# Predicting YouTube Video Popularity

*Springboard Capstone Project*

*by Sara Peters*

Summer 2019

## *Capstone Project Deliverables*

The deliverables for this capstone project include code, final paper documentation, and a visual slide deck presentation.

## *Background*

Ranked as the second-most popular website in the world, the Google-owned video-hosting site YouTube claims that one billion hours of user-generated content is watched on the broadcast site each day. Yes, that's right. One billion hours per day. This global phenomenon is thanks in large part to the video site's machine learning recommendation capabilities. Although some trends are predictable, like when a new song is uploaded from a popular artist or when a new superhero movie trailer is released and gains millions of views in a single day, others are more surprising, like a viral cat video that transpires into an overnight sensation or a new dance craze that is quickly shared across the internet.

The record-breaking exposure channels can receive on the video platform has changed the game of marketing and afforded new career opportunities. "YouTubers," also known as YouTube personalities, celebrities, or content creators, are people who have gained popularity from their videos on their YouTube channels. Those who are able to amass a huge following can take advantage of corporate sponsorships and sign deals to have product lines of their own.

## *Introduction to the Problem*

So how does one become a YouTuber? How does a YouTube video become popular in the first place? The answer partially lies in YouTube's Trending tab capabilities. The Trending tab features videos algorithmically, using a combination of factors, including total watch time, view count, rate of growth in viewership, the age of the video and other user-engagement markers such as shares, comments, and ratings. Videos that meet a certain threshold are then given a high-profile spot on YouTube's home page and can rack up millions or even billions of views in a short period of time. But the question still remains: How does a video get on the Trending tab to begin with?

Because there are a number of factors to consider when predicting what makes a video popular to one audience and not another, this project will attempt to solve the following problem:

**What factors affect how popular a YouTube video will be and, using such factors, can we predict popularity for any video?**

### *Potential Clients*

Private individuals to large production companies have used YouTube to grow their audience base. Because of this large range of interests, there are several clients who would want to know the answer to this project's question. In particular, entertainment content creators such as movie studios or musical artists would be interested in knowing how to ensure that their videos reach a wide audience who would then purchase a movie ticket or buy a single to download. Likewise, advertisers who use music in their commercials to sell a product such as a luxury car would want to know which artists are trending to increase the likelihood of viewership for their sponsored video ads. Beyond entertainment products, clients may want to create a home improvement channel, document breathtaking travel destinations, or share information through video lessons designed for kids.

### *The Data Source*

This project used an open-source data set containing a daily record of the top YouTube videos over a period of several months called *Trending YouTube Video Statistics*. Within this data set, 16 variables were collected across 10 different regions (USA, Great Britain, Germany, Canada, France, Russia, Mexico, South Korea, Japan, and India). The data set was released on Google's public data science platform, Kaggle, and collected using the YouTube API. Click on this **link** to access the data set.[1]

The 16 variables include: a unique video ID, the trending date, the published time, the video title, the channel title, the category (which varies based on region), tags, the video's description, view count, the number of likes and dislikes, whether the ratings were disabled, comment count, whether the comments were disabled, thumbnail link, and whether there was a video error or if the video was removed for violating YouTube's terms of service.

### *Deeper Dive into the Data Set*

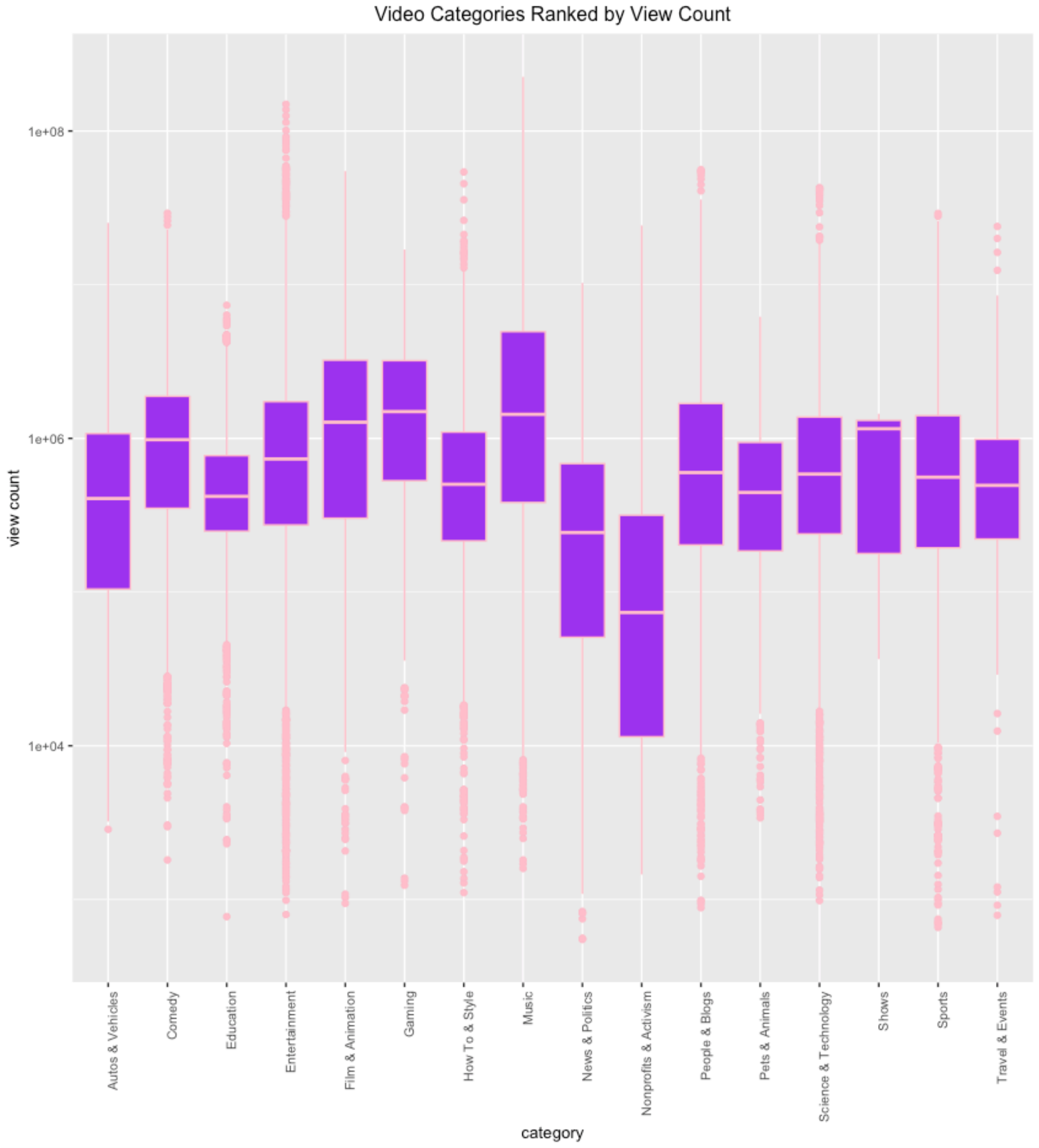
To solve the problem, **factors** and **popular** have to be defined. Based on the variables collected in the original data set above, *tags* will be the **factors**. *Views*, *likes*, and *comments* will be summed to create a single measure to define what makes a video **popular**. For this project, the *title* and *description* factors were not used because doing so could bias the results based on the particular style the content creators used to write them.

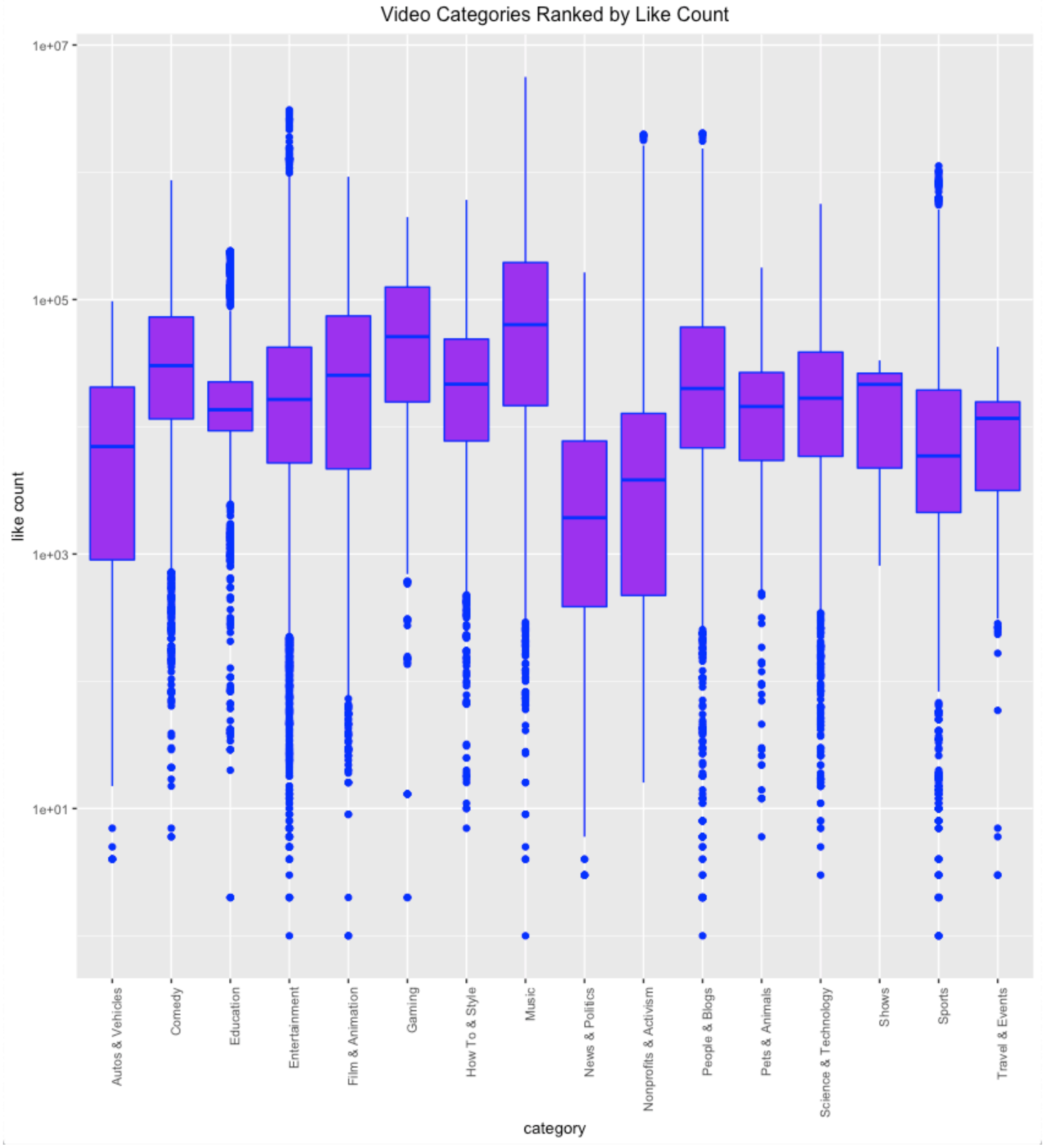
Ultimately, the purpose of the project will be to build a model that predicts whether a YouTube video is classified as popular or unpopular based on the *tags* used. For example, if the client uses _____ *tags*, the video will be either popular or unpopular.

