The Use of Modern Statistical Methods to Predict the Successfulness of a Kickstarter

Joey Marques

Mentor: Dr. Jebessa Mijena

Department of Mathematics

Georgia College and State University 2018

Contents

Acknowledgements…… 3

Abstract…….. 4

Introduction……... 5

Methods…….. 6

Cleaning The Data……. 6

Multiple Linear Regression…….. 8

Logistic Regression…… 15

Linear Discriminant Analysis……... 19

Quadratic Discriminant Analysis……. 24

Tree-Based Models……. 27

Conclusion…… 31

Works Cited……. 32

Acknowledgements

I would like to thank Jebessa Mijena for mentoring me throughout my research and preparing me for the expectations of the job field I hope to enter. I would also like to thank my family for their constant support throughout this semester.

Abstract

In this research, we analyze Kickstarter data from Kaggle in hopes of being able to predict how successful a project would be. Our goal was to predict how much money a Kickstarter would raise, the state of a Kickstarter (successful or failed) and the ratio of the amount pledged vs the goal wanted, based off of the amount of days a project was active, the category, the amount of backers, and the country of origin. The modern statistical methods used include linear regression, logistic regression, quadratic discriminate analysis, linear discriminate analysis, and tree-based models.

Introduction

Kickstarter is an online crowdfunding website that allows new businesses to present their projects to the world and acquire the necessary funding they need to bring them to life. The motivation behind this project is to see if it is at all possible to predict how successful on project will be. This research uses modern statistical methods such as Linear Regression, Logistic Regression, Linear Discriminant Analysis, Quadratic Discriminant Analysis, and Tree-Based models to see if there is a correlation between the amount of backers, the country of origin, how many days a project was active, and the category of a project with the amount of money earned, the pledged/goal ratio, and the overall state of successful or failed. The data analysis was all conducted in Rstudio and cleaned in part in excel. The data of Kickstarter projects came from Kaggle which is a website that contains mounds of data on a variety of different topics.

Methods

Cleaning The Data

To start this project off I downloaded the data from Kaggle on 8/16/2018. The original data consisted of 13 variables: ID, Name, Category, Main Category, Currency, Deadline, Goal, Launched, Pledged, State, Backers, Country, and USD Pledged. The original file also contained 323,571 rows of data. Before one can accurately analyze data, the data must be cleaned. This entails removing any rows of data that contain missing data points and removing bad data points as well.

There is a function in Rstudio called "na.omit," this runs through my entire data set and removes any rows with missing data. Next, I examined my data and decided what variables would be useful and not useful in predicting how successful a project would be. The variables I removed were "ID," "Main Category," and "Currency". I also deleted the variables "Launched" and "Deadline" and replaced them with "Days". Then added a "pledged goal ratio" variable as well. There is an Excel formula that can calculate the difference of days between two dates, along with one that can find the dividend between two values in different columns.

After that I used excel to analyze bad data points. I made an excel table out of my data and then made a pivot table to see the responses to each variable. For instance, in the category variable there was a response that said "6", or in the "USD Pledged" variable there was a "yes" recorded, so these needed to be removed. In my analysis I was only interested in projects that were "failed" and "successful". The other responses included were "live," "cancelled," and "suspended". Because these three last options are up to the discretion of the owner of the Kickstarter there was no way to predict these states, so they were removed as well. In the country variable one country Mexico was also removed. In our entire dataset we started with 323,571 rows of data, and after removing the rows with empty data points there were a total of four Kickstarter projects out of the cleaned data (281,092 Kickstarter projects), so these were removed as well.

Now we were ready to bring the data into Rstudio and start analyzing it. We loaded the data into R and split it into test and training sets. The training data consisted of two thirds of our entire data (187,392 projects), while the remaining one third was saved for the test data (93,696 projects). The training data is used to make the models with each method used, and then the test data is run through the model. We then compared the predictions made with the model to the actual results

of the test data to see how accurate the model was. For training data, one typically wants to use between sixty to eighty percent of the total data while the test data will compromise of the remaining unused data. We chose to use two thirds of the data because we have so much that we did not want to risk overfitting the model.

The various methods used to analyze the data are available through different packages in R. All of these needed to be downloaded accordingly. We analyzed this data using the following methods: Linear Regression, Logistic Regression, Linear Discriminant Analysis, Quadratic Discriminant Analysis, and Tree Based Methods.

Multiple Linear Regression

Linear Regression is the modern statistical method that assumes that a quantitative response *Y* is related to predictors $x_0, x_1, x_2, \ldots x_p$ in a linear fashion. Starting with Simple Linear Regression (trying to predict one response with one predictor) is the best way to fully explain how this process works. Mathematically this relationship can be written as:

$$Y \approx \beta_0 + \beta_1 X$$

Where *Y* is our response variable, β_0 is the y-intercept and β_1 is the slope term for the predictor *X*. The training data is used to create a model which will calculate the y-intercept and slope term that will be later used in the prediction formula:

$$\hat{y} = \widehat{\beta_0} + \widehat{\beta_1}$$

The hat symbol (^) indicates that the value has been estimated/ calculated in a way that could hold some error to its true value.

To estimate these coefficients, observe the figure below which contains one predictor and one response variable.

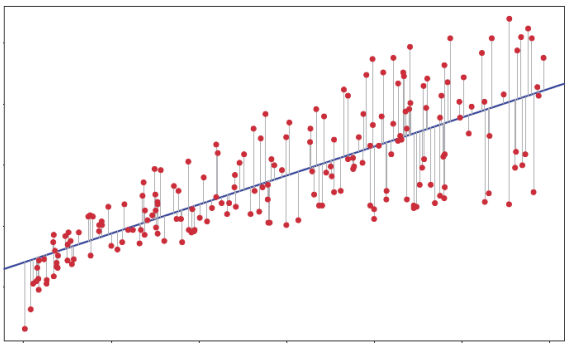


Figure 1

The goal of Linear Regression is to minimize the distance (gray lines) between the predicted values (the solid blue line) and the actual values (red points) as much as possible. This approach is called minimizing the least squares.

