Michael Guo & Thomas Sieben

EECS 349 Final Report

December 11, 2018

Task.

Have you ever been nervous about submitting a comment on YouTube? Or curious what comments get the most likes? Our project's task is to train a ML model on a viral YouTube videos and comments dataset, and correctly classify a comment as popular or unpopular. The popular classification is based on the number of likes a comment receives. If it is above a certain threshold, which is empirically determined based on the video ID, then that comment has a popular classification.

This task's importance pertains to understanding what goes into making the most liked comment possible. We think this topic will be of interest to those involved in social media and YouTube specifically.

Data set.

We are using a dataset on Kaggle titled "Trending YouTube Video Statistics and Comments" which provides information and features on top trending videos. The attributes we are looking at are likes, video ID, and the content of the comment itself, which contains a sentence of words that we feed into our models. We also had to take into account malformed comments, which included comments with emojis or other improperly formed data. This required our dataset to be cleaned.

One important consideration of the dataset that would be valuable to approach in further iterations of this project would be the sample comment size for given videos. For example, some video IDs did not have very many comments associated with them, despite being considered “viral” videos for the number of views that they received. Either the comments on these videos had been disabled, or there just weren’t many comments (i.e., less than five).

Approach.

For our model, we implemented our code within Python3 using both nltk and sk-learn libraries. We looked at six different techniques, ZeroR, naive bayes (nltk), complement naive bayes (sk-learn), multilayer perceptron (sk-learn), support vector machines (sk-learn), linear classifier with SGD training (sk-learn), and logistic regression (sk-learn).

Our accuracies were determined by setting a threshold for each video to determine if a comment was popular or not. In determining this, we first grouped our dataframe by video_id to get the comments for each specific video. Next, we calculate the average number of likes a given comment is likely to receive for each video. We found that there were outliers in the distribution of likes, particularly when content creators comment on their own videos. This would result in a comment receiving hundreds if not thousands of likes and skewing our “average likes” metric upward. However, we determined this to not negatively impact our model as we would expect a popular comment to receive more likes than the mean value. Thus, for each video, we labeled our data according to a unique cutoff (mean value of likes per comment) to classify whether a comment was “viral” or “not viral”.

Results.

The results of running our different models produced a large difference in accuracies. Our best-performing model was the logistic regression. The ZeroR accuracy was simply guessing that a comment would not be viral, which produces a baseline accuracy that is relatively high, 87.16%. Our Logistic Regression model, however, managed to net a nearly 5% increase in performance, which is substantial improvement given that there isn't much room to grow. Interestingly, even though Naive Bayes and Logistic Regression operate in the same hypothesis space, the NB classification was the only model that actually resulted in worse performance. We are happy with the accuracy of our LR model, as based on the comment sentiment for a video, we can correctly and more accurately classify a comment as popular or not.

Model	**Accuracy/Correct Classifications (%)**
ZeroR	87.16
Naive Bayes (nltk)	60.85
Complement Naive Bayes (sk-learn)	74.42
Support Vector Machines	88.07
Multi-Layer Perceptrons	89.58
Linear Classifier (SGD)	90.92
Logistic Regression	91.82

Future work.

To further improve this project and provide a more interesting twist on it that might be exciting for users would be to implement a component using recurrent neural networks to generate new comments. These new comments would be considered popular under testing through our models and would be interesting to see how these generated popular comments might vary from video to video.

We could also look at fine-tuning different parameters of our logistic regression model to continually improve the accuracy of our classifier. During this project, we tried utilizing different logistic regression solvers (i.e 'liblinear', 'newton-cg', and 'sag') but were unable to improve the performance of the model. Since this model comes from the sk-learn package, we would be limited to using the different parameters that this function takes in for training and testing, but we could also try improving our dataset to remove examples that only contain one or two comments, for example.

Contributions.

This project was completed by Michael Guo and Thomas Sieben. Michael took charge of writing the scripts to utilize the nltk and sk-learn libraries to study the accuracy of different models on our data set. Thomas created the website and report to analyze the results that we acquired. Both Thomas and Michael discussed ideas and approaches on how to choose the dataset, clean it, and modify our approach to create a fully-fleshed project.