**YouTube Video Like/Dislike Ratio Prediction Using Comment Sentiment**

*Authors: Charlie Neuenschwander, Christopher Apton, Jiachen Zhang, Cici Chang*

**Files**: <https://drive.google.com/drive/folders/1gseykEYM9clUN3rZhAdYm3QBINivaMVP?usp=drive_link>

**Introduction**

In 2021 YouTube decided to remove the dislike metric from videos on their platform. Viewers could still dislike videos, but these dislikes could only be viewed by the video creator. In an effort to eliminate hate and spam, YouTube inadvertently created another problem. Knowing the number of dislikes a video has is crucial for certain types of videos. In the case of how-to videos or other educational resources, it could be a matter of life and death. Without knowing the credibility of a video, viewers are often left with the task of sifting through comments on the video to find out if others deemed the video positive or negative. This process is typically long and very random, but it may not have to be. By using sentiment analysis, researchers have devised methods to extract knowledge from these comments and create prediction and classification models. As the comments are unstructured data there is lots of data cleaning and preprocessing to be done. Our goal is to explore these methods for extracting knowledge from comments and create models to determine if the sentiment of comments can predict the overall sentiment of a video. In particular, we want to distinguish our results from previous in the field by focusing our attention on how-to videos, a particularly at-risk selection of videos due to the removal of visible dislikes.

**YouTube Comments Explained**

Although it seems clear that the comments of a video will inform one of the video’s quality, there are a couple complications that make this data analysis task difficult. The contents of a comment section can vary wildly based on the subject of a video and the users interacting. Some audiences are more respectful and engage in a Q&A, while others are seemingly random and filled with people “trolling.” Likewise, there are typically many comments with references to the video contents that a typical sentiment analysis model can not pick up on. In order to extract maximum value from the comments they need to be cleaned, pruned, and pre-selected for training conditions.

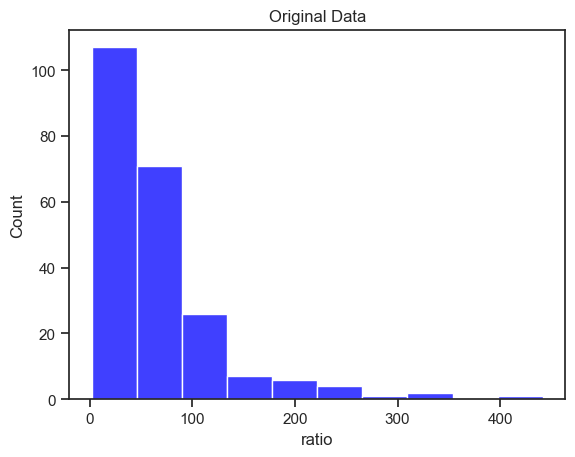
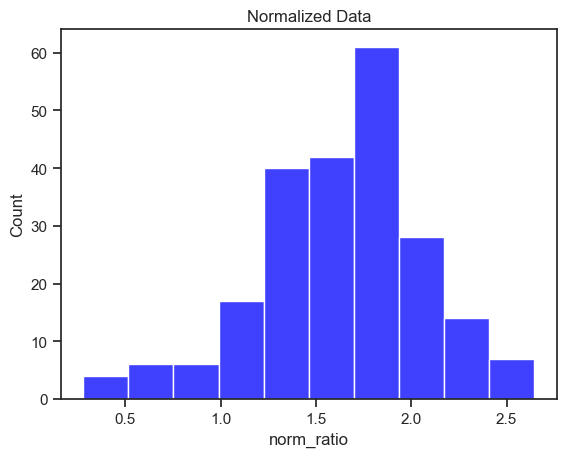
**Data Collection**

Data was collected and synthesized from two sources. The first source was from a Kaggle dataset with dislike information from thousands of YouTube videos. It is difficult to find information on a video’s dislikes since they were removed from public view, but this dataset had stored information from years prior and also stored the video id attribute which would prove to be helpful later on while aggregating. We filtered the dataset for video titles that contained “how to”, and extracted 240 unique video ids. These ids were used in the next data collection process, using YouTube’s API.

