Popularity Prediction

EECS 349 Northwestern Spring 2018

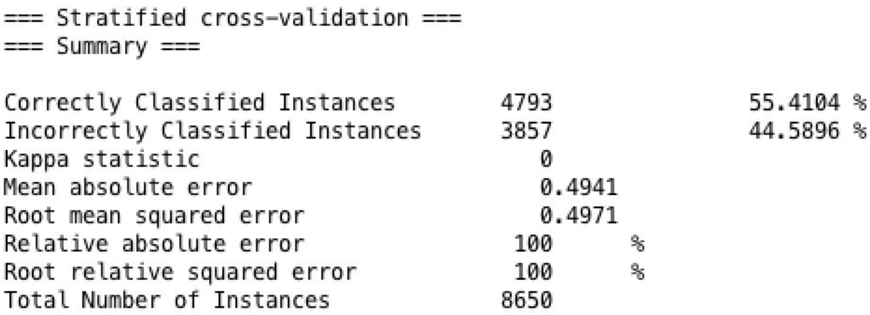
George Malty and Shankar Dutt Salwan
georgemalty2020@u.northwestern.edu | shankar97@u.northwestern.edu

This project uses machine learning to be able to predict the level of likeability of YouTube videos according to the general audience given the attributes of the video uploader's subscriber count, the video's category ID (such as entertainment versus education), number of views, number of likes and number of comments. The inputs do not include information on the number of dislikes on the video or the ratio of likes to dislikes. The goal is to predict whether a certain video's general acceptance is high or low. A video that has a high ratio of "likes" to "dislikes" is considered a "liked" video, meaning that the general audience agrees with / likes the video. Conversely, a video with a low ratio of "likes" to "dislikes" is considered "disliked," meaning that the general audience disagrees with it or that it is more controversial. This is interesting because such a correlation does not seem intuitive at first; it initially appears that a popular video may have a 10:1 like:dislike ratio or a 100:1 like:dislike ratio and that the number of views and likes would not really indicate the number of dislikes. It seems like information about the actual content and current cultural trends would decide the ratio of likes to dislikes. However, it is also really likely for either really loved or really controversial videos to be shared and go viral. Thus, the number of views in comparison with the number of likes can provide some indication as to whether a video is liked or disliked (ratio of likes to dislikes). In addition, using the subscriber count of a the video's uploader may help as well. If someone has a lot of subscribers, then maybe that video is likely to be liked, since a lot of people seem to like the uploader. This could be useful because, if such correlations are found, then the same correlations could be used to predict public opinion on other matters, even those not expressed in a popular YouTube video. For example, the algorithm could be run for an article with comparable statistics (article reads instead of video views, and publisher subscriptions instead of YouTube channel subscriptions, etc) to predict public opinion on the article's content (to predict if it is liked or disliked).

The dataset being used at hand is a combination of data collected from Kaggle and scraped using the YouTube Data API. The dataset collected on Kaggle is used to collect the video IDs of about 7000 of the trending videos in the recent past on YouTube. These are fed into a script that uses the YouTube Data API, which goes through each video to get the channel and video data for our dataset. The rest of the data was gathered from a number of the top YouTube channels as ranked by SocialBlade.com. A script was written to go through 50 videos uploaded recently from each of these channels and gather the appropriate video and channel data. Through the course of the data collection done, the final attributes scraped for each video were as follows:

the category ID of the video (such as entertainment versus education), the number of likes, dislikes, comments, and views on the video as well as the number of subscribers for the video uploader's channel. The number of likes and dislikes was used to find the like:dislike ratio for each video, for which the median was around 40. Videos with a ratio below this threshold were labeled as "Disliked" and the rest were labeled as "Liked." A few additional attributes were created to hold the rounded number of likes (to the nearest 500) as well as video views (to the nearest 50000). When the data was used for machine learning, the information for the number of dislikes and the like:dislike ratio was removed (these would make "predicting" whether a video is "Liked" or "Disliked" trivial).