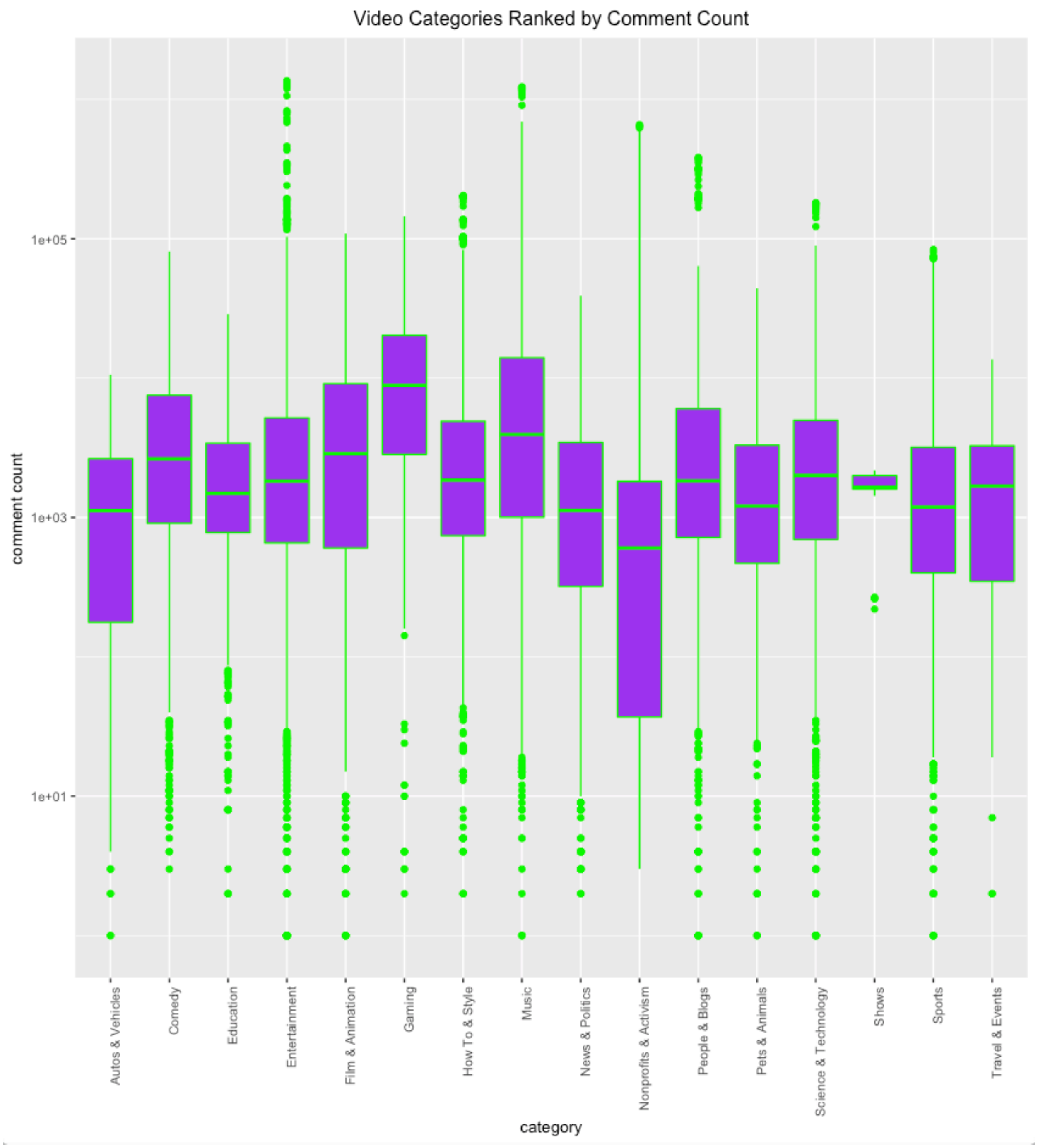
An important feature included in the original data set is the observations collected across 10 different regions. Due to time constraints, the predictive model for this project will not account for the country in which a video trended and instead only analyze U.S. videos. However, this information could be quite significant for clients who serve an international base.

[1] *Trending YouTube Video Statistics:* https://www.kaggle.com/datasnaek/youtube-new/home

During the data exploration phase of the project, it was discovered that the majority of the videos mostly fall into two categories – *Entertainment* and *Music.* When looking at the relationship between video category and *view*, *like*, and *comment* count, it's clear that the *Entertainment* and *Music* categories received the largest amount of user engagement across these three factors. This is important to recognize and will be discussed later in this paper.



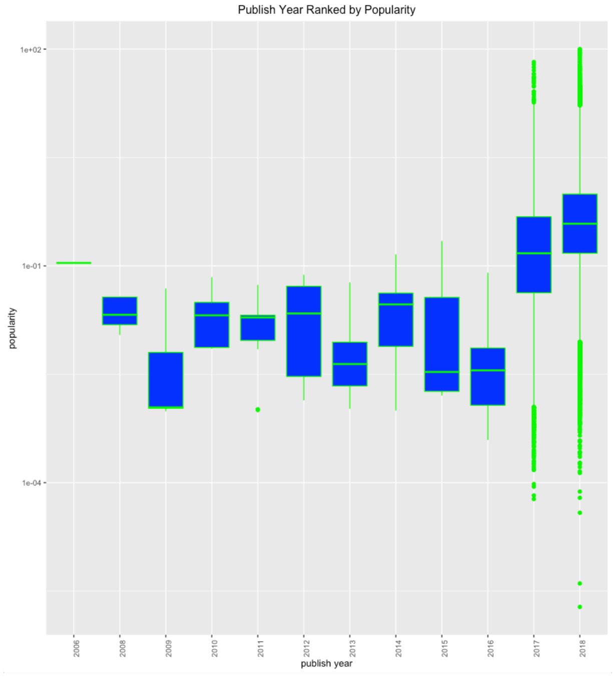

Video Categories Ranked by Like Count
like count
1e+07
1e+05
1e+03
1e+01
Autos & Vehicles
Comedy
Education
Entertainment
Film & Animation
Gaming
How To & Style
Music
News & Politics
Nonprofits & Activism
People & Blogs
Pets & Animals
Science & Technology
Shows
Sports
Travel & Events
category

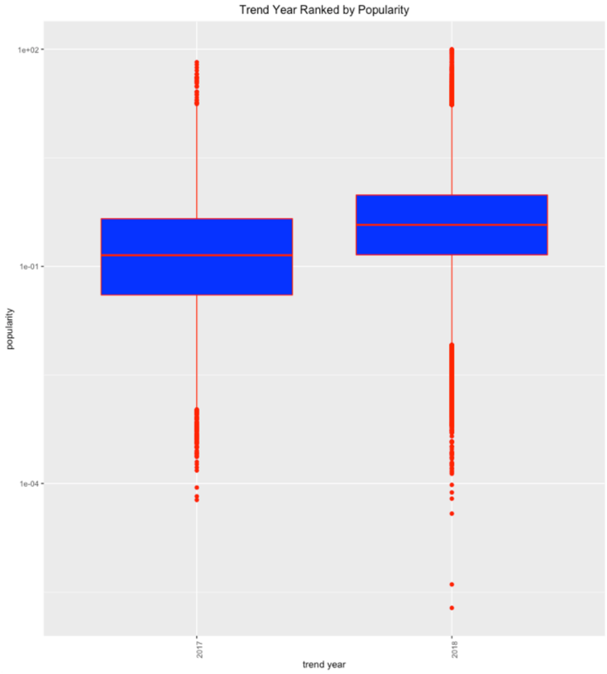


Further statistical analysis revealed that the range for the published year differed from that of the trending year. As can be seen in the table below, the trending year only covers 2017 and 2018 while the published year accounts for 2006 and 2008 through 2018. This is important to recognize because there is no data for the trending year before 2017, so a comparison of these two time markers cannot be made beyond these two years (2017 & 2018). Likewise, to include the single observation in 2006 for the published year would skew the data given that over 30K observations were made in 2018.

| Number of Videos *Published* in Each Year | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 2006 | 2008 | 2009 | 2010 | 2011 | 2012 | 2013 | 2014 | 2015 | 2016 | 2017 | 2018 |
| **1** | **11** | **14** | **19** | **27** | **24** | **44** | **32** | **35** | **35** | **10428** | **30279** |
| **Number of Videos *Trending* in Each Year** | | | | | | | | | | | |
| | | | | | | | | | | 2017 | 2018 |
| | | | | | | | | | | **9600** | **31349** |

The box plots below compare the videos' *popularity* by the year they were first published to the videos' *popularity* by the year they trended. Future research that analyzes *popularity* across time should only include videos that were published in 2017 and 2018.





## *Data Wrangling Steps*

After cleaning and wrangling the original data set, there was a total of 40,949 observations. The most important steps taken to clean up the original data set to use in this project include: (a) creating a *popularity* measure, which would serve as the outcome or dependent variable and (b) pre-processing the *tags* factor to transform the character strings into independent variables.

The outcome or dependent variable *popularity* is a single measure that was created from a combination of multiple factors. The factors used to create the *popularity* measure include the number of video *views*, *likes*, and *comments*. Because the comments and ratings were disabled for some videos, those videos had 0 *likes* and/or 0 *comments*. To adjust for this, those videos where the comments and ratings were marked as disabled in the data set were replaced with the average count for the *comments* (mean = 8,447) and the average count for the *likes* (mean = 74,267).

A formula was created to form the *popularity* measure. In the formula, the *views*, *likes*, and *comments* were summed for each video in the data set. A sensible weight was chosen and assigned to each factor to create the total *popularity* score. To determine how each factor should be weighted, research was done to understand YouTube's ranking system. Based on how the platform ranks videos, it was determined that *likes* were the most valuable indicator of *popularity* because a viewer is actively showing their approval of the video. *Views* are the second most valuable indicator of *popularity* because the more *views* a video receives, the more exposure it gets. However, YouTube measures *total watch time* and ranks this as a more important factor than the number of *views* a video acquires. In other words, YouTube gives more weight to videos that a viewer watches for a longer period of time. For example, even if Video A had twice as many *views* as Video B, if Video B was 5 times longer than Video A, Video B would still be ranked higher. Lastly, *comments* were the least most valuable indicator of *popularity* because these may represent negative sentiments, which would signal a viewer's disapproval of the video. In the data set used for this project, type of sentiment (positive or negative) was not accounted for. Although video *comments* have shown to have a very strong correlation with rankings, the relationship is not significant as it is for both *likes* and *views*.[2] Therefore, this project doubled the weight of *likes*, kept *views* the same, and decreased the weight of *comments* by half to determine relative *popularity* for each video.

Here is the weighted *popularity* formula:

```
USvideos_new$popular <-  1*USvideos_new$views + 2*USvideos_new$likes +
0.5*USvideos_new$comment_count
```

Once the formula was built to generate the *popularity* measure, it was then normalized so that all observations were on the same scale between 0 to 100.

```
USvideos_new$popular_norm <- rescale(USvideos_new$popular, to = c(0, 100))
```

To use the *popularity* measure as a categorical, dependent variable for a classification model, a threshold was applied to determine whether a video was popular or unpopular. When looking at the data, if the threshold was set at 50 (so that those videos with a normalized *popularity* score of 50 or above would be popular and those below 50 would be unpopular), then only 34 out of the total 40,949 videos would be considered popular. This is a very strict threshold and wouldn't provide enough useful data to build an accurate predictive model. This is because if a model was trained using this threshold, the model would predict that all videos are popular. The accuracy of the model would be high, but it would not be very useful because it is predicting that everything is popular.