The second dataset was obtained from YouTube’s API. The YouTube API is run by Google which allows for the extraction of video information such as title, publication date, and comments on their website. Using python we iterated through the list of video ids and extracted comment information for each video. This included text itself, the number of likes a comment had, and whether or not it was termed a “parent comment”. Essentially, whether or not a comment was the first in a chain. There were 15 videos which no longer existed on the platform, but the rest of the videos yielded a total of over one million comments from 225 videos. We ran into an issue with a daily API cap so we split the task of collecting data for each member and eventually merged the data back together to provide us enough data for the model.

**Data Manipulation and Exploratory Analysis**

After extracting the necessary data we began to clean the data. Firstly, we analyzed the Kaggle dataset. We feature engineered the ratio column, which contained the true like ratio of videos (number of likes divided by the number of dislikes). We found that the distribution followed an exponential curve. Because of this, we did a log normalization of the ratios of each video to improve the distribution. Our bi-variate analysis, on the hypothesis that the view count and like ratio attributes would be positively correlated because more views typically mean a video is well received by the platform’s users and algorithm. Upon analysis we observed a slightly positive correlation. Combining these two variables together with a video’s comment sentiment should increase the ability for a model to predict. In this dataset, there are other useful columns which can be used such as tags, description, or comments. Tags has the tags of each video which the author put on. Description is the description of the video. The Comments column are not the comments from the comment section but comments from the author. Below are two histograms where one is the ratio column before and after transforming.



Next, we looked into the comment dataset. The dataset contains diverse information ranging from basic video and commenter details to intricate aspects of the comments themselves, such as publication dates and comment threads. Looking into the comment dataset, most of the effort goes into pre-processing the data for modeling. Some key features we need for our model include the content of the comments, number of comments, their like counts, and a distinction between parent and reply comments.

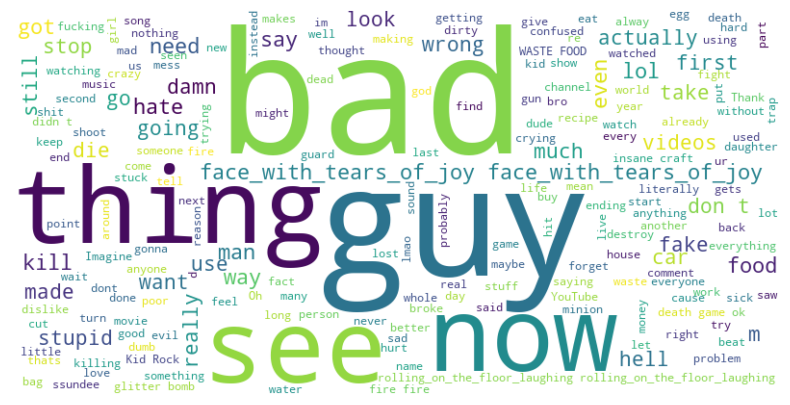
We had to do some research to find the best way to address emojis either by removing them, turning them into text data, or even running pre-trained emoji deep learning models to determine the best meaning for each sentence [2]. We decided it would be best to perform the second option and utilize the emoji library in python to extract the word definitions for each emoji such as “smiling face”, “face with sunglasses”, etc.

We continued to pre-process by sorting the data and dropping the channelId column since it’s not going to be relevant for our use case. Next, since we can see if there’s a parent comment or not based on if the parentId exists, we created a new column ‘isParent’ which returns True or False. We focused on parent comments as they provided more direct and unambiguous insights into viewer sentiments compared to the often tangential and complex nature of reply threads. Then, we dropped the rows which aren’t parent comments (isParent is False). Next, we created an isPopular column which returns true if there are more than 2 likes so we can ideally put more weight on the well-received comments.

The next steps are to prepare our data for analysis where we use a transformer model. First we do lemmatization from the comments column in the dataset. Next we obtain the stop words and remove them which leaves us with a word token column which each row contains an array of each word reduced down to its base form and removed stop words. Next, we can drop the rows without any word tokens. We use the glove-twitter-50 embedding model from HuggingFace which was trained on 2B tweets, 27B tokens, 1.2M vocab in order to convert the words into vector embeddings. The model could not interpret the ‘@’ symbol in the comments so we had to remove it from each comment in the dataset. With our cleaned comment text data, we converted it into vector embeddings. Unfortunately this method doesn’t work well in long sentences since the results will all just blend right in. In hindsight, a better approach would have been to either use PCA to reduce dimensions or just use sentence embeddings directly from another transformer model which can.