Mathematically speaking let us define a residual e_i as the difference between the actual and predicted value of the response. So, we can define this new value (RSS) the residual sum of squares as:

$$RSS = {e_1}^2 + {e_2}^2 + \cdots + {e_n}^2$$

Which is broken down to:

$$RSS = (y_1 - \widehat{\beta_0} - \widehat{\beta_1} x_1)^2 + (y_2 - \widehat{\beta_0} - \widehat{\beta_1} x_2)^2 + (y_1 - \widehat{\beta_0} - \widehat{\beta_1} x_n)^2$$

Where the intercept and slope terms are equal to:

$$\widehat{\beta_0} = \hat{y} - \widehat{\beta_1}\bar{x}$$

$$\widehat{\beta_1} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

Where x bar and y bar are the sample means.

Our goal is to minimize this RSS value. In our research we had more than one predictor, we had 4 predictors, 2 quantitative and 2 qualitative. Say we were to have two predictors in our model, then our graphical representation would look like the graph below:

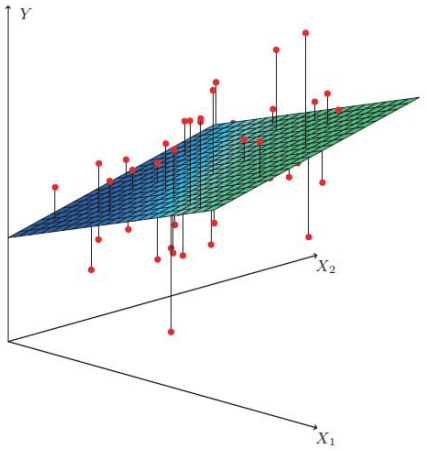


Figure 2

With two predictors we now have a 3-dimensional space. As the numbers of predictors increase so will our space by a factor of n+1. The math is overall the same, but it is important to note that it would be in matrix notation as opposed to a simple (x,y) point in a 2-dimensional space.

The following table is our model made with the training data:

```
## Call:
## lm(formula = usd.pledged ~ m.category + Days + country + backers,
##     data = train.Data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -9522963    -2342     -921      389  8672511
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)          -5.315e+02  4.083e+03  -0.130  0.89643
## m.categoryComics     -2.696e+03  9.037e+02  -2.984  0.00285 **
## m.categoryCrafts     -6.474e+02  9.837e+02  -0.658  0.51047
## m.categoryDance       1.341e+02  1.315e+03   0.102  0.91878
## m.categoryDesign      5.133e+03  6.722e+02   7.636 2.25e-14 ***
## m.categoryFashion     1.070e+03  7.163e+02   1.494  0.13509
## m.categoryFilm.Video  1.286e+03  5.558e+02   2.314  0.02065 *
## m.categoryFood        1.013e+03  6.816e+02   1.486  0.13727
## m.categoryGames      -1.452e+03  6.554e+02  -2.215  0.02673 *
## m.categoryJournalism -3.281e+02  1.244e+03  -0.264  0.79198
## m.categoryMusic      -2.912e+02  5.753e+02  -0.506  0.61276
## m.categoryPhoto       2.717e+02  8.726e+02   0.311  0.75551
## m.categoryPublishing -9.314e+02  6.094e+02  -1.528  0.12644
## m.categoryTech        9.120e+03  6.631e+02  13.752  < 2e-16 ***
## m.categoryTheater     3.553e+02  8.474e+02   0.419  0.67503
## Days                  4.397e+01  1.009e+01   4.356 1.33e-05 ***
## countryAU            -3.250e+03  4.152e+03  -0.783  0.43372
## countryBE            -1.382e+03  5.584e+03  -0.247  0.80454
## countryCA            -1.702e+03  4.099e+03  -0.415  0.67793
## countryCH            -1.436e+02  5.310e+03  -0.027  0.97842
## countryDE            -9.326e+02  4.306e+03  -0.217  0.82855
## countryDK            -1.761e+03  4.820e+03  -0.365  0.71485
## countryES            -3.077e+03  4.561e+03  -0.675  0.49984
## countryFR            -7.719e+02  4.410e+03  -0.175  0.86106
## countryGB            -5.280e+02  4.066e+03  -0.130  0.89667
## countryHK            -6.163e+03  1.069e+04  -0.577  0.56425
## countryIE            -2.045e+03  5.136e+03  -0.398  0.69052
## countryIT            -1.408e+03  4.436e+03  -0.317  0.75091
## countryLU            -2.400e+02  1.235e+04  -0.019  0.98450
## countryNL            -1.990e+03  4.333e+03  -0.459  0.64610
## countryNO            -2.437e+03  5.246e+03  -0.464  0.64230
## countryNZ            -2.515e+03  4.613e+03  -0.545  0.58564
## countrySE            -2.838e+03  4.566e+03  -0.622  0.53418
## countrySG            -1.039e+04  9.035e+03  -1.150  0.25018
## countryUS            -2.112e+02  4.044e+03  -0.052  0.95834
## backers               6.151e+01  1.435e-01 428.689  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 55980 on 187356 degrees of freedom
## Multiple R-squared:  0.4998, Adjusted R-squared:  0.4997
## F-statistic:  5349 on 35 and 187356 DF,  p-value: < 2.2e-16
```

Table 1

Before I get into which values are significant and what they mean, I would like to explain how the qualitative variables are interpreted in the table mentioned on the previous page. There is one level (subset of category and country) missing from the table. That category would be Art, and country Austria. That is because each category and country is compared to a base value. For example, we would interpret the design category by saying "on average a design Kickstarter would make five thousand one hundred thirteen dollars more than art." In the country predictor we would interpret US by saying "on average a project originating in the United States would make two hundred eleven dollars less than a Kickstarter originating from Austria."