Again, after looking at the data, if the threshold was set at 1, this would show 8,712 videos as being popular. While this is a significant improvement over the threshold of 50, the majority of these 8,712 videos have at least 2 million *views*, which is still quite a bit. It's reasonable to

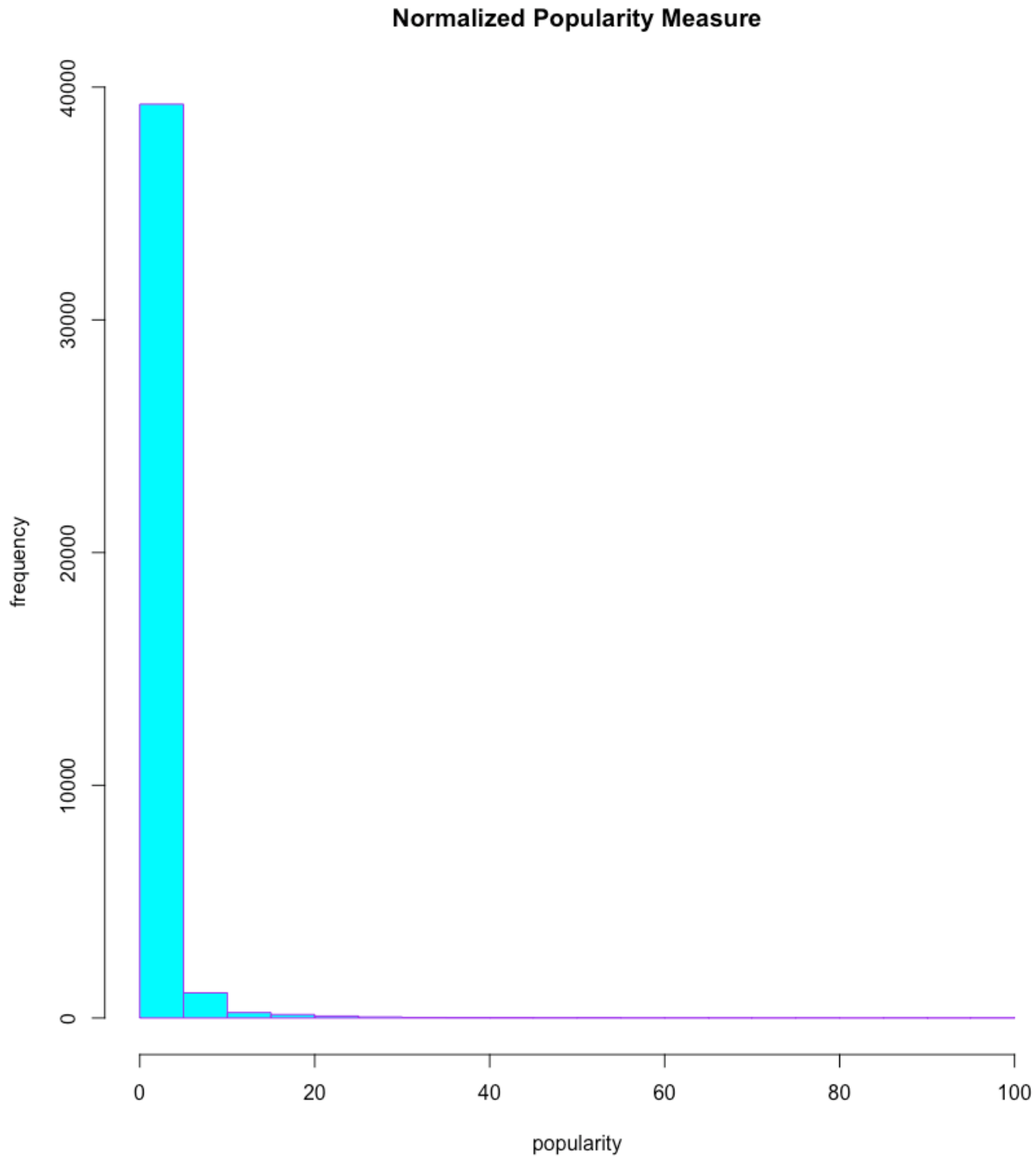
[2] Dean, B. (2017, February 28). We Analyzed 1.3 Million YouTube Videos. Here's What We Learned About YouTube SEO. Retrieved July 15, 2019, from https://backlinko.com/youtube-ranking-factors

conclude that popular videos could include those with at least 1 million *views*. Therefore, the threshold was lowered to 0.5.

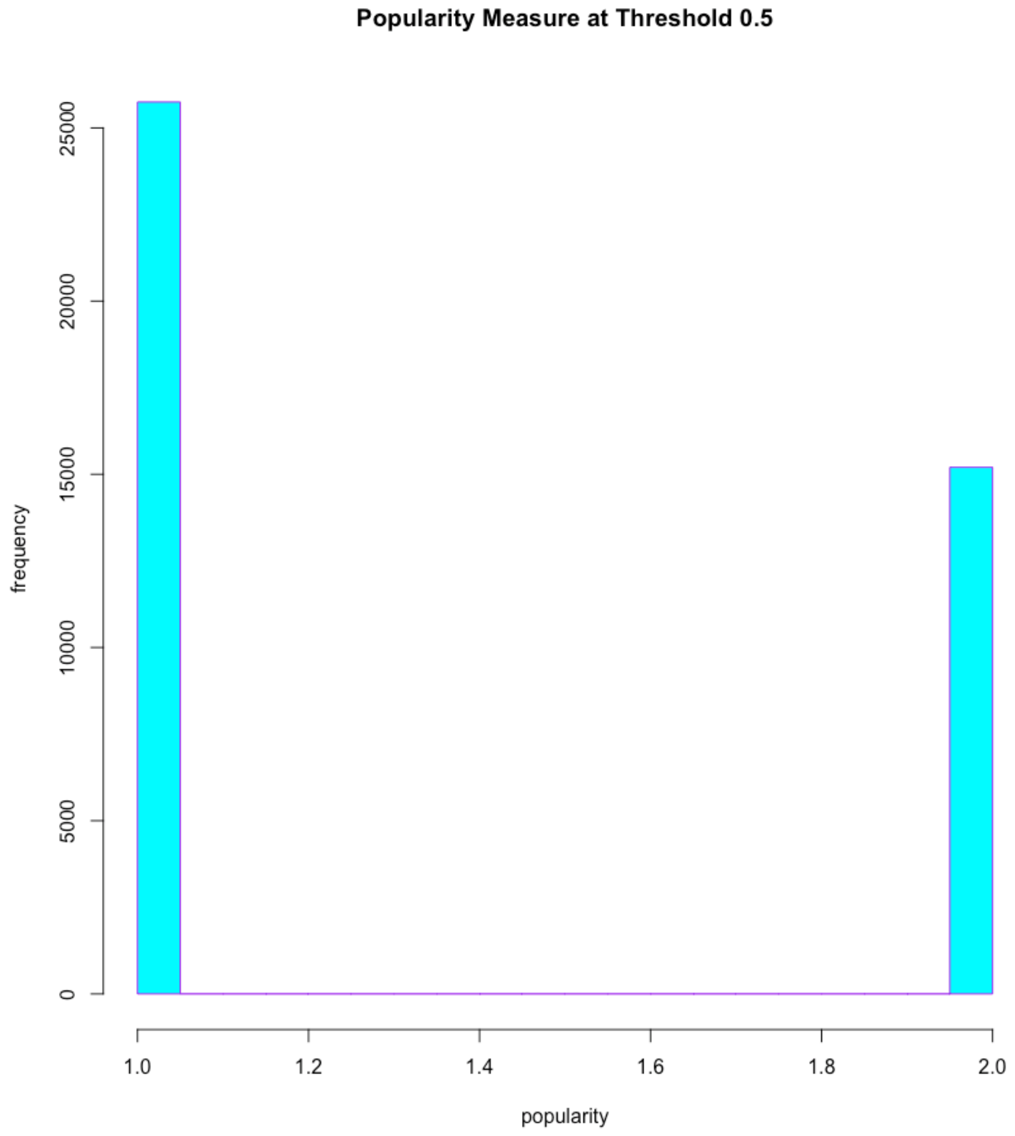
Now, when looking at the data set, most videos at or above the applied threshold of 0.5 have 1 million *views*, a 5-figure *like* count, and a 4-figure *comment* count. Using this refined threshold, the following rules will hold: If a video is popular, it will have a score of 0.5 or above; however, if a video is unpopular, it will have a score of below 0.5.

```
USvideos_new$popular_video = as.factor(USvideos_new$popular_norm >= 0.5)
```

Looking at the shape of the data using a histogram, it's clear that the normalized *popularity* measure is a highly skewed metric.



Because of this, a second histogram was created to see the shape of the data where it was split at the threshold of 0.5.

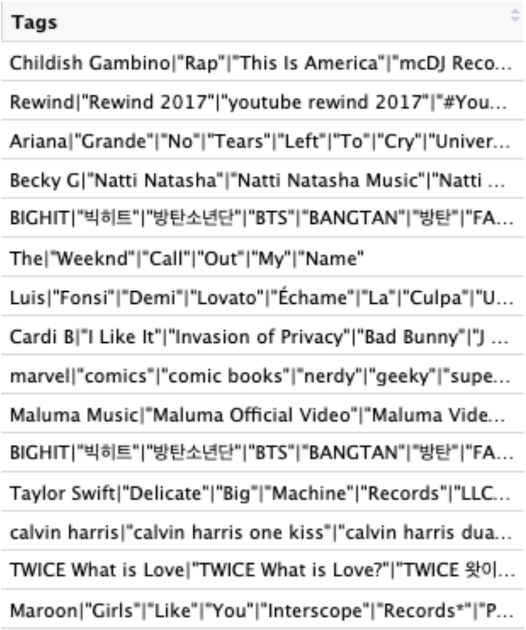


```
table(USvideos_new$popular_video)
```

In the end, 15,201 videos were categorized as popular while 25,748 videos were categorized as unpopular. The 0.5 threshold produces more reasonably balanced classes with a ratio of about 3:5, suggesting that the machine is learning something because it is not seeing the videos as either all popular or all unpopular. Thus, this threshold will be used to build the classification models. In fact, when the models in this project use the 0.5 threshold to classify videos as either popular or unpopular, the confusion matrices will show a relatively even split of videos that were correctly identified as popular or unpopular as well as those that were mistakenly identified as popular or unpopular. This performance metric seems to tell a more truthful story regarding the accuracy of the prediction models applied in this project.

The *tags* are the independent variables or predictors used in this project to predict the outcome or dependent variable, *popularity*.

A limitation of the current project stems from the original *tags* factor. In the data set, the *tags* were compiled into a list separated by the pipe or vertical bar character (|). Because of this, some of the *tags* were misspelled or contained punctuation, character, or spacing issues. For example, one *tag* is “ed sheeran” and another *tag* is “edsheeran”. If these *tags* were not combined, this would result in two different *tags* each counted once instead of one *tag* being counted twice. It’s too hard to tell if misspellings such as this were done intentionally to match the common search terms viewers use to look up a video. It makes sense to use the *tags* as they were originally entered; however, the frequencies would be quite low if similar *tags* were not compiled together. These low frequencies would not produce very useful findings for potential clients.



| Tags |
|---|
| Childish Gambino\|"Rap"\|"This Is America"\|"mcDJ Reco... |
| Rewind\|"Rewind 2017"\|"youtube rewind 2017"\|"#You... |
| Ariana\|"Grande"\|"No"\|"Tears"\|"Left"\|"To"\|"Cry"\|"Univer... |
| Becky G\|"Natti Natasha"\|"Natti Natasha Music"\|"Natti ... |
| BIGHIT\|"빅히트"\|"방탄소년단"\|"BTS"\|"BANGTAN"\|"방탄"\|"FA... |
| The\|"Weeknd"\|"Call"\|"Out"\|"My"\|"Name" |
| Luis\|"Fonsi"\|"Demi"\|"Lovato"\|"Échame"\|"La"\|"Culpa"\|"U... |
| Cardi B\|"I Like It"\|"Invasion of Privacy"\|"Bad Bunny"\|"J ... |
| marvel\|"comics"\|"comic books"\|"nerdy"\|"geeky"\|"supe... |
| Maluma Music\|"Maluma Official Video"\|"Maluma Vide... |
| BIGHIT\|"빅히트"\|"방탄소년단"\|"BTS"\|"BANGTAN"\|"방탄"\|"FA... |
| Taylor Swift\|"Delicate"\|"Big"\|"Machine"\|"Records"\|"LLC... |
| calvin harris\|"calvin harris one kiss"\|"calvin harris dua... |
| TWICE What is Love\|"TWICE What is Love?"\|"TWICE 왓이... |
| Maroon\|"Girls"\|"Like"\|"You"\|"Interscope"\|"Records*"\|"P... |

Because the *tags* were character vectors, text analysis was performed to make this factor more manageable. First, the data had to be pre-processed to apply the Bag of Words technique, which would transform the text into independent variables. Instead of building customized code to wrangle the *tags*, the Bag of Words technique was used because it is a more scalable process. The *tags* were converted into a corpus, or a collection of documents, to prepare for the pre-processing steps. The tm_mp function was applied to perform each operation: converting all letters to lowercase, removing all punctuation, removing all stop words, and stemming, which used a rule-based algorithm. It should be noted that the pre-processing steps were applied to English-only words and because this data set includes Korean letters, this is a limitation of the project.

