[ECS 175] Report

Group 8

12/07/2020

Introduction (1 page)

Problem Statement

Our project aims to predict whether a given YouTube video is classified as "trending" or "non-trending" by Google. We did this by creating our own dataset through Google's YouTube Data v3 API which contained present and historical information about a video. This problem is a binary classification problem as we are predicting one of two classes. Thus, we implemented prediction with both Neural Network and Logistic Regression models, and compared the two to find the best model overall. In addition, we expanded the problem scope from simple binary classification to multiclass classification by utilizing clustering. Although this does not directly solve our original problem, it gave us another avenue to predict a YouTube video's success.

Group Members

Data Gathering Sub-Group:

- **Contribution:** Wrote and manage script to get data from Youtube Data API
- **Members:** Ted Kahl, Rohail Asad

Data Processing Sub-Group:

- **Contribution:** Processed data from API into effective dataset
- **Members:** Phalgun Krishna, Prajwal Singh, Seth Damany

Machine Learning Sub-Group:

- **Contribution:** Created and optimized ML models
- **Members:** Cameron Yuen, Owen Gao, Theresa Nowack, Trevor Carpenter

Documentation and Web App Sub-Group:

- **Contribution:** Created web app and managed documentation
- **Members:** Josh McGinnis, Keith Choung, Nikhil Razdan, Thu Vo

Group Leader:

- **Contribution:** Organized milestones and facilitated communication
- **Members:** Nikhil Razdan

Project Links

- **GitHub:** www.github.com/nsrazdan/ECS-171-Group8
- **YouTube Data v3 API:** developers.google.com/youtube/v3/
- **Original Kaggle Dataset:** www.kaggle.com/datasnaek/youtube-new

Literature Review (1 page)

Primary Source

In looking at other sources with similar topics, we found that various studies defined popularity differently. Our group used YouTube's trending feature as our popularity metric. Another similar study from Stanford used videos' view counts instead. This study, titled *YouTube Videos Prediction: Will this video be popular?*, analyzed the topic of predicting a video's success from the perspective of a YouTube content creator. Instead of a simple binary classification "trending?" target variable like our group utilized, the researchers Li, Kent, and Zhang from Stanford divided the classification into 4 disjoint categories: non-popular, overwhelming praises, overwhelming bad views, and neutral videos. They used multi-class classification to analyse and quantify the success of a video in seeing whether it was reacted to positively or negatively. The researchers used similar attributes as our group, such as view count and duration. However, they also included an important attribute that fits in line with their 'YouTuber' point of view: time gap. This takes into account the possibility that frequent and regular uploads are favored by YouTube's recommendation/trending selection process as opposed to a sporadic upload schedule. By using the YouTube dataset from Kaggle that inspired our project's dataset, this particular research paper bears many similarities with our own.

Li, Kent, and Zhang found that extreme gradient boosting with attributes {time gap, category, and description} produced the best results with the highest F1 score out of the other methods they tried. Instead of downsampling to account for the imbalanced data, the team of researchers added class weights. They report that the highest indicators of popularity (eg. view count) are a video's category, description and time gap. When concluding their research paper, they note that the issue of overfitting remains a concern. To improve, they suggested adding more attributes such as video thumbnails and subtitles, as well as expanding the dataset for a more balanced set of videos. This particular source provides a clear framework to position our own project. Importantly, we added a large set of non-trending videos in our data to give it enough information to classify trending status.

http://cs229.stanford.edu/proj2019aut/data/assignment_308832_raw/26647615.pdf

More Related Works

The Towards Data Science article by Arvind Srinivasan, "YouTube Views Predictor," discussed a model that utilizes very interesting and unique variables our group had not thought of in its prediction model. Although the two projects differ slightly given that the "YouTube Views Predictor" model predicts the number of views a YouTube video will get and ours predicts whether a YouTube video will become trending, many of our features remain the same. However, the creators included other features such as a "Clickbait Score," a NSFW Score," and whether a YouTube video's title contained words related to common or popular YouTube genres to better determine its popularity.