The way R interprets a qualitative predictor with multiple levels is by creating dummy variables. What this means is that the slope for the predictor is listed in this table, but the specific predictor that is involved with a specified project is going to be equal to one, while the rest will be equal to zero. So, if we are looking at a Design Kickstarter from the United States the rest of the predictors in the category predictor will be equal to zero (multiplied by their respective coefficients) as will the other countries involved. This ensures that only the specified levels of the qualitative predictors affect the outcome.

Our equation will thus have the form of

(specified category)(1)+(unspecified category)(0)+(unspecified category_2)(0)+(unspecified category_3)(0)+…+(Days)(43.97)+(specified country)(1)+(unspecified country_2)(0)+(unspecified country_3)(0)+…+(backers)(61.51)=USD Pledged

The column Pr(>|t|) is known as the p-value. The rule with this value is that the smaller it is the more correlation there is between this predictor and the given response variable. R makes this relatively simple for us to point out, the predictors with more correlation with the response are indicated with ***, with the more asterisks (3 being the most) the more significant the p-value is and the more correlation between the predictor and the response there is.

After the table is the Residual error. This value is the average amount that the model was off per our training data. This value is 55,980. Meaning that our model was off by that average per prediction. While this is not great note that we are analyzing a huge amount of data, and that our data set for USD Pledged ranges from $0-$20,338,986 across hundreds of thousands of projects.

Next, we see the R-squared value. This is the most important value in determining whether there is an overall correlation between the predictors and the response. R^2 measures the proportion of variability in our response that can be explained by our predictors. The closer to one the more closely correlated the two are. In our model we have an R^2 value of .4998, which is not great, but not terrible at the same time.

To asses the model accuracy we ran our test data through the model and then calculated the Root Mean Square Error. The Root Mean Square Error is calculated by using the formula:

$$RMSE = \frac{1}{n}\sqrt{\sum_{j=1}^{n}(y_j - \hat{y}_j)^2}$$

This basically gives us the Residual Error compared to our test data. When we calculated this for our model we got the result $73,124. Meaning that when we ran our test data through the model and compared what our model predicted vs what the actual results were we were off by an average of $73,124.

Our next model was trying to predict the pledged goal ratio for a Kickstarter. The following table represents our model based off our training data:

```
## Call:
## lm(formula = Ratio.pledged.goal ~ m.category + Days + country + 
##     backers, data = train.Data)
## 
## Residuals:
##    Min     1Q Median     3Q    Max 
##   -569     -5     -2     -1 104244 
## 
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)    
## (Intercept)           -0.5290563 23.8603041  -0.022    0.982    
## m.categoryComics       4.1238533  5.2807778   0.781    0.435    
## m.categoryCrafts      -2.1189841  5.7488002  -0.369    0.712    
## m.categoryDance       -2.8935573  7.6862013  -0.376    0.707    
## m.categoryDesign      -2.7638033  3.9281111  -0.704    0.482    
## m.categoryFashion     -2.8869540  4.1856399  -0.690    0.490    
## m.categoryFilm.Video  -2.1133435  3.2481211  -0.651    0.515    
## m.categoryFood        -2.6631693  3.9830351  -0.669    0.504    
## m.categoryGames        4.7900243  3.8297787   1.251    0.211    
## m.categoryJournalism  -3.1173721  7.2699984  -0.429    0.668    
## m.categoryMusic        4.5452201  3.3616785   1.352    0.176    
## m.categoryPhoto       -2.8664516  5.0994798  -0.562    0.574    
## m.categoryPublishing  -1.6475884  3.5614330  -0.463    0.644    
## m.categoryTech        -0.0999388  3.8751756  -0.026    0.979    
## m.categoryTheater     -2.6296766  4.9521579  -0.531    0.595    
## Days                   0.0339856  0.0589893   0.576    0.565    
## countryAU              0.4193199 24.2626127   0.017    0.986    
## countryBE             -0.5286858 32.6322251  -0.016    0.987    
## countryCA              1.2906414 23.9547501   0.054    0.957    
## countryCH              0.3890015 31.0313341   0.013    0.990    
## countryDE             -0.2640480 25.1661070  -0.010    0.992    
## countryDK              0.1390448 28.1657988   0.005    0.996    
## countryES             -0.4818350 26.6512819  -0.018    0.986    
## countryFR              2.2706935 25.7709769   0.088    0.930    
## countryGB              0.4066663 23.7608850   0.017    0.986    
## countryHK              0.4197041 62.4713417   0.007    0.995    
## countryIE              0.1919606 30.0114211   0.006    0.995    
## countryIT              0.0413981 25.9204972   0.002    0.999    
## countryLU              0.7954299 72.1906692   0.011    0.991    
## countryNL             -0.0805972 25.3234279  -0.003    0.997    
## countryNO             -0.4731553 30.6569152  -0.015    0.988    
## countryNZ              1.0595064 26.9581929   0.039    0.969    
## countrySE             -0.1829951 26.6826738  -0.007    0.995    
## countrySG              1.2604557 52.7996310   0.024    0.981    
## countryUS              3.2551765 23.6333429   0.138    0.890    
## backers                0.0062281  0.0008385   7.427 1.11e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 327.2 on 187356 degrees of freedom
## Multiple R-squared:  0.000407,   Adjusted R-squared:  0.0002202 
## F-statistic: 2.179 on 35 and 187356 DF,  p-value: 6.762e-05
```

Table 2

The model on page thirteen is exactly what one does not want to see with a model. We can see that none of the predictors are significant in predicting the response (aside from backers), and that our Residual Standard Error states that our model was off by an average of 327.2 per actual value compared to our training data. Then we also see that our R^2 coefficient is .000407, which is very close to zero. This tells us that our response is not linearly correlated to our predictors at all (this value comes out to a 0% correlation). When we ran our test data through our RMSE came out to 80.94, meaning that the ratios we predicted were off by an average of 80.94. While this model was not good in predicting this ratio, it does not mean one cannot not predict this. It just means that this model with these predictors are not related to this specific response.