Once the four pre-processing steps were completed, the word frequencies were extracted to be used in the prediction problem. This was done by using the Document Term Matrix function to generate a matrix where the rows correspond to documents (in this case videos) and the columns correspond to words (in this case *tags*) that are associated with those videos. The values in the matrix are the number of times a *tag* appears with each video. Of the 40,949 total videos in the original data set, there were 19,856 *tags*.

Upon inspecting the matrix, it is determined that the data is sparse. When data is sparse, meaning there are many zeros in the matrix, the findFreqTerms function is used to look at what the most frequent *tags* are, so that those *tags* that don't appear too often can be removed. Out of the 19,856 *tags*, only 2,499 *tags* appeared at least 100 times or more with the videos. This means that there are a lot of *tags* that will be pretty useless for a prediction model if they were all to be included. The number of *tags* is an issue for two main reasons. The first reason is computational such that more *tags* means there will be more independent variables, which usually suggests that it will take longer to build a model. Second, when building a model, the ratio of independent variables to observations will affect how well the model will generalize.

Because it isn't useful to count similar *tags* separately, *tags* that referred to the same song or person could have been consolidated into one *tag* name to increase a *tag*'s frequency count. Although this process was not performed on this project, it would be limited by the pop-culture knowledge of the data scientist who cleaned and wrangled the data. For example, there are many *tags* that refer to the artist *Jennifer Lopez* but not everyone working on this data set would know that her stage name is *J. Lo* and that one of her nicknames is *Jenny from the Block* based on her 2002 hit single of the same name. Therefore, the frequency for the *tag jennifer lopez* may be higher or lower depending on the data scientist's fluency in pop-culture references. This is a potential dimension to consider for future research.

For the present project, *tags* that didn't appear very often were removed from the data set. To do this, the removeSparseTerms function was used. The sparsity threshold, or the percentage of *tags* that were kept, was 0.995 such that only those *tags* that appeared in 0.5% or more of the videos were kept. In the newly constructed sparse matrix, only 740 *tags* were retained or 3.7% of the previous count of 19,856. This much more reasonable number will help construct a more useful and accurate predictive model. To build this model, the sparse matrix was then converted into a data frame. Because the data frame was built using text analytics, all variable names were converted to appropriate names using the make.names function so that none started with a number. For example, "2017" was converted to "X2017".

Before splitting the data into a training and testing set to run the text analytics models, the outcome variable, *popularity*, was added. Looking at the updated data frame, there are 740 independent variables that represent the frequencies of the *tags* used for at least 0.5% of the videos and the last variable is the dependent, outcome variable *popularity* (whether or not a video was popular).

Lastly, the data was split into a training set and a testing set to then actually build a model. To do this, 70% of the data was put into the training set and a seed was set so that the results of this project could be reproducible, if needed.

Now, the data is ready and the predictive models can be built.

### *Framing the Main Question*

The original question for this capstone project was:

*What factors affect how popular a YouTube video will be and, using such factors, can we predict popularity for any video?*

To reframe this question as a machine learning problem, the following was used:

**Which tags predict popularity for a YouTube video?**

This is a supervised learning problem because all the data is labeled and the algorithms employed learn to predict the output from the input data. More specifically, the algorithms, or models, learned from the training data set to make predictions on the unseen data of the testing data set. The best predictive model was selected after it achieved an acceptable level of performance.

For this project, a classification supervised learning problem was used because the output variable, in this case *popularity*, is a categorical variable in that a video was ranked as being either popular or unpopular.

## *The Approach: Machine Learning Techniques and Evaluation*

For this project, three machine learning techniques were applied in addition to constructing a simple baseline model.

### *The baseline model*

The baseline model that always predicts an unpopular video was built by making tables of just the outcome variable.

In the training set, there are 18,024 observations that are unpopular and 10,641 observations that are popular. The accuracy of the baseline model that always predicts non-popular (or unpopular videos) is 0.6288 or about 63%.

```
table(trainSparse$popular_video)
```

In the testing set, there are 7,724 observations that are unpopular and 4,560 observations that are popular. The accuracy of the baseline model that always predicts non-popular (or unpopular videos) is 0.6287 or about 63%.

```
table(testSparse$popular_video)
```

### *The logistic regression model*

A logistic regression model can be used to predict the categorical outcome (whether a video was popular or not). The logistic regression model computes probabilities that can be used to assess the confidence of the prediction.

```
USvideosLOG <- glm(popular_video ~ ., data = trainSparse, family = binomial)
```

The AIC, or the measure of the quality of the logistic model that accounts for the number of variables used compared to the number of observations is 27,922.

To get the probabilities of the predictions:

```
predictLOG <- predict(USvideosLOG, type= ”response”)
```

To see if the model is predicting higher probabilities for the actual popular videos as expected:

```
tapply(predictLOG, trainSparse$popular_video, mean)
```

For all of the TRUE popular videos, the model predicts an average probability of about 0.59. For all of the FALSE popular videos (meaning unpopular) the model predicts an average probability of about 0.24. This is a good sign because it looks like the model is predicting a higher probability for the actual popular videos.

A threshold of 0.5 was initially selected and applied.

```
table(trainSparse$popular_video, predictLOG > 0.5)
```

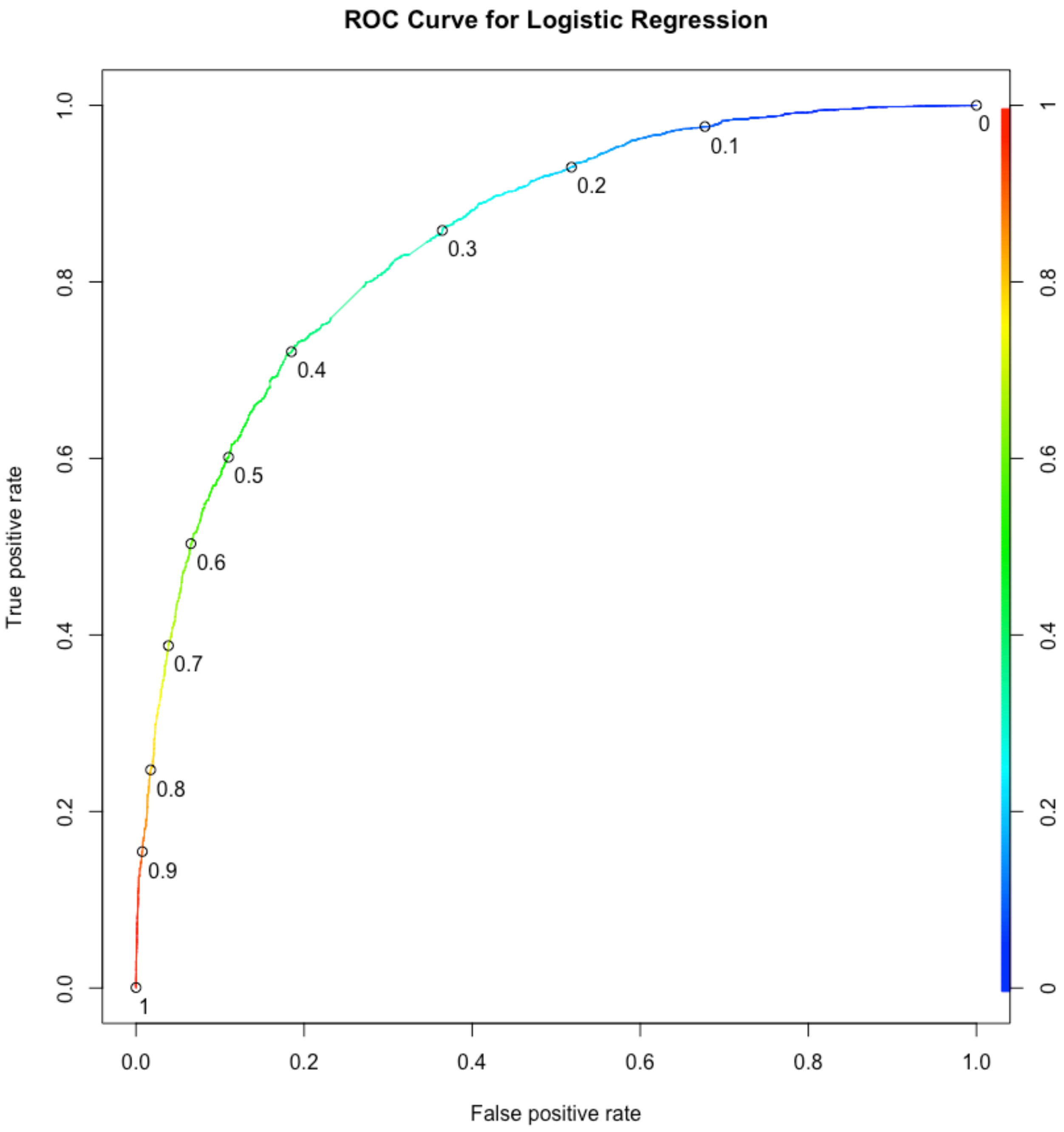
Results show 0.6014 as the Sensitivity, or the True Positive Rate, which is the percentage of actual popular videos that were classified correctly and 0.8900 as the Specificity, or the True Negative Rate, which is the percentage of actual unpopular videos that were classified correctly. Because this project is more concerned with having a high Sensitivity, or high True Positive Rate, a new threshold was selected and applied to minimize the False Positive Rate and maximize the True Positive Rate. This was done because the goal of this project's prediction model is to discover which *tags* are associated with popular videos, so being able to more accurately determine which videos are popular is a higher priority.

To help decide which threshold to use, a ROC curve was constructed and then plotted.

```
predLOG <- prediction(predictLOG, trainSparse$popular_video)
```

```
prefLOG <- performance(predLOG, “tpr”, “fpr”)
```

```
plot(prefLOG, main = “ROC Curve for Logistic Regression”, colorize = TRUE, print.cutoffs.at = seq(0 ,1, 0.1), text.adj = c(-0.2, 1.7))
```



Based on the ROC curve, a threshold of 0.3 would be best in order to maximize the True Positive Rate while minimizing the False Positive Rate. Using this threshold will increase the Sensitivity or the percentage of actual popular videos the model classified correctly.

After the new threshold of 0.3 is applied:

```
table(trainSparse$popular_video, predictLOG > 0.3)
```

The Sensitivity went up to 0.8583 (compared to 0.6014) while the Specificity went down to 0.6356 (compared to 0.8900).

To get the AUC value on the training set:

```
as.numeric(performance(predLOG, “auc”)@y.values)
```

The AUC value on the training set, or area under the curve, is 0.8497.

Next, the logistic regression model was used to make predictions on the testing set.

```
predictLOG_test <- predict(USvideosLOG, type = ”response”, newdata = testSparse)
```

Using the table function, a confusion matrix was created with a threshold of 0.3, which is the optimized threshold based on the training data.

```
table(testSparse$popular_video, predictLOG_test > 0.3)
```

The accuracy of this model is 0.7095 or about 71%, which is better than the baseline model of 63%.