This project uses a decision tree implemented using the scikit Python package (sklearn). Various datasets of varying sizes were used. In the end, to get the most representative data, they were merged into a set of about 8,000 videos. The data was randomly split into a 70% train set and a 30% test set (because there is a large amount of videos to work with). Because the line between which ratios are "liked" and which are disliked is around the median, ZeroR gives 55% accuracy. However, the decision tree was able to get an average of 83% accuracy when run 100 times using the 70/30 split for train and test data. The accuracy obtained from ZeroR can be seen in figure 1.



```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances        4793               55.4104 %
Incorrectly Classified Instances      3857               44.5896 %
Kappa statistic                          0
Mean absolute error                      0.4941
Root mean squared error                  0.4971
Relative absolute error                100      %
Root relative squared error            100      %
Total Number of Instances             8650
```

Figure 1: ZeroR accuracy for dataset obtained

Interestingly, multilayer perceptrons did not perform very well. This suggests that there are simple correlations / rules that give a good prediction accuracy. This can be rationalized by considering the logic proposed earlier: a high number of likes in comparison to the number of views may mean that a video is likely to be liked, and a high number of subscribers may mean that someone is likely to post things people like (since lots have subscribed). To substantiate this, we can see the decision tree obtained from the training in figure 2 and 3. As the trees are very large, the splits can be seen better on our webpage. A decision tree can capture such simple rules well. The decision trees implemented using the package used both the available criterion separately to find the best option. The two criterion used were information gain and gini index. We can see the resultant accuracies obtained from running our trees in figure 4.

Figure 2: Decision Tree Split for info gain

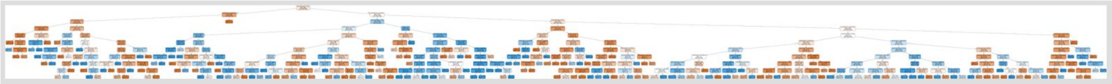
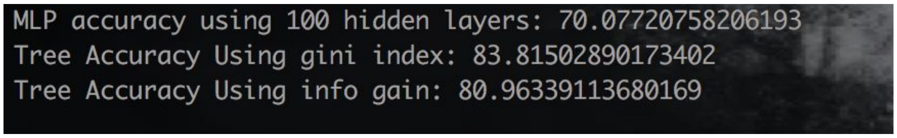


Figure 3: Decision Tree Split for gini index



```
MLP accuracy using 100 hidden layers: 70.07720758206193
Tree Accuracy Using gini index: 83.81502890173402
Tree Accuracy Using info gain: 80.96339113680169
```

Figure 4: Accuracies obtained from test set

To further improve this model, more data can be gathered that is more evenly distributed among the categories. This may make the data more representative given any new video, whereas now it may be more skewed towards, for example, entertainment videos (since those are likely to go viral and be the main content of many of the top channels according to SocialBlade.com).

This project also revealed lots of other interesting information about correlations in YouTube video data. Initially, the goal was to predict the amount of views a video on YouTube is likely to get. This may be helpful to people on YouTube who make their livings off of these views. However, after trying many different classifiers and various attributes, including using the YouTube Data API to write a script to find the history of the YouTube channel (recent average number of likes, dislikes, and views on videos as well as this information for just the video on the channel that was uploaded prior to the one being predicted), we learned that there was no strong enough correlation to even get to the nearest hundred thousand views. This made sense, as we then looked at some YouTube channels and observed the pattern of views on videos. They varied widely, and there seemed to be no distinct correlation between those that had many views and those that did not. However, the number of likes and views was heavily correlated. Given the number of likes (in addition to the other data), the number of views could be predicted with more than a 0.9 correlation coefficient. Thus, exploring these relationships more, we found that the number of likes is not at all correlated to the number of dislikes. Like:dislike ratios seemed to be just about equally probably being low, such as 10:1, or being high, such as 70:1, for example. However, the number of subscribers, views, comments, and other data we used in the end seemed to somehow predict the ratio of likes to dislikes with a high correlation coefficient using linear regression (above 0.8). This is where the logic about the magnitude of subscribers and the ratio of views to likes (and comments, as well) might suggest whether a video is “liked” or not, relating to the like:dislike ratio. According to this logic, we could throw out all other data pertaining to channel history (previous video statistics) and have similar accuracies. This is what we observed, so we threw out much of the data (the average likes, views, and dislikes of previous videos on the channel as well as this information for just the video prior to the one being examined on the same channel). This did not change the accuracy of the prediction by more than a percent or two. Exploring this further, we found the final correlations that defined our model in the end. Removing one of the remaining attributes (most notably the number of likes, views, or subscribers) did change the accuracy significantly. This confirms that these do indicate the public attitude / acceptance of the video, as the proposed logic suggested.

Division of Labor:
Both: Looking for preliminary data sets, experimenting with attributes and classifiers using Weka

Shankar: Learning and implementing sclearn to modify final classifier, creating / overseeing website

George: Learning and implementing YouTube Data API to gather data, overseeing final report