https://towardsdatascience.com/youtube-views-predictor-9ec573090acb

Clustering was a method that the Machine Learning team would like to have implemented in order to discover groups of videos which would allow us to further focus the scope of our predictions of trendability. *Clustering the Unknown - The Youtube Case* by Amit Dvir, Angelos K. Marnerides, Ran Dubin, Nehor Golan did exactly that and took 100,000 video streams to cluster unknown videos based on their title and grouped them with the use of K-means clustering and the help of NLP formulations and Word2Vec. They were able to identify many unique clusters that had their own traits purely based on video title and not by any other traits given by the metadata of youtube videos.

https://www.researchgate.net/publication/332376497_Clustering_the_Unknown_-_The_Youtube_Case

Trending Videos: Measurement and Analysis studies Youtube's trending videos in terms of viewership lifecycle and other basic statistics of their content. Researchers also collected a list of non-trending videos in order to do comparative analysis between trending and non-trending videos. To distinguish the difference between trending and non-trending videos, they conducted comparative analysis on (1) the standard video feeds, which provide basic statistics of the videos and (2) video uploaders' profile. Moreover, the study used Granger Causality (GC), which provides deeper insight onto viewership pattern, to derive directional-relationships among trending-video time-series. The study concluded that there's a distinct difference between the statistical attributes of trending and non-trending videos. GC measurement confirms the directional relationship between trending videos and other videos in the dataset, and among different categories of trending videos.

https://www.researchgate.net/publication/266262149_Trending_Videos_Measurement_and_Analysis

Dataset Description (0.5 pages)

Our dataset consists of data corresponding to two types of Youtube videos: "trending" and "non-trending". The trending videos are the 200 videos designated as "trending" within the United States by Youtube at the time of the query at 12:54 AM on November 9th. The "non-trending" videos are 23,035 arbitrary videos returned by time-bounded search queries within the same time interval as the trending videos. The columns of the dataset can be divided into two categories: "initial columns" and "update columns". Initial columns were set when the videos were first retrieved, and are as follows:

- **Title, Description, Tags:** String values set by the video creator for this video
- **View Count, Likes, Dislikes, Comment Count, Ratings Disabled, Comments Disabled:** Popularity metrics recorded by YouTube for this video.
- **Ratings Disabled, Comments Disabled:** set to True if like and dislike count/comment count, respectively, could not be found.
- **Duration, Published At (date published), Dimension, Definition, Category Id:** Information describing the video. The Duration and date published are in ISO format.
- **Channel View Count, Channel Subscriber Count, etc.:** Popularity metrics and other data about the video creator's YouTube channel
- **Time Retrieved:** A timestamp set when the video was retrieved. In ISO format like other dates.
- **Is Trending:** Our target attribute. "True" for trending videos and "False" for non-trending.
- **Video Id, Channel Id, Thumbnail Image:** Unique values with no use to our model.
- **Update columns:** consist of updated popularity metrics, along with a timestamp for each update. There are eleven sets of update columns, each recording the current view count, like count, dislike count, and comment count at the time of the update.

The group initially started with a dataset of 200 trending videos from Kaggle. But, a dataset of only positive samples is unsuitable for binary classification, so we decided to also gather a dataset of non-trending videos. We understood that the result of a YouTube search is not an unbiased random sample, so we tried to compensate by gathering as many videos as possible by searching repeatedly within narrow time intervals. 21,000 non-trending videos remained after removing those with missing values and duplicates. This made for an imbalanced dataset, so we adjusted the balance of non-trending videos with downsampling. Update columns and later trending datasets were gathered mainly to support a prediction of a video's future popularity, which did not end up being our primary goal.

We decided to calculate the engagement rate by adding the number of likes and number of comments and then dividing that by the total number of subscribers. We removed columns such as the video id, thumbnails as those columns are unique for every video and do not contribute to training the model. To keep the metric of the duration of a video consistent, we converted the duration into seconds. We kept string columns such as description and title in the processed dataset, as they could be used for sentiment analysis and can assist in prediction.

Proposed solution and experimental results (4-5 pages)

Proposed Solution

As our initial task was one of binary classification, checking if a YouTube video is trending or not, we utilized primarily the YouTube definition of a "trending" video. This way, we could theoretically predict if a video would make it onto the trending chart that YouTube chooses on their own. In order to accomplish this, we first built two models, one logistic to predict the class that a video would fall into and the other a neural network.