To get the AUC value on the testing set:

```
predLOG_ROCR <- prediction(predictLOG_test, testSparse$popular_video)
```

```
as.numeric(performance(predLOG_ROCR, “auc”)@y.values)
```

This gives an AUC value of 0.8373 or about 84% on the testing set, which means that the logistic regression model can differentiate between popular videos and unpopular videos pretty well.

### *The CART model*

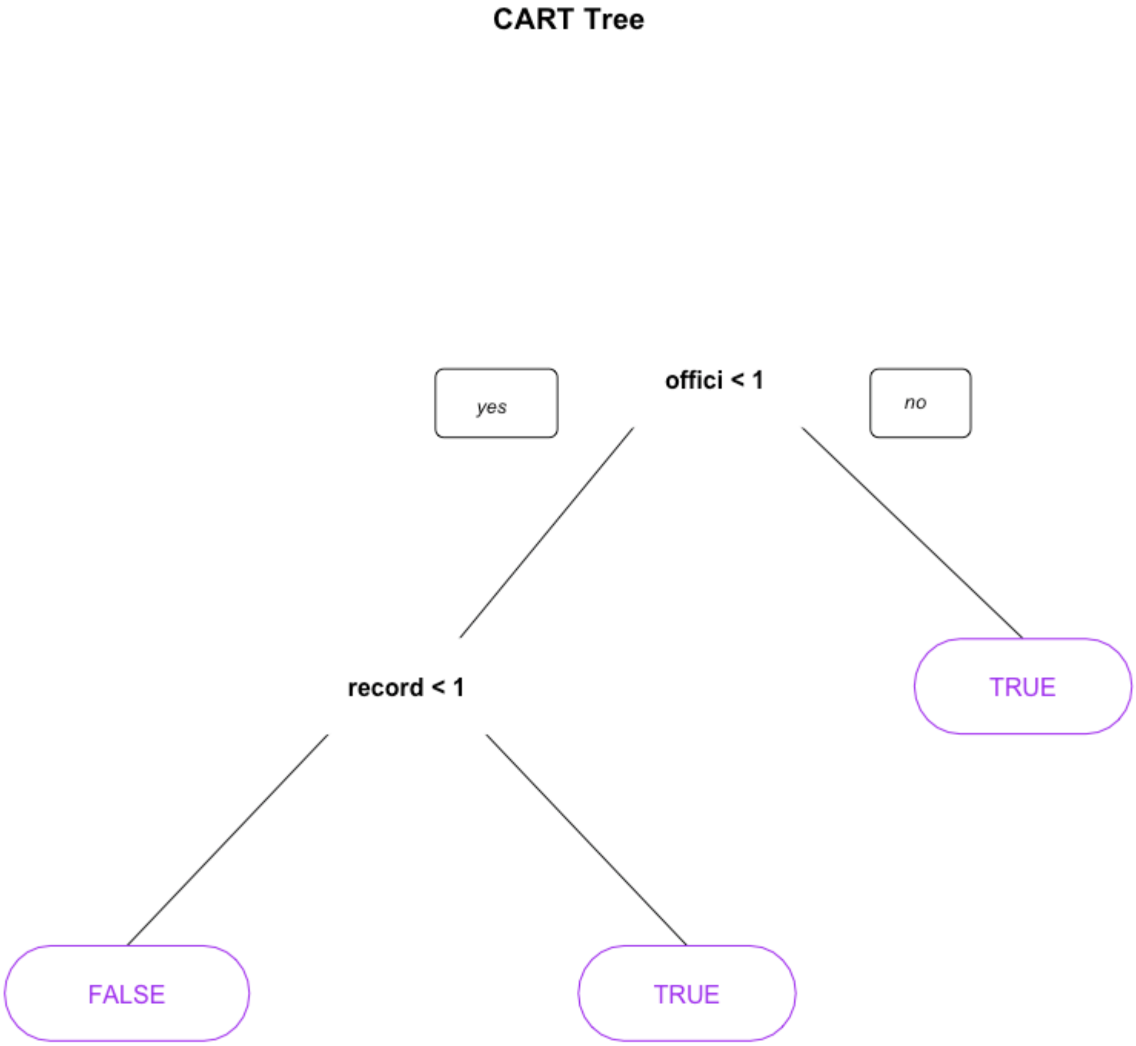
Besides a logistic regression model, CART (Classification and Regression Trees) can be used to predict a categorical outcome (in this case *popularity*). An advantage of the CART model is that it is very interpretable. This model produces a CART tree, which is a series of decision rules presented in a way that can easily be explained.

First, the CART model was built:

```
USvideosCART <- rpart(popular_video ~ ., data = trainSparse, method = “class”)
```

Then the CART model was plotted:

```
prp(USvideosCART, main = “CART Tree”, col = “purple”)
```



According to the CART tree, if the *tag* “offici” is in the video *tags*, then predict TRUE or predict that the video is popular. If the *tag* “offici” is not in the video *tags*, but the *tag* “record” is, then again predict TRUE or predict that the video is popular. If the *tags* “offici” and “record” are not in the video *tags*, then predict FALSE or predict that the video is not popular. This intuitively makes sense because the top two video categories in this data set are *Entertainment* (for 9,964 videos) and *Music* (for 6,472 videos), which would include these two *tags*.

To evaluate the performance of the CART model, predictions were made on the training and testing sets:

```
predictCARTTrain <- predict(USvideosCART, newdata = trainSparse, type = “class”)
```

```
predictCART <- predict(USvideosCART, newdata = testSparse, type = “class”)
```

Confusion matrices were then built to compute the accuracy of the CART model on the training and testing sets:

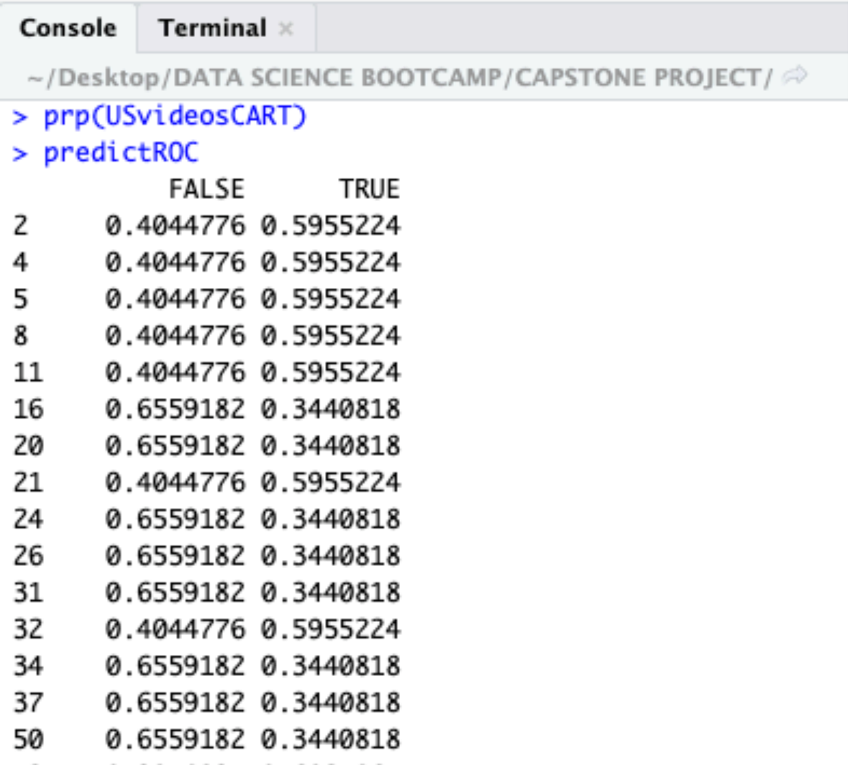
table(trainSparse$popular_video, predictCARTTrain)

table(testSparse$popular_video, predictCART)

The accuracy of the CART model on the training set is 0.6499 or about 65% and the accuracy of the CART model on the testing set is 0.6548 or about 65%.

Next, the ROC curve was generated for the CART model:

predictROC <- predict(USvideosCART, newdata = testSparse)



```
> prp(USvideosCART)
> predictROC
          FALSE      TRUE
2     0.4044776 0.5955224
4     0.4044776 0.5955224
5     0.4044776 0.5955224
8     0.4044776 0.5955224
11    0.4044776 0.5955224
16    0.6559182 0.3440818
20    0.6559182 0.3440818
21    0.4044776 0.5955224
24    0.6559182 0.3440818
26    0.6559182 0.3440818
31    0.6559182 0.3440818
32    0.4044776 0.5955224
34    0.6559182 0.3440818
37    0.6559182 0.3440818
50    0.6559182 0.3440818
```

The output displays two columns for each of the testing set observations, labeled FALSE and TRUE. The first column (FALSE) is the percentage of training set data in the same subset as that testing set observation that had outcome FALSE. The second column (TRUE) is the percentage of training set data in the same subset as that testing set observation that had outcome TRUE. The second column (TRUE) can be interpreted as the probability that that testing set observation has outcome TRUE. This second column (TRUE) is used to pick the threshold value when evaluating the CART model.

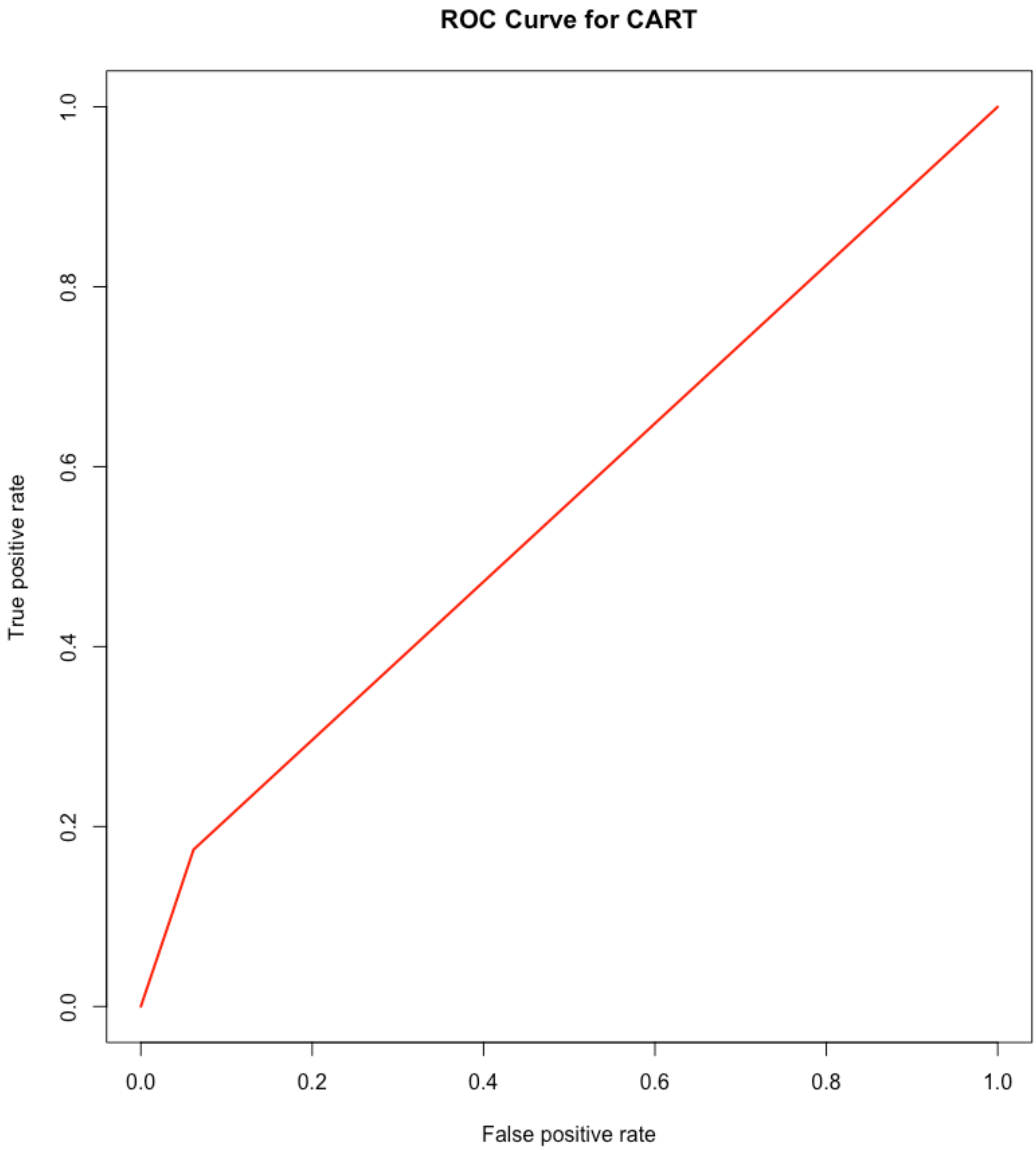
The ROC curve is generated for the CART model just as was done for logistic regression:

pred <- prediction(predictROC[, 2], testSparse$popular_video)

perf <- performance(pred, “tpr”, “fpr”)

Then the ROC curve was plotted:

```
plot(perf, main = “ROC Curve for CART”, col = 2, lwd =2)
```



To get the AUC value on the testing set:

```
as.numeric(performance(pred, “auc”)@y.values)
```

This gives an AUC value of 0.5564 or about 56% on the testing set, which means that the CART model doesn’t differentiate between popular videos and unpopular videos too well.

