{n_{+1}}\right)\right] \tag{5}$$

Because our predictor values contain missing values we allow C5.0 to use several imputation methods. When calculating the information gain, the information statistics are calculated using the non-missing data then scaled by the fraction of non-missing data at the split. C5.0 also can treat missing values as an extra branch of the node. In addition, missing values are split fractionally on each node by the size of each class at the node.

Once the initial tree is grown, transformed and things such as boosting, pruning, predictor importance, winnowing, parallelizing, and blah blah are established we will talk about them here, however I am very tired.

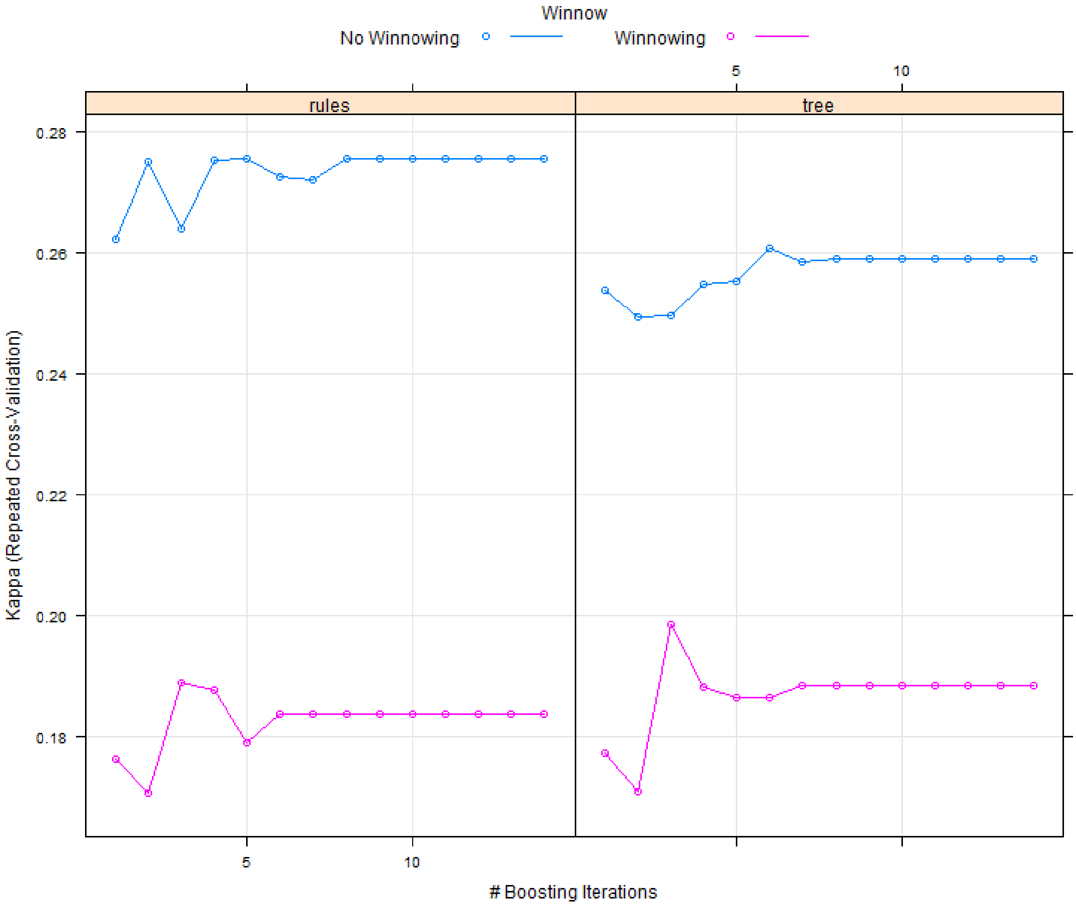
```
library(caret)

library(doParallel)
 cl <- makeCluster(3)

 registerDoParallel(cl)

c50Grid <- expand.grid(.trials = c(1:14),
                       .model = c("tree", "rules"),
                       .winnow = c(TRUE, FALSE))
```

```
ctrl <- trainControl(method = "repeatedcv",
             repeats = 10,
             returnResamp = "final",
             savePredictions = FALSE,
             classProbs = TRUE,
             summaryFunction = defaultSummary,
             selectionFunction = "best",
             preProcOptions = list(thresh = 0.95, ICAcomp = 3, k = 5),
             allowParallel = TRUE)



c5Fitvac <- train(offer.inv ~evaluation+ ask_dollar + ask_equity + profit_term_mos + reve
                  data = sharks,
                  method = "C5.0",
                  tuneGrid = c50Grid,
                  metric = "Kappa", # not needed it is so by default
                  trControl = ctrl,
                  importance=TRUE, # not needed
                  preProc = c("center", "scale"))
```

4 results

After running the data for our iterations.

Figure 1: Kappa from repeated cross validation and boosting grouped by winnowing.



```
if $ask\ equity \leq -1.035349$ then
    Offer
else
    if $Profitable\ Months \leq -0.9589436$ then
        return No Offer
    else
        if $Revenue > 0.5757602$ then
            return No Offer
        else
            if $Revenue \leq 0.2292188$ then
                return No Offer
            else
                return Offer
            end
        end
    end
end
```

	No Offer	Offer
No Offer	44.00	3.00
Offer	6.00	18.00

Table 2: Confusion Matrix for Shark Offers