Logistic regression is a good way to perform classification using probabilities and it is less sensitive to the prediction threshold and any outliers that may be in the data than linear regression is. We experimented with various combinations of hyperparameters, such as the 'newton-cg', 'liblinear', 'sag', 'saga', or 'lbfgs' solvers and the 'l1', 'l2', or 'elasticnet' penalties while fitting our model. We found that the Limited-memory Broyden–Fletcher–Goldfarb–Shanno (LBFGS) solver with l2 penalty produced the best model with the highest accuracy (about 0.8451883). Like the Newton method, LBFGS gives more accurate results, because it uses both the first and second derivatives of the likelihood function when updating the model's weights, but it only uses an approximation of the second derivative, making it less computationally expensive than newton-cg. L2 regularization prevents any of the weights from getting too large, giving too much influence to a single attribute and leading to potential over-fitting, while also stopping any of the weights from being minimized all the way down to zero (feature selection).

Solver	Penalty	Accuracy	True Negatives	False Positives	False Negatives	True Positives
newton-cg	none	0.8368201	166	12	27	34
	l2	0.8451883	166	12	25	36
liblinear	l1	0.8368201	165	13	26	35
	l2	0.8410042	166	12	26	35
sag	none	0.8451883	167	11	26	35
	l2	0.8451883	174	6	30	29
saga	none	0.8451883	167	11	26	35
	l1	0.8410042	167	11	27	34
	l2	0.8410042	167	11	27	34
	elasticnet, l1_ratio = .1	0.8410042	167	11	27	34
	elasticnet, l1_ratio = .5	0.8410042	167	11	27	34
	elasticnet, l1_ratio = .9	0.8410042	167	11	27	34
lbfgs	l2	0.8451883	166	12	25	36
	none	0.8451883	167	11	26	35

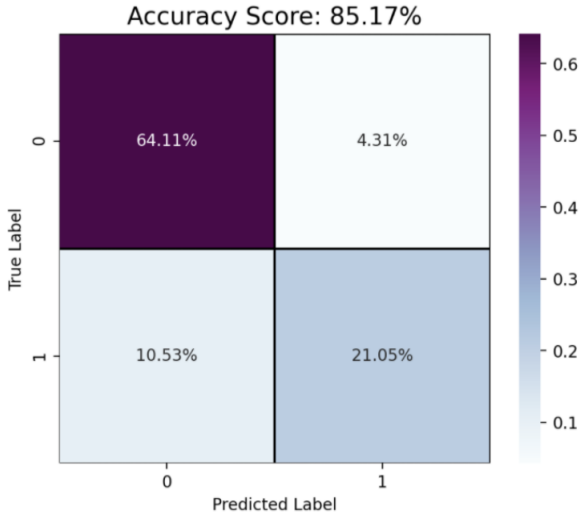
From here, we still faced a binary classification problem, but wanted to try fitting it to a more complex model to see if we could obtain better prediction results. Artificial Neural Networks (ANN) can model much more complex relationships than logistic regression can. The model with the greatest precision that we found was a five-layer ANN with thirty-six nodes in the input layer, twenty nodes in the first hidden layer, thirty in the second hidden layer, twenty in the last hidden layer, and a single node in the output layer for our binary classification of trending or not trending. We used the Adam optimizer because it uses separate

learning rates for each weight which it updates based on both the average and the variance of the recent rate of change for that weight. This makes it a better option for handling larger amounts of data since it can adapt its learning as it goes to address the different behaviors and contributions of our different parameters. We chose to use sigmoid activation functions because the sigmoid function provides a smooth gradient that does not jump between values and produces an output value that is bound between 0 and 1 which normalizes the output of each neuron. We used mean squared error as we found that it worked best with our data set. With mean squared error, the more data used, the less error there due to the average of the sum of squares being used. The learning rate of an ANN is the rate of which controls how much to change the model in response to the estimated error each time the model weights are updated. Finding the right learning rate is key to a successful and accurate model in that a learning rate that is too low will not produce enough positives and a learning rate that is too high will produce too many positives, i.e. false positives. We found that the value that produced a balance result was a learning rate of 0.005. An epoch is defined as a one cycle through the whole training dataset. We settled on 300 epochs. The batch size is a hyperparameter that controls the number of training data samples that are used. We settled on a batch size of 200. Using different optimizers we got different values for accuracy, recall and precision. The following chart shows our findings. Again, we settled on using the Adam optimizer.