To improve upon the accuracy of the CART model, it was necessary to make sure the proper cp was selected. When cross-validation is used in R, a parameter value called cp, known as complexity parameter, is employed. For this project, the k-fold cross-validation method was applied to properly select the parameter value. This method averages the accuracy of the model over the k-folds to determine the final parameter value to use. The cp measures the trade-off between model complexity and accuracy on the training set. If the parameter value is too small, then the accuracy is lower because the model is probably overfit to the training set. If the parameter value is too large, the accuracy is also lower because the model is too simple.

Therefore, cross-validation was used to select the cp value for the CART tree by way of the k-folds method.

First, the number of folds were defined. For this project, 10 folds were applied.

```
fitControl <- trainControl(method = “cv”, number = 10)
```

Second, the possible values for the cp were selected.

```
cartGrid <- expand.grid(.cp = (1:50)*0.01)
```

Then cross-validation was performed to validate the parameters for the CART tree.

```
train(popular_video ~ ., data = trainSparse, method = “rpart”, trControl = fitControl, tuneGrid = cartGrid)
```

The output showed the accuracy for different values of cp. The final value used for the model was cp = 0.02. A new CART model was created using this value of cp found through cross-validation.

```
USvideosCV <- rpart(popular_video ~ ., data = trainSparse, method = “class”, control = rpart.control(cp = 0.02))
```

Next, predictions were made using this new model:

```
predictCV <- predict(USvideosCV, newdata = testSparse, type = “class”)
```

A confusion matrix was then built to compute the accuracy of the new CART model on the testing set:

```
table(testSparse$popular_video, predictCV)
```

The accuracy of the cross-validated CART model is 0.6548 or about 65%, which is the same as the accuracy for the original CART model. Although cross-validation is used to ensure that a good parameter value is selected, which will often significantly increase the model’s accuracy, in this case this did not hold true. The accuracy for the cross-validated CART model did not increase. Therefore, this k-folds cross-validation method is confirming that a smart parameter value has already been applied to the original CART model.

The CART model has shown improvement over the baseline model (65% accuracy for the CART model compared to 63% accuracy for the baseline model). However, the CART model did not perform better than the logistic regression model (65% accuracy for the CART model compared to 71% accuracy for the logistic regression model). This may be due to a limitation of the CART model in that these models do not perform well on smaller data sets. Although one may argue that this data set is not particularly small, the sparsity of the data did not help.

### *The random forest model*

The random forest model is similar to CART and was designed to improve the prediction accuracy of CART. This model works by building a large number of CART trees.

```
USvideosRF <- randomForest(popular_video ~ ., data = trainSparse)
```

With the random forest model built, predictions were made on the training set.

```
predictRFTrain <- predict(USvideosRF, newdata = trainSparse)
```

A confusion matrix was then built to compute the accuracy of the random forest model on the training data:

```
table(trainSparse$popular_video, predictRFTrain)
```

Using the training set, the random forest model has an accuracy of 0.9189 or about 92%.

Next, predictions were made on the testing set:

```
predictRF <- predict(USvideosRF, newdata = testSparse)
```

A confusion matrix was then built to compute the accuracy of the random forest model on the testing data:

```
table(testSparse$popular_video, predictRF)
```

Using the testing set, the random forest model has an accuracy of 0.9064 or about 90.6%. This is significantly better than the CART model, which only had an accuracy of 65% and the baseline model, which only had an accuracy of 63%. The random forest model also showed improvement over the logistic regression model, which had an accuracy of 71%.

The ROC curve is generated for the random forest model:

```
predictROC_RF <- predict(USvideosRF, type = “prob”, newdata = testSparse)[, 2]
```

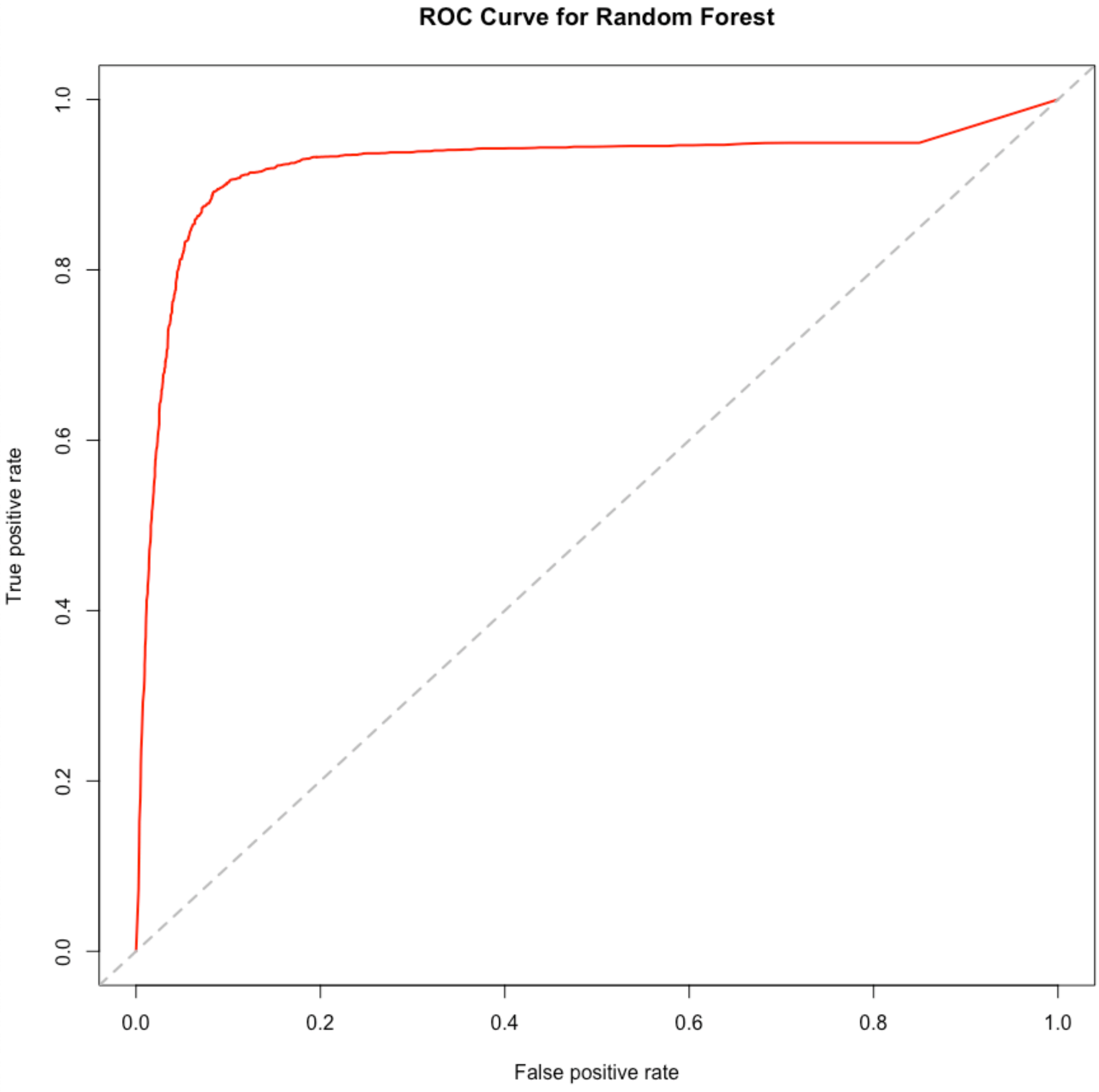
```
predRF <- prediction(predictROC_RF, testSparse$popular_video)
```

```
perfRF <- performance(predRF, “tpr”, “fpr”)
```

Then the ROC curve was plotted:

```
plot(perfRF, main = “ROC Curve for Random Forest”, col = 2, lwd = 2)
```

```
abline(a = 0, b = 1, lwd = 2, lty = 2, col = “gray”)
```



To interpret this ROC curve, recall that perfect classification happens at 100% True Positive Rate and 0% False Positive Rate. Thus, perfect classification happens at the upper left-hand corner of the graph. The closer the red line comes to that corner, the better the model is at classification. The gray diagonal dashed line represents random guess. The distance of the red line over the gray line represents how much better the random forest model is doing at making predictions than random guess. As shown by the ROC curve in the graph above, the random forest model is significantly better than random guess.

To get the AUC value on the testing set:

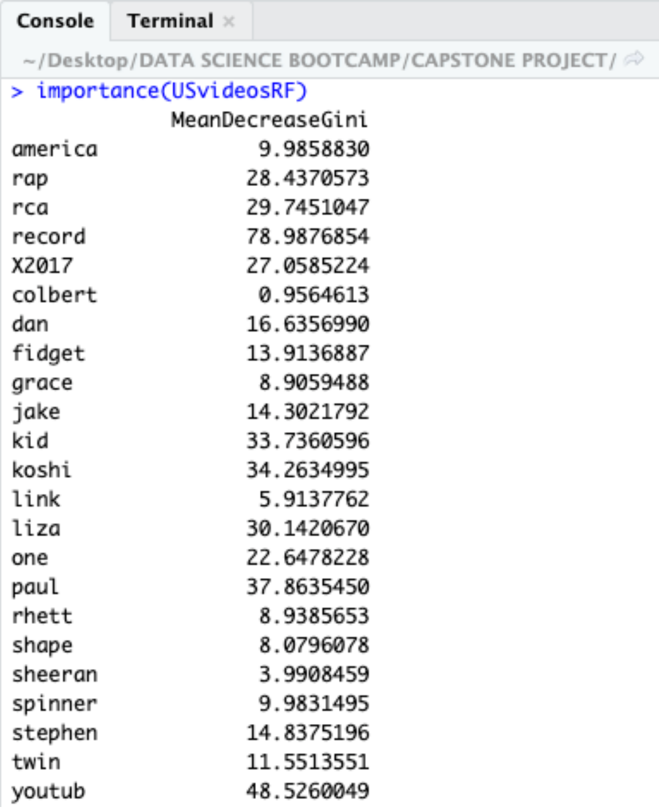
as.numeric(performance(predRF, “auc”)@y.values)

This gives an AUC value of 0.9234 or about 92% on the testing set, which means that the random forest model differentiates between popular videos and unpopular videos very well.

To see which *tags* are important in classifying *popularity* based on the random forest model, two functions were used.