Logistic Regression

We use this method if the response we are trying to predict is a qualitative one (something non-numeric). This process is known as classification. Because we are only looking at two types of responses "successful" or "failed" R will assign a dummy variable to each one. In our case this was zero for failed, and one for successful. A probability by definition will always fall between zero and one, so our outputs will follow the graph below:

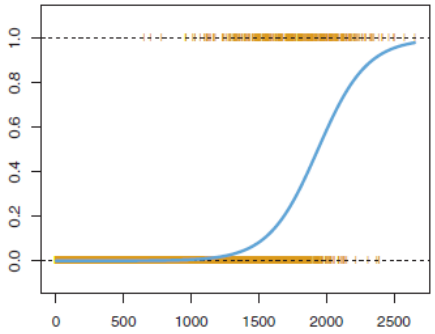


Figure 3

Our model rounds any probability less than .5 as failed, and any probability greater than .5 as successful. To obtain a model that gives us a probability, a logistic function will be used as seen below:

$$p(X) = \frac{e^{\beta_0+\beta_1 X_1+\cdots+\beta_p X_p}}{1+e^{\beta_0+\beta_1 X_1+\cdots+\beta_p X_p}}$$

The coefficients are unknown here and must be estimated based on the available training data. In Linear Regression the least squares approach was used, but we cannot use that here, instead we use the maximum likelihood method. This can be formalized by the likelihood equation below:

$$l(\beta_0,\beta_1) = \prod_{i:y_i=1} p(x_i) \prod_{i':y'=0} (1-p\left(x_{i'}\right))$$

The mathematical details of the workings of this formula are beyond the scope of my research, but if you would like to read more on the workings of this formula please read *An Introduction to Statistical Learning with Applications in R* pg 133. Note that this formula is with one predictor so in our data analysis this would be worked out with multiple predictors.

The model my training data produced is given in the table below:

```
## Call:
## glm(formula = state ~ m.category + Days + country + backers,
##     family = binomial, data = KickstarterClean, subset = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -8.4904  -0.7070  -0.4263   0.6544   2.8364
##
## Coefficients:
##                          Estimate Std. Error z value Pr(>|z|)
## (Intercept)           -0.8143726  0.2328050  -3.498 0.000469 ***
## m.categoryComics      -0.1740027  0.0400821  -4.341 1.42e-05 ***
## m.categoryCrafts      -0.6850397  0.0440865 -15.539  < 2e-16 ***
## m.categoryDance        0.7546421  0.0538923  14.003  < 2e-16 ***
## m.categoryDesign      -1.3680448  0.0330410 -41.404  < 2e-16 ***
## m.categoryFashion     -1.0010186  0.0331216 -30.222  < 2e-16 ***
## m.categoryFilm.Video  -0.2747553  0.0229336 -11.980  < 2e-16 ***
## m.categoryFood        -1.1263002  0.0319332 -35.271  < 2e-16 ***
## m.categoryGames       -1.6049642  0.0342491 -46.861  < 2e-16 ***
## m.categoryJournalism  -0.9012035  0.0609378 -14.789  < 2e-16 ***
## m.categoryMusic        0.2077883  0.0234526   8.860  < 2e-16 ***
## m.categoryPhoto       -0.4742596  0.0375507 -12.630  < 2e-16 ***
## m.categoryPublishing  -0.6030615  0.0259781 -23.214  < 2e-16 ***
## m.categoryTech        -1.9030353  0.0374588 -50.803  < 2e-16 ***
## m.categoryTheater      0.6896440  0.0345252  19.975  < 2e-16 ***
## Days                  -0.0245194  0.0004910 -49.935  < 2e-16 ***
## countryAU              0.3332681  0.2366995   1.408 0.159137
## countryBE              0.3822185  0.3068922   1.245 0.212967
## countryCA              0.4199406  0.2340031   1.795 0.072718 .
## countryCH              0.1614136  0.3027250   0.533 0.593894
## countryDE              0.1299372  0.2458547   0.529 0.597144
## countryDK              0.7164166  0.2636372   2.717 0.006579 **
## countryES             -0.0657433  0.2641046  -0.249 0.803416
## countryFR              0.4473165  0.2493259   1.794 0.072797 .
## countryGB              0.6667946  0.2323752   2.869 0.004112 **
## countryHK              0.5846031  0.5925717   0.987 0.323862
## countryIE              0.0371409  0.2888445   0.129 0.897687
## countryIT             -0.2095115  0.2593494  -0.808 0.419186
## countryLU              0.9014053  0.5510669   1.636 0.101893
## countryNL              0.0270415  0.2483209   0.109 0.913284
## countryNO              0.2131899  0.2922043   0.730 0.465640
## countryNZ              0.5729537  0.2548980   2.248 0.024590 *
## countrySE              0.4426489  0.2540952   1.742 0.081498 .
## countrySG              1.4743823  0.4507369   3.271 0.001071 **
## countryUS              0.6317181  0.2315746   2.728 0.006373 **
## backers                0.0270727  0.0001677 161.389  < 2e-16 ***
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Table 4

The table on the prior page generally works the same as the Linear Regression. Here we can see the category of a Kickstarter greatly affects whether one succeeds or not (remember this is compared to Art), along with days, and a few specific countries (again compared to Austria), and the amount of backers. Surprisingly there are only a select few categories that are more likely to succeed when compared to art, and the table also shows that the longer a project is active the less likely it is to succeed.

To analyze the accuracy of this model we compare how many projects the model accurately predicted as failed and as successful compared to their actual results. There is a package in R called "carret" which builds a confusion matrix and computes the specificity (the rate of correctly predicting failed projects when they actually failed otherwise known as the true negative rate) and the sensitivity (the rate of correctly predicting successful projects when they were actually successful otherwise known as the true positive rate).

Our model accuracy is shown in the table below:

```
StatelRegressionPred     0     1
                   0 52474 12792
                   1  3545 24885