Optimizer	Accuracy	Recall	Precision
Adadelta	0.22	0.00	0.00
Adagrad	0.22	0.00	0.00
Adam	0.10	0.00	0.80
Adamax	0.10	0.00	0.79
Ftrl	0.22	0.00	0.00
Nadam	0.12	0.00	0.76
RMSprop	0.10	0.00	0.79
SGD	0.22	0.00	0.00

Experimental Results

The performance of the logistic regression model is summarized in the confusion matrix shown in the figure below. A false positive is the least likely to be generated by our model while a false negative has roughly a 10.5% chance of being generated.



The receiver operating characteristic (ROC) curve is provided in Figure 2. This ROC curve shows that our generated model performs much better than random guessing. The rates of true positives and false positives are shown for all thresholds from zero to one. Our model uses a threshold of 0.5 which may be identified on the graph when the x value is 0.0431. (reword this later) 1: Confusion matrix generated by the Logistic Regression model with a threshold of 0.5.

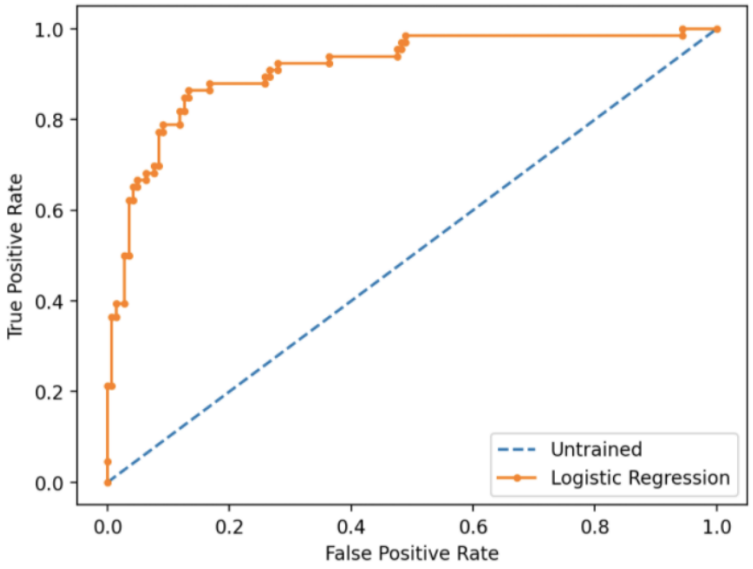


Figure 2: ROC Curve showing performance of the logistic regression model at various thresholds

The performance of the neural network model is shown in Figure 3. It can be seen that a false positive result is the least likely we are to encounter (5.74%) while a false negative is the second least likely to be encountered (7.66%).

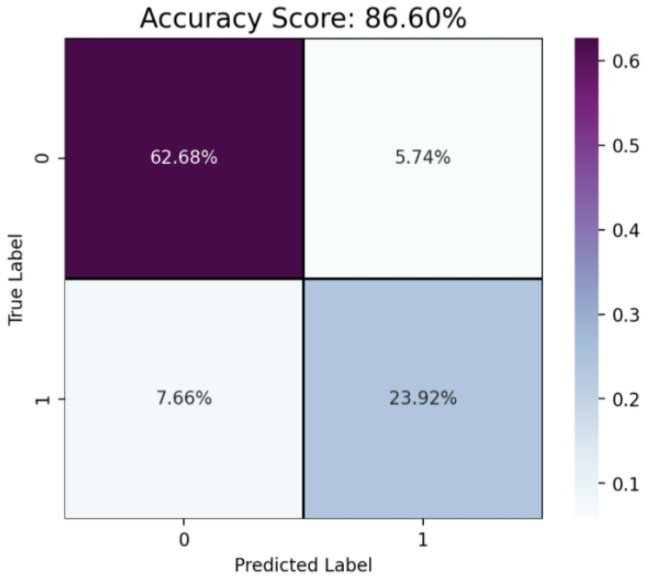


Figure 3: Confusion matrix generated by the neural network model with a set threshold of 0.5.