The first function gives a textual representation of how important the *tag* variables are in classifying *popularity*:

importance(USvideosRF)

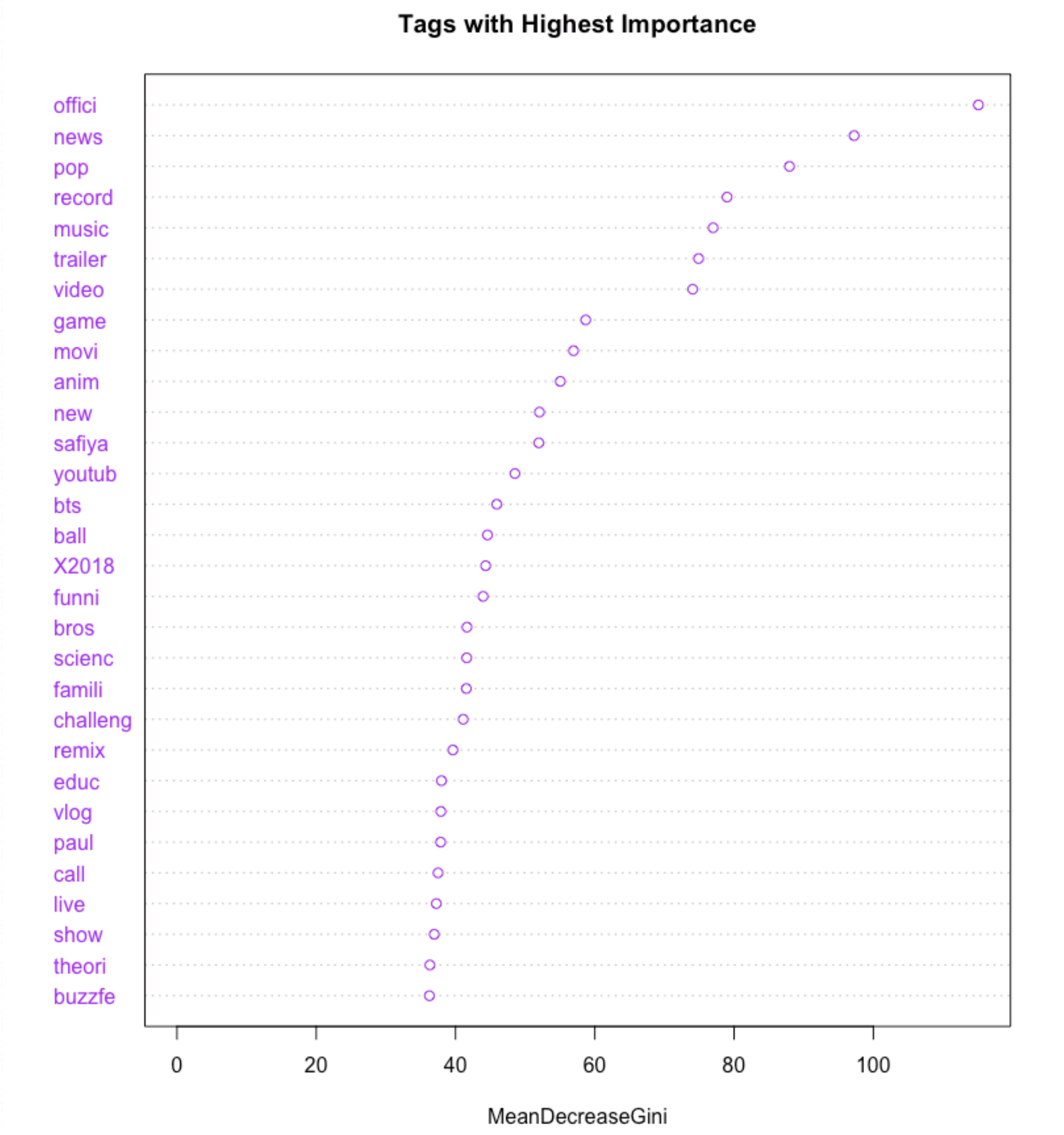


```
> importance(USvideosRF)
             MeanDecreaseGini
america             9.9858830
rap                28.4370573
rca                29.7451047
record             78.9876854
X2017              27.0585224
colbert             0.9564613
dan                16.6356990
fidget             13.9136887
grace               8.9059488
jake               14.3021792
kid                33.7360596
koshi              34.2634995
link                5.9137762
liza               30.1420670
one                22.6478228
paul               37.8635450
rhett               8.9385653
shape               8.0796078
sheeran             3.9908459
spinner             9.9831495
stephen            14.8375196
twin               11.5513551
youtub             48.5260049
```

It should be explained here that Mean Decrease in Gini is the average or mean of a variable’s total decrease in node impurity, weighted by the proportion of samples reaching that node in each individual decision tree in the random forest. This means that a *higher Mean Decrease in Gini score* indicates *higher variable importance*. Therefore, *tags* with a higher Mean Decrease in Gini number are more likely to be associated with higher ranking popular videos.

The second function gives a graphical representation of how important the *tag* variables are in classifying *popularity*:

```
varImpPlot(USvideosRF, main = “Tags with Highest Importance”, col = “purple”)
```



Based on the Variable Importance Plot, the most important *tags* included: “offici”, “news”, “pop”, “record”, “music”, “trailer”, and “video”. This intuitively makes sense given that the CART tree also showed “offici” and “record” as *tags* predicting *popularity* and that the majority of these *tags* are associated with the top two video categories of the data set, *Entertainment* (with 9,964 videos) and *Music* (with 6,472 videos).

Using the threshold and parameters established in the training set, which showed significant accuracy in the testing set as well, the final random forest model will be retrained to make predictions on all of the data.

```
USvideosRF <- randomForest(popular_video ~ ., data = tagsSparse)
```

```
predictRFFM <- predict(USvideosRF, newdata = tagsSparse)
```

A confusion matrix is created to compute the accuracy of the final model on all of the data:

```
table(tagsSparse$popular_video, predictRFFM)
```

After running the final random forest model on all of the data, the model now has an accuracy of 0.9151 or about 91.5%. This percentage falls nicely between the two training (accuracy of 92%) and testing (accuracy of 90.6%) data sets.

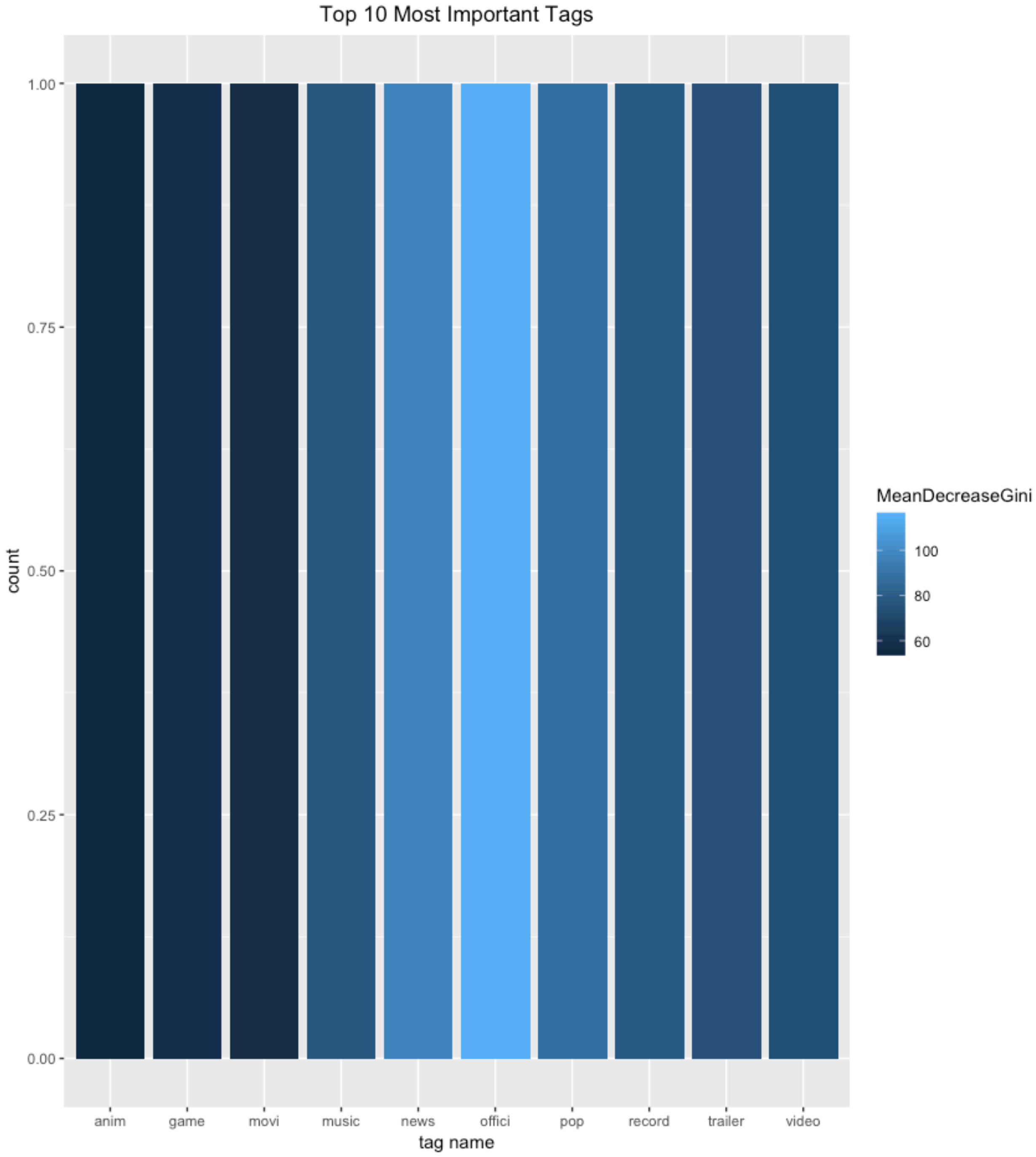
### *Findings*

Of the machine learning techniques applied to this project, it turns out that the random forest model is the most accurate of all the models tested to determine a video's *popularity* based on the *tags* used. In addition to having an accuracy of 91.5%, the model has an AUC of 92%, which means that the model can differentiate between popular and unpopular videos very well. Thus, it's fair to say that this random forest model can reasonably pick *popularity* given our data set. However, it should be noted that the random forest algorithm took over 10 hours to run. Depending on the data scientist's timeline, the logistic regression model may be the best practical choice.

According to the random forest model, the top ten most important *tags* to include are shown in the table below. It's important to note that during the pre-processing stages of wrangling the data to get it ready for the text analysis, some of the *tags* were stemmed so that only the root of the word was kept while the end was discarded. For example, "offici" could mean "official" or "officially". "Movi" could mean "movie" or "movies". "Anim" could mean "animate", "animation", or "animations".