               Accuracy : 0.8256
                 95% CI : (0.8232, 0.8281)
    No Information Rate : 0.5979
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.6222
 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.6605
            Specificity : 0.9367
         Pos Pred Value : 0.8753
         Neg Pred Value : 0.8040
             Prevalence : 0.4021
         Detection Rate : 0.2656
   Detection Prevalence : 0.3034
      Balanced Accuracy : 0.7986

       'Positive' Class : 1
```

Table 5

The results on the previous page shows us that this model was overall 82.56% accurate. The model was 66.05% accurate when predicting true successful Kickstarters and 93.67% accurate in predicting true failing Kickstarters. To read the confusion matrix start at the far left of the table. That side is labeled "StatelRegressionPred" the following value in the first column first row is the true negative rate (where the zeroes intercept) and incorrectly predicted failing projects in the second row of the first column. In the second column first row we can see where the model incorrectly predicted failed projects when they were in reality successful, and in the second row of the second column we can see where the model correctly predicted the successful projects (true positive rate).

Linear Discriminant Analysis

Logistical Regression involves predicting the probability of a response given the predictors. This method involves working backwards (meaning given the response what are the predictors) and then using Bayes' Theorem to flip this back to the normal formula. In other words:

$$\Pr(Y = k|X = x) = \frac{\pi_k f_k(x)}{\sum_{l=1}^{K} \pi_l f_l(x)}$$

Where π_k is equal to the prior probability of the group, and where $f_k(X)=\Pr(X=x|Y=k)$ is known as the density function of X for an observation that comes from the kth class. $f_k(x)$ is large if the probability that an observation in the kth class has X=x, and small if it is unlikely that the kth class has X=x.

We refer to $p_k(x)$ as the posterior probability that an observation X=x belongs to the kth class. Meaning the above equation is equal to $p_k(x)$, that is the probability that an observation belongs to kth class, given the predictor value of that observation.

Suppose we assume $f_k(x)$ is normally distributed, then $f_k(x)$ can be written as:

$$f_k(x) = \frac{1}{\sqrt{2\pi\sigma_k}} \exp(-\frac{1}{2\sigma_k^2}(x - \mu_k)^2)$$

So $p_k(x)$ is now equal to:

$$p_k(x) = \frac{\pi_k \frac{1}{\sqrt{2\pi\sigma_k}} \exp(-\frac{1}{2\sigma_k^2}(x - \mu_k)^2)}{\sum_{l=1}^{k} \pi_l \frac{1}{\sqrt{2\pi\sigma_k}} \exp(-\frac{1}{2\sigma_k^2}(x - \mu_l)^2)}$$

If we take the log of both sides, we acquire the discriminant equation:

$$\delta_k(x) = x * \frac{\mu_k}{\sigma^2} - \frac{\mu_k^2}{2\sigma^2} + \log(\pi_k)$$

Which is the linear discriminant function, note that this is the case with one predictor and in our research we have four predictors, so the equation would be working with multiple predictors instead. The linear discriminant coefficients that R calculates are derived from what our predictor (x) is multiplied by.

Overall R computes the linear discriminant coefficients based of the prior probabilities and mean of each predictor in each response group through the maximum likelihood function. Then the equation that R uses to score each observation is made up of:

(linear discriminant$_1$)(X_1)+(linear discriminant$_2$)(X_2)+…(linear discriminant)(X_p)

Note that the qualitative predictors will still use dummy variables 0 and 1 accordingly in each prediction.

R calculates a decision boundary to place each predictor based off what scores are produced and will classify these predictions into one of two classes (being successful and failed in our case). In a graphical sense R chooses a line to split the successful and failed projects. All points to the left of this line will be failed, while to the right of this line are successful, an example is shown below. If you would like to read more about how this method works, please read *An Introduction to Statistical Learning with Applications in R* pg 138.

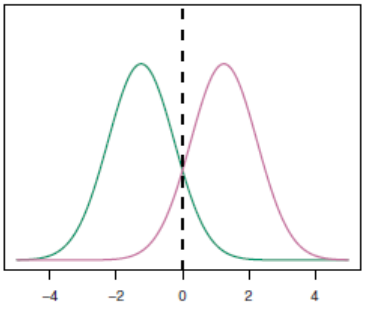


Figure 4

Below is our model produced from our training data:

```
## Call:
## lda(state ~ m.category + Days + country + backers, data =
KickstarterClean,
##     subset = train)
##
## Prior probabilities of groups:
##     failed successful
##  0.5981846  0.4018154
##
## Group means:
##            m.categoryComics m.categoryCrafts m.categoryDance
## failed           0.01999197       0.02753022     0.006316071
## successful       0.03956333       0.01436976     0.018088370
##            m.categoryDesign m.categoryFashion m.categoryFilm.Video
## failed           0.07177840        0.06852224            0.1744681
## successful       0.07079963        0.03814229            0.1890779
##            m.categoryFood m.categoryGames m.categoryJournalism
## failed         0.08100272      0.07765734          0.015888309
## successful     0.04673493      0.08194218          0.007623146
##            m.categoryMusic m.categoryPhoto m.categoryPublishing
## failed           0.1151345      0.03406932           0.11844418
## successful       0.1933012      0.02561855           0.09090668
##            m.categoryTech m.categoryTheater     Days  countryAU
## failed         0.09700700        0.02004550 35.38152 0.02228467
## successful     0.04484906        0.05314953 32.42218 0.01255030
##               countryBE  countryCA   countryCH   countryDE   countryDK
## failed     0.0014362817 0.04084928 0.001837727 0.009420581 0.002658459
## successful 0.0006640371 0.02694662 0.000770283 0.004727944 0.002071796
##              countryES   countryFR  countryGB    countryHK   countryIE
## failed     0.004933315 0.006101967 0.08583791 0.0002141041 0.001980463
## successful 0.001965550 0.004249837 0.08227419 0.0001062459 0.001195267
##              countryIT    countryLU   countryNL    countryNO   countryNZ
## failed     0.006985147 0.0001248941 0.008947768 0.0019626210 0.003693296
## successful 0.002031953 0.0001195267 0.003652204 0.0007968445 0.002908482
##              countrySE    countrySG countryUS   backers
## failed     0.004282082 0.0001873411 0.7949239  16.93669
## successful 0.002842079 0.0003585800 0.8492105 250.78255
```