Clustering

While we were developing the above models, we recognized some secondary issues with our initial problem. YouTube, being a large company run by Google, is constantly gathering data on videos at a rate and level

extremely difficult to replicate. While we were able to get decent prediction values on our testing data, that was data resulting in weeks of gatherings.

We began to ask what would be necessary to accurately predict based on a snapshot of a YouTube video. Removing all data that cannot be mined at a singular point in time would leave statistics about the video at that moment such as likes, views, comments, and other attributes of the video itself such as duration, title, and description. Predicting with only these momentary attributes however results in poor statistics for our testing data:

Logistic Regression: Accuracy: 0.742857
Neural Network: Accuracy: 0.64

Confusion Matrix	Non-Trending (True)	Trending (True)
Non-Trending (LR)	138	6
Trending (LR)	48	18
Non-Trending (NN)	126	18
Trending (NN)	34	32

Clearly it is inaccurate to classify the trending algorithm with only a snapshot of a YouTube video. However, that does not mean it is impossible to classify videos in general. This is when we turned to unsupervised learning and clustering. Our flaw with predicting based on a snapshot was that we were trying to emulate an algorithm that takes into account time as a large factor, whereas we could not use that in a momentary prediction. But if we could cluster the data we did have based on momentary data into a self-defined number of clusters with the K-Means Clustering algorithm, we could predict the cluster a video would be in, and from that gather general data predictions about that video.

Using three labels instead of two, for example, on the data from before, would produce the following results:

Logistic Regression: Accuracy 0.9952380952380953
Neural Network: Accuracy 0.9571

Confusion Matrix	Label 1 (True)	Label 2 (True)	Label 3 (True)
Label 1 (LR)	87	1	0
Label 2 (LR)	0	72	0
Label 3 (LR)	0	0	50
Label 1 (NN)	83	3	2
Label 2 (NN)	0	72	0
Label 3 (NN)	3	1	46

Based on the data, we can see that adding an extra label instead of simply isolating trending videos from non-trending videos greatly increases the accuracy of a model. We utilized a WCSS graph to check the optimal cluster value if we were to cluster the data further as such and reported based on the elbow of the graph below that this value is greater than 2 clusters.

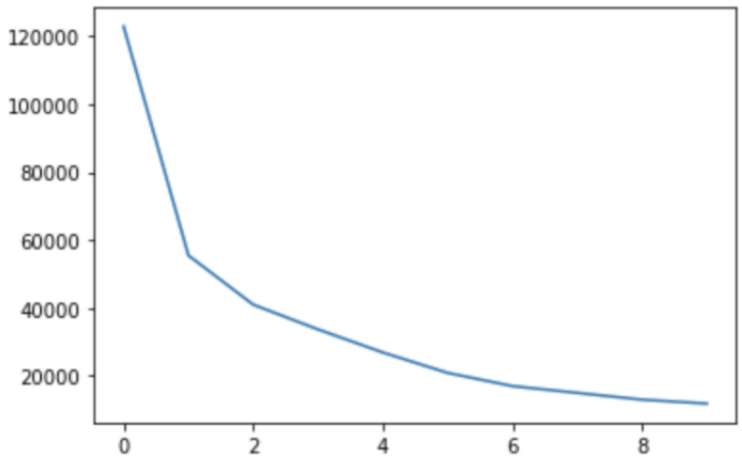


Figure 4: WCSS versus K-values of clustering the downsampled data

Conclusion and discussion (0.5 pages)

References (unlimited pages)

1. Li, Yuping, et al. Stanford, 2019, *YouTube Videos Prediction: Will This Video Be Popular?*

2. Srinivasan, Aravind. "Youtube Views Predictor." Medium, Towards Data Science, 17 Dec. 2017, https://towardsdatascience.com/youtube-views-predictor-9ec573090acb

3. Dvir, Amit, et al. Arial University & Lancaster University, 2019, *Clustering the Unknown - The Youtube Case*

4. Barjasteh, Iman, and Ying Liu. Michigan State University, *Trending Videos: Measurement and Analysis*