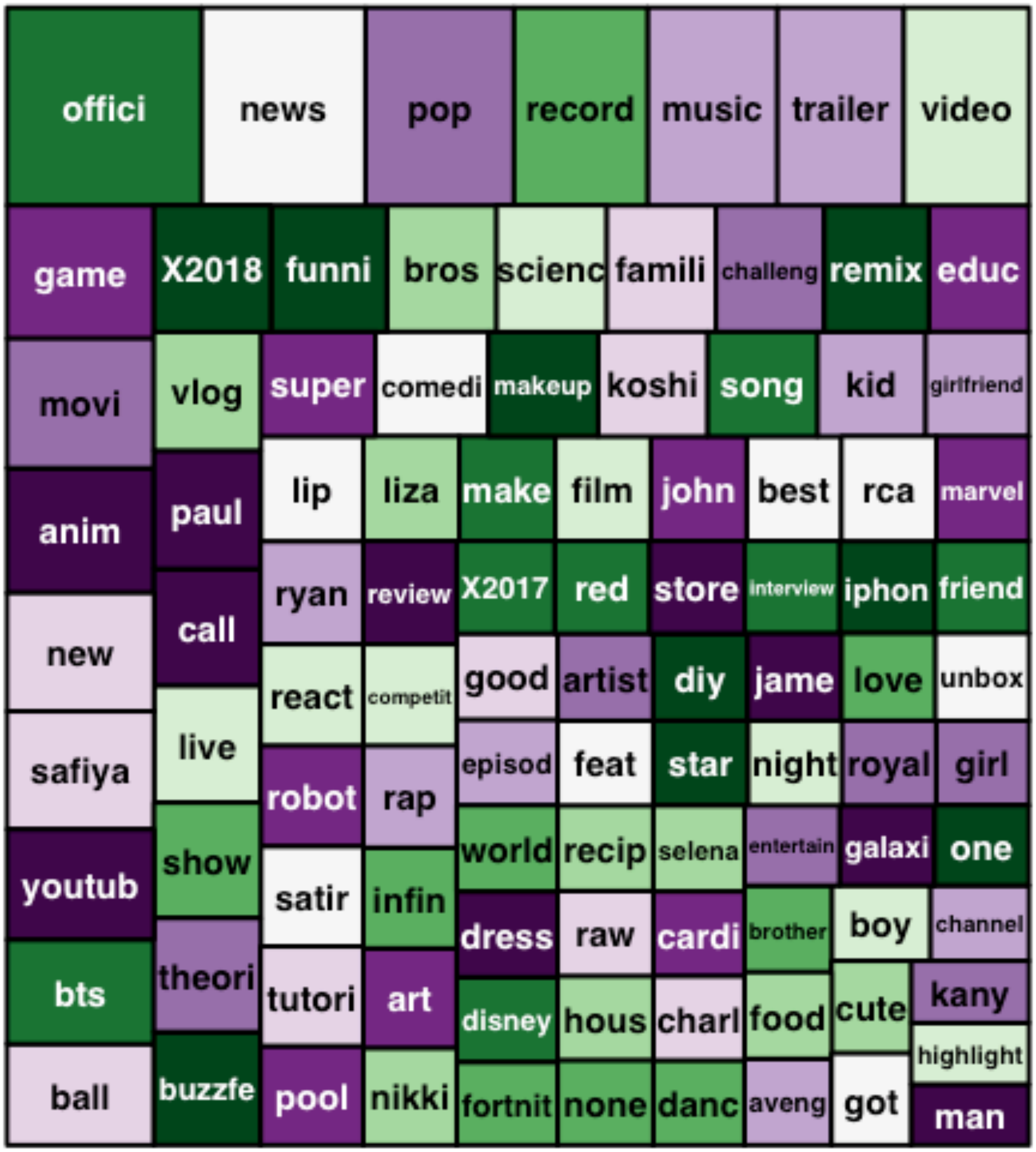
| Tag Name | Mean Decrease in Gini | Tag Name | Mean Decrease in Gini |
|---|---|---|---|
| **offici** | 115.08 | **trailer** | 74.90 |
| **news** | 97.23 | **video** | 74.05 |
| **pop** | 87.96 | **game** | 58.69 |
| **record** | 78.99 | **movi** | 56.94 |
| **music** | 76.99 | **anim** | 55.05 |

Another way to display the findings is graphically. The bar plot below illustrates the top ten most important *tags* to include with a popular video. The lighter shades of blue represent those *tags* that are the most important ("offici") while the darker shades of blue represent those *tags* that are less important ("anim"). Remember that this visual represents *tags* that are most important when it comes to predicting *popularity* across all video categories. Potential clients may find it more useful to see a bar plot created of the top ten *tags* by category. For instance, clients producing *News & Politics* videos may want to include the *tag* "news" but would most likely not find much use for the *tag* "anim", short for "animate", "animation", or "animations".



The following TreeMap shows the top 100 highest ranked *tags* based on their Mean Decrease in Gini score. Based on this visual representation of the data, it's easier to see which *tags* are most important when it comes to predicting *popularity* across all video categories. The larger the *tag's* associated box, the more important the *tag* is. For instance, "offici" has the largest box, which makes sense for a number of reasons. The "offici" *tag* has the largest Mean Decrease in Gini score, it was one of the two *tags* in the CART tree, and it is a common *tag* associated with *Entertainment* and *Music* video categories, both of which happen to represent 40% or the largest proportion of videos in the entire data set.

100 Most Important Tags to Predict Popularity



### *Future Research*

Looking at a table of the entire data set, it’s clear that the majority of videos only represent a handful of categories.

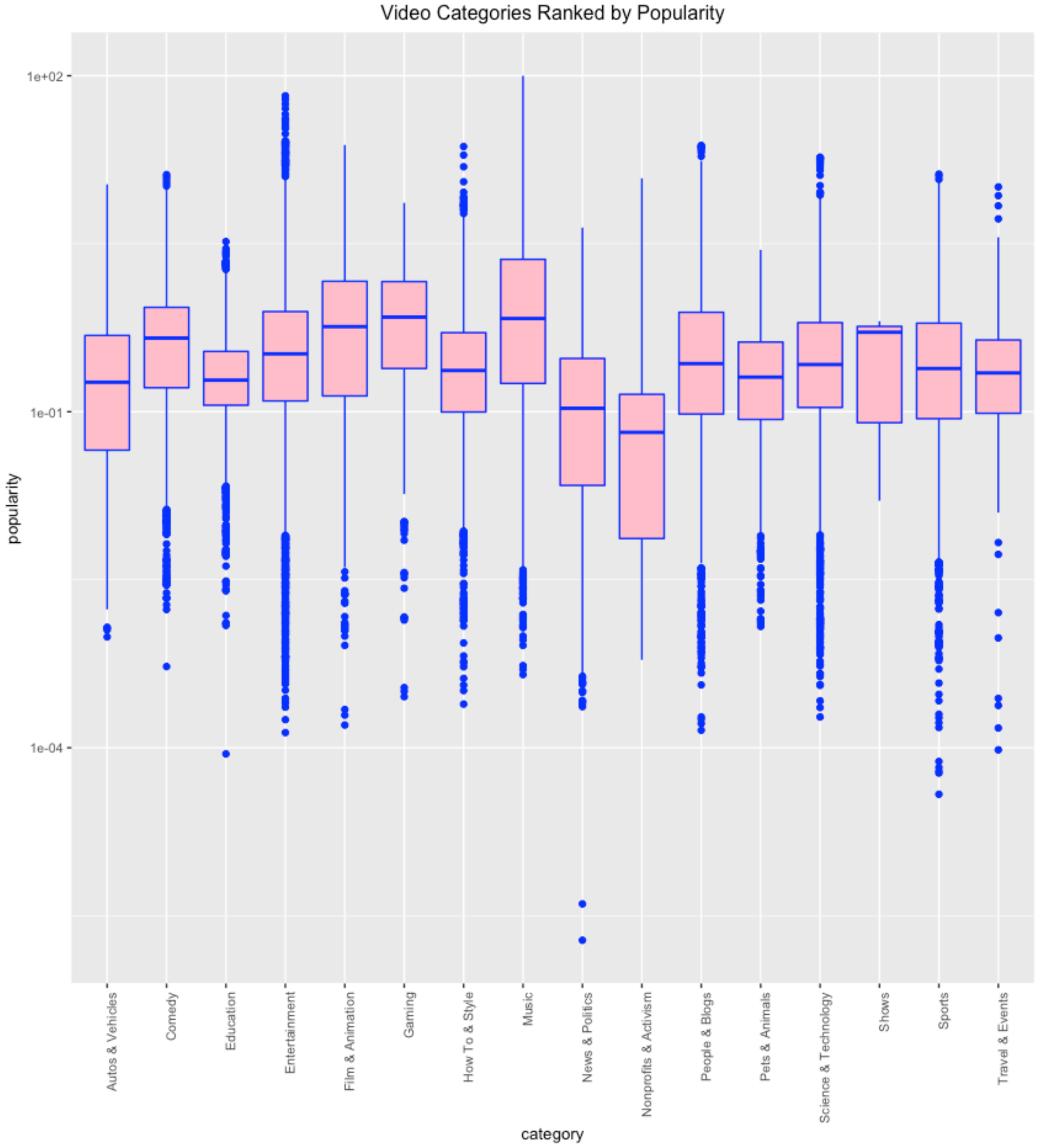
```
table(USvideos_new$categories)
```

| **Entertainment** | **Music** | **How To & Style** | **Comedy** |
|---|---|---|---|
| 9964 | 6472 | 4146 | 3457 |
| **People & Blogs** | **News & Politics** | **Science & Technology** | **Film & Animation** |
| 3210 | 2487 | 2401 | 2345 |
| **Sports** | **Education** | **Pets & Animals** | **Gaming** |
| 2174 | 1656 | 920 | 817 |
| **Travel & Events** | **Autos & Vehicles** | **Shows** | **Nonprofits & Activism** |
| 402 | 384 | 57 | 57 |

Because the entire data set was used to generate a predictive model to answer this project’s question, the findings provide limited value to clients whose videos are in categories other than the top two categories, *Entertainment* and *Music*.

Future research would want to run the final random forest model on each of the 16 video categories. Doing so would show which *tags* were determined to be most important in predicting *popularity* based on the selected category and provide more general utility for multiple clients. This is important because some *tags* are reflective of the category a particular video resides in. For example, the *tag* “record” frequently appears for videos in the *Music* category. If a client whose videos were directed towards the *Science & Technology* category used this *tag*, it would most likely not effectively target the right audience for the video’s content.

The box plot below shows the normalized *popularity* measure by category for all videos in the data set. As to be expected, the *Entertainment* and *Music* categories show the highest correlation with *popularity* such that videos within these two categories are more likely to be popular than those videos in other categories. This is similar to the relationship between video category and *view*, *like*, and *comment* count that was previously discussed in this paper. Again, it's clear that the *Entertainment* and *Music* categories received the largest amount of user engagement, making them the most popular, influencing the most important *tags* a client should assign their video when posting it to YouTube.



The good news is that it seems as though the final random forest model would work well to determine the most important *tags* by category. In fact, when applying the final random forest model to subsets of the data that only contained videos within a single category, the model's accuracy held up quite well. When applied to data for the *How To & Style* category, the accuracy of the random forest model was 0.9578 or about 96%. Likewise, when applied to data for the *Entertainment* category, the accuracy of the model was 0.9214 or about 92%.

### *Client Recommendations*

As previously stated, there are several clients who would want to know the answer to this project's question: **Which tags predict popularity for a YouTube video?**

First, it seems that clients may not need to worry too much about including multiple variations of the same *tag*. YouTube is a search engine and can use artificial intelligence to guess what word (or *tags*) might have a similar meaning and filter the results accordingly, providing more effective and relevant search queries.

Second, even though the top 100 most important *tags* seemed to be primarily related to *Entertainment* and *Music* categories, a client should still see if any *tags* in that list could apply to their video. For instance, the *tag* "new" can apply to many different topics along with the multiple meanings associated with the *tag* "live". A client should find the most generic *tags* in this list and then apply them to their YouTube video posting, regardless of whether the video they want to post is typically associated with either of these two video categories (*Entertainment* and *Music*). Also several of the *tags* refer to positive attributes like "super", "best", and "good", which can be applicable to video content for many clients.

Third, as discussed earlier, videos that were published and trended in 2017 and 2018 were included in the data set used for this project. According to the list of the 100 most important *tags* that predict *popularity*, both years were included as *tag* names. Therefore, it would be a good idea for a client to consider adding the year of the video as a *tag* name to increase the *popularity* of the video. For example, "2019" would be used as a *tag* for a video posted this year. (Remember that because the data frame was built using text analytics, all variable names were converted to appropriate names so that none started with a number. For example, "2017" was converted to "X2017" and so forth. That is why the *tags* "X2017" and "X2018" appear in the TreeMap featured on page 26.)

###