```
## Coefficients of linear discriminants:
##                                  LD1
## m.categoryComics       0.9320593536
## m.categoryCrafts      -1.3789318768
## m.categoryDance        1.5588719064
## m.categoryDesign      -0.3714154100
## m.categoryFashion     -1.2538848871
## m.categoryFilm.Video  -0.1263875310
## m.categoryFood        -1.1877880795
## m.categoryGames       -0.3480441565
## m.categoryJournalism  -1.3904538367
## m.categoryMusic        0.6889354811
## m.categoryPhoto       -0.7466169958
## m.categoryPublishing  -0.7517413300
## m.categoryTech        -1.4356504171
## m.categoryTheater      1.4467784098
## Days                  -0.0333303526
## countryAU              0.2678789311
## countryBE              0.0379695356
## countryCA              0.4470794486
## countryCH              0.0318589170
## countryDE              0.1497192881
## countryDK              0.7899078614
## countryES             -0.1129538521
## countryFR              0.7166711637
## countryGB              0.8485183073
## countryHK              0.1408399692
## countryIE              0.3460383192
## countryIT             -0.3107594500
## countryLU              1.1608543635
## countryNL             -0.0627071854
## countryNO             -0.1823788542
## countryNZ              0.7221289817
## countrySE              0.4011508172
## countrySG              2.3695131853
## countryUS              1.0377160643
## backers                0.0005176701
```

Below are our results produced when we ran the test data through the model:

```
lda.class    failed successful
  failed      48500      25572
  successful   7519      12105

               Accuracy : 0.6468
                 95% CI : (0.6438, 0.6499)
    No Information Rate : 0.5979
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.203
 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.3213
            Specificity : 0.8658
         Pos Pred Value : 0.6168
         Neg Pred Value : 0.6548
             Prevalence : 0.4021
         Detection Rate : 0.1292
   Detection Prevalence : 0.2094
      Balanced Accuracy : 0.5935

       'Positive' Class : successful
```

Overall this model was 64.68% accurate. It had a true negative rate (specificity) of 86.58% and a true positive rate (sensitivity) of 32.13%.

Quadratic Discriminant Analysis

Quadratic Discriminant Analysis is related to LDA, except it assumes a quadratic relationship between the predictors and the response. QDA assigns an observation to a response class with an equation similar to LDA, but a bit more complicated, if you would like to read how this equation works please read *An Introduction to Statistical Learning with Applications in R* pg 149. The main difference between this model and LDA is that the decision boundary is a quadratic line as opposed to linear as seen in the example below:

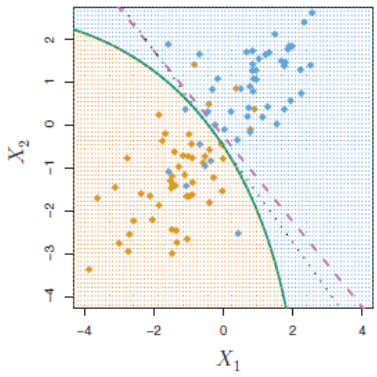


Figure 5

R assigns an observation to the class for which the above equation is the largest.

Below is our model acquired from the training data:

```
## Call:
## qda(state ~ m.category + Days + country + backers, data = KickstarterClean
,
##     subset = train)
##
## Prior probabilities of groups:
##     failed successful
##  0.5981846  0.4018154
##
## Group means:
##            m.categoryComics m.categoryCrafts m.categoryDance
## failed           0.01999197       0.02753022     0.006316071
## successful       0.03956333       0.01436976     0.018088370
##            m.categoryDesign m.categoryFashion m.categoryFilm.Video
## failed           0.07177840        0.06852224            0.1744681
## successful       0.07079963        0.03814229            0.1890779
##            m.categoryFood m.categoryGames m.categoryJournalism
## failed         0.08100272      0.07765734          0.015888309
## successful     0.04673493      0.08194218          0.007623146
##            m.categoryMusic m.categoryPhoto m.categoryPublishing
## failed           0.1151345      0.03406932           0.11844418
## successful       0.1933012      0.02561855           0.09090668
##            m.categoryTech m.categoryTheater     Days  countryAU
## failed         0.09700700        0.02004550 35.38152 0.02228467
## successful     0.04484906        0.05314953 32.42218 0.01255030
##               countryBE  countryCA   countryCH   countryDE   countryDK
## failed     0.0014362817 0.04084928 0.001837727 0.009420581 0.002658459
## successful 0.0006640371 0.02694662 0.000770283 0.004727944 0.002071796
##              countryES   countryFR  countryGB    countryHK   countryIE
## failed     0.004933315 0.006101967 0.08583791 0.0002141041 0.001980463
## successful 0.001965550 0.004249837 0.08227419 0.0001062459 0.001195267
##              countryIT    countryLU   countryNL    countryNO   countryNZ
## failed     0.006985147 0.0001248941 0.008947768 0.0019626210 0.003693296
## successful 0.002031953 0.0001195267 0.003652204 0.0007968445 0.002908482
##              countrySE    countrySG countryUS   backers
## failed     0.004282082 0.0001873411 0.7949239  16.93669
## successful 0.002842079 0.0003585800 0.8492105 250.78255
```

Below are our results based of the test data:

```
qda.class    failed successful
  failed      22804       7077
  successful  33215      30600

               Accuracy : 0.57
                 95% CI : (0.5668, 0.5731)
    No Information Rate : 0.5979
    P-Value [Acc > NIR] : 1

                  Kappa : 0.1969
 Mcnemar's Test P-Value : <2e-16

            Sensitivity : 0.8122
            Specificity : 0.4071
         Pos Pred Value : 0.4795
         Neg Pred Value : 0.7632
             Prevalence : 0.4021
         Detection Rate : 0.3266
   Detection Prevalence : 0.6811
      Balanced Accuracy : 0.6096

       'Positive' Class : successful
```

This model was less accurate than our LDA model. This tells us that our predictors are more closely related to the response in a linear fashion. The true positive rate was 81.22% and our true negative rate was 40.71%. While this model was much better than the LDA and logistic regression model in sensitivity, it preformed overall worse at being 57% accurate in predicting whether a Kickstarter was successful or not.

Tree-Based Models

Tree-Based models segment the predictor space into certain regions based off a group of observations that share a similar mean or mode. There are two main steps that go into building a tree.

1. We divide the predictor space—that is, the set of possible values for $X_1, X_2, \ldots, X_p$—into J distinct and non-overlapping regions, $R_1, R_2, \ldots, R_j$
2. For every observation that falls into the region R_j, we make the same prediction, which is the mean of the response values for the training observations in R_j .

To create these regions, we want to find them in a way that minimizes the RSS given by:

$$\sum_{j=1}^{J} \sum_{i \in R_j} (y_i - \hat{y}_{R_j})^2$$

Where the second term is the mean response for the training observations within every jth box. We cannot consider every split possible in this so there is another technique called recursive binary splitting. In short this means we consider all points for our first split, and then consecutively split the space as we move down our tree. More specifically we define the half planes as:

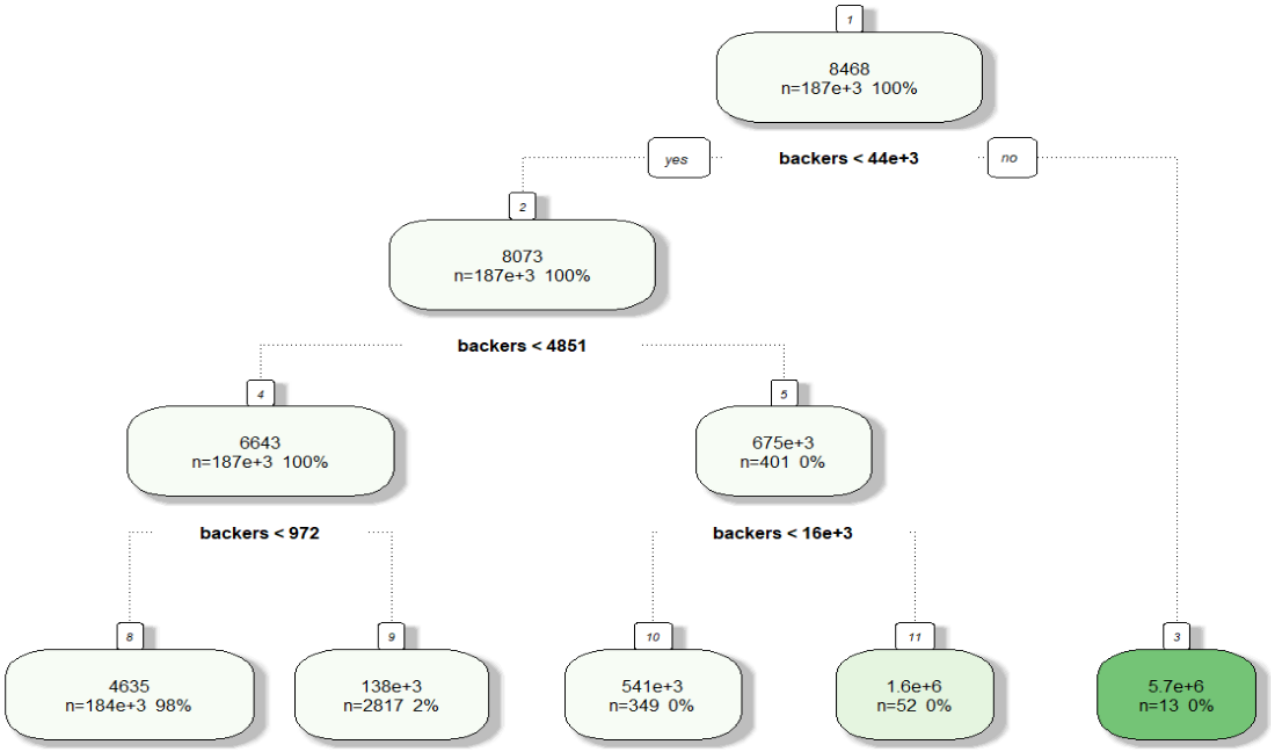
$$R_1(j, s) = \{X | X_j < s\} \text{ and } R_2(j, s) = \{X | X_j \geq s\}$$

We are looking for a value of *j* and *s* that minimize the equation:

$$\sum_{i:\, x_i \in R_1(j,s)} (y_i - \widehat{y_{R_1}})^2 + \sum_{i:\, x_i \in R_2(j,s)} (y_i - \widehat{y_{R_2}})^2$$

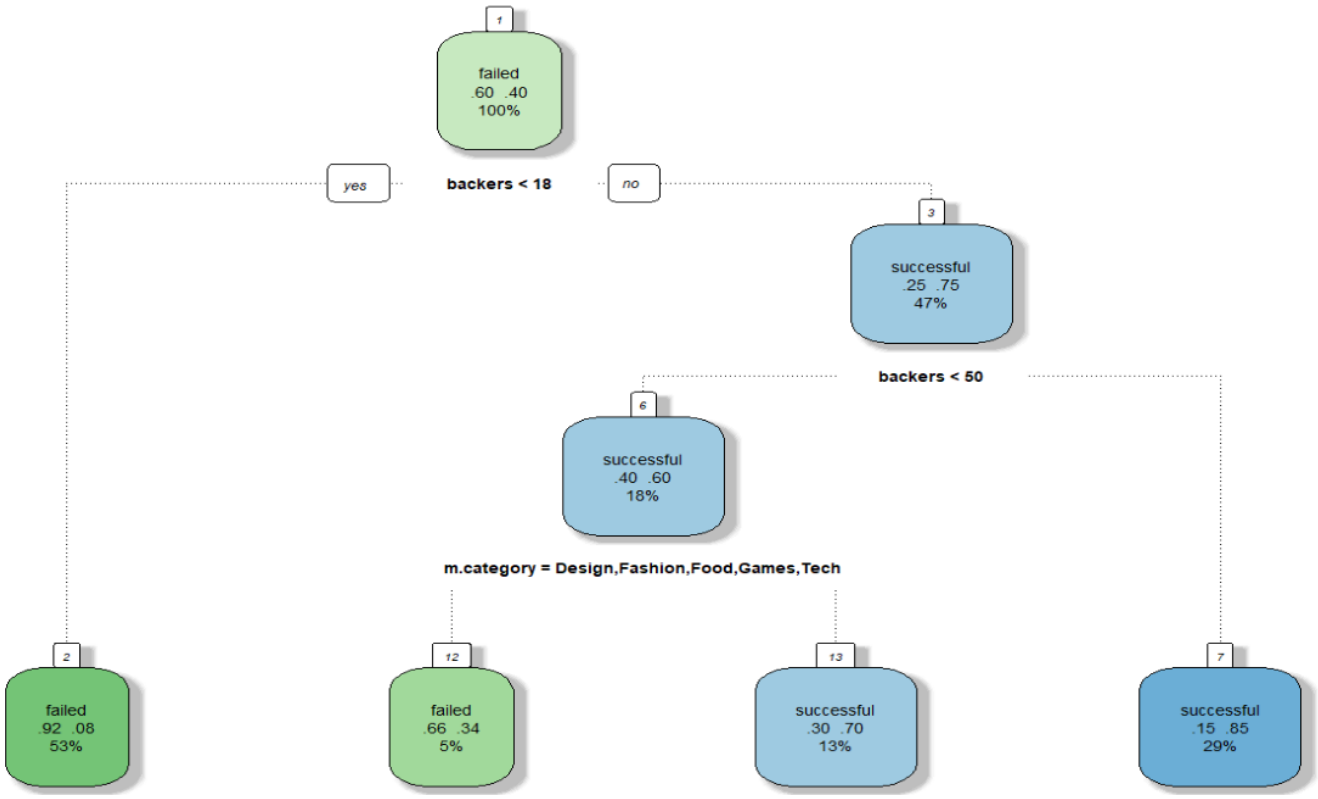
Then we continue to break the data into regions until some type of stopping criteria is reached. R will automatically decide what splits best describe the data by continually running through different splits and deciding which is best. Once the regions have been calculated we can predict the response of a given test observation by using the mean of the specified region to which the observation belongs. R will automatically choose the predictor to split the regions into, this predictor is the one that affects the response the most.

We made a tree-based model to see if we could accurately predict the amount of money a Kickstarter would raise as well. Below is our tree based off the training data set.



In the first region all of our projects were considered. The total mean was $8,468. Then we look at the condition under the region “backers<44,000”, if the condition is true we move to the left. If it is false we move to the right and so on and so forth. As with the Liner Regression model we calculated the RMSE with our test data and found that this model was off by an average of $83,494.

The process is a bit different if we want to build a classification tree. For this method we predict that each observation belongs to the most commonly occurring class of training observations in the region to which it belongs. R automatically runs through these calculations and the following tree was produced based off our training data set.



This tree is read the same way as the prior tree. In the first region where the entirety of the training data is examined we see that the majority of the projects failed. Then we look at the condition and see "backers <18", if it is true we move to the left, and if it is false we move to the right and examine more regions. In the first box when see ".6" and ".4", these indicate how many of the projects were failed or successful off the bat, and the entire region is labeled as to which of the groups was more prominent. If we move to the right to the next region we see ".25" and ".75" with the 47% in the region as well. This indicates that out of the total 100% that 47% can be broken down into the two statistics in this specified region, and that the majority of these projects were successful, so the region is labeled as such.

Running our test data through this tree gave us the following results:

```
cv.pred      failed successful
  failed      46430       4106
  successful   9589      33571

               Accuracy : 0.8538
                 95% CI : (0.8516, 0.8561)
    No Information Rate : 0.5979
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.7031
 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.8910
            Specificity : 0.8288
         Pos Pred Value : 0.7778
         Neg Pred Value : 0.9188
             Prevalence : 0.4021
         Detection Rate : 0.3583
   Detection Prevalence : 0.4606
      Balanced Accuracy : 0.8599

       'Positive' Class : successful
```

This model was very accurate with a Sensitivity of 89.10% and a Specificity of 82.88% and was overall 85.38% accurate.

Conclusion

The Linear Regression model for USD Pledged was the most accurate in trying to predict how much money a Kickstarter would raise. This model's RMSE was $73,124 while the tree model had an RMSE of $83,494. While we were not able to build a model that could accurately predict the pledged/goal ratio of a Kickstarter this does not mean that this quantity cannot be predicted, what it means is that the predictors and the model we choose could not accurately show any relationship. It is possible that another model could show some relationship. We also tried to build a tree model for predicting the pledged/goal ratio, but the algorithm in R could not determine a most important predictor.

In our classification models the Tree-Based model was the most overall accurate with correctly predicting the state of a Kickstarter 85.38% of the time and producing the most accurate Sensitivity of 89.10%. A few interesting notes are that while our Logistic Regression model was not the most accurate, it did have the most accurate Specificity of all the state models we created of 93.67%. Also, while our LDA model was not the most accurate model to predict the state of a Kickstarter it did have a better Specificity than our Tree-Based model as well (86.58% compared to our Tree Model of 82.80%).

References

James, Gareth, et al. *An Introduction to Statistical Learning with Applications in R*. Springer, 2017 pp.. 62,73,131,140,150