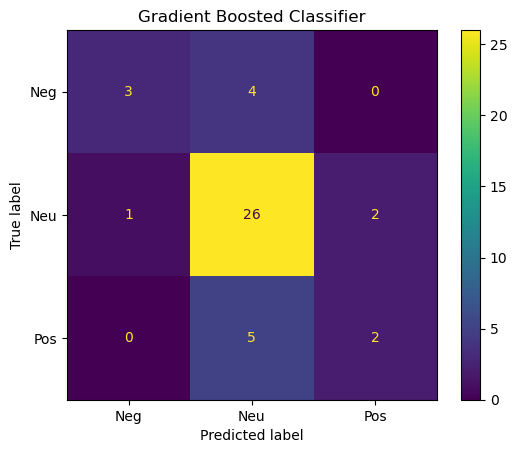
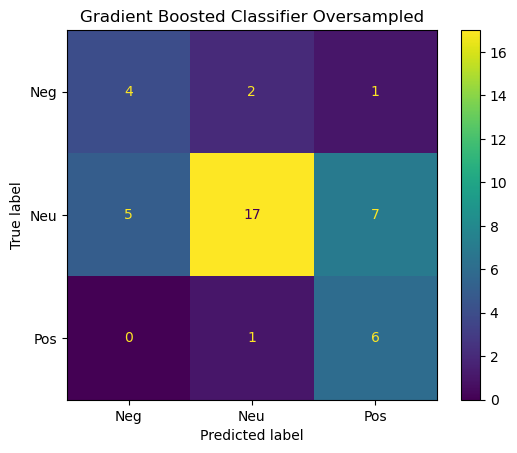
*Sentiment Analysis*

We also looked into the Sentiment Analysis of each videoId by taking the average sentiment of each comment in the video. Using the cleaned text from the previous dataset, we chose to use the python NLTK library to load in a pre-trained sentiment analysis model called vader\_lexicon. Our chosen model categorized comments into one of three categories: Negative, Neutral, Positive. Afterwards, we conducted analysis to ensure the model performed its job correctly by separating common words in negative and positive comments using two separate word clouds. Below the word cloud to the left corresponds with positive comments, whereas negative comments are on the right side.

It’s interesting to observe that we can see some obvious inconsistencies such as where emojis with face\_with\_tears\_of\_joy is considered bad when in reality it’s usually for something funny or a good thing. Also, how that guy is considered a negative comment on the word cloud. In reality, words can mean different things in different contexts, so this tool is a great starting point, however, it fails to discern the multiple interpretations of words. 

**Experiments**

*Classification*

After finding the sentiment of the comments in each video we wanted to explore the relationship between comment sentiment and direct user feedback mechanisms (likes/dislikes). We decided to first create a Classification model. In order to do this we had to prepare our data. First, We aggregated the sentiment score of each comment together using the groupby function in python. This enabled us to find positive, neutral, and negative comments in each video and then find the ratio. Then, the ratio of likes to views on each video was calculated for use as a predictor variable. If we know the views and likes of a video we can sometimes infer the overall ratio based on other videos. We then transformed the normalized like ratio value from continuous to categorical. This split was about 25,50,25 due to the nature of the normalization. Using the ratio for the three predictor variables we created and normalized like ratio for each video we developed a train and test dataset. On this dataset we used 80/20 stratified train test split to test on a Decision Tree, Gradient Boosted Classifier, and Random Forest Classifier. The most successful of these models was a tie between GBC and the RF Classifier at 72% accuracy. One of the common problems we encountered was low recall on negative sentiment videos. Due to the structure of the data, majority neutral videos, the classification model mostly categorized videos as neutral. It is important to our objective that we can identify videos appropriately, not just label them as neutral. To address this bias we implemented upsampling with SMOTE for the positive and negative comments. With an even sample size for each we ran each model again and received some interesting results. The best performing model using this method was the Gradient Boosted Classifier with 63% accuracy, but the recall for positive and negative comments was much improved across the board. 

The loss in accuracy may have been too much to warrant such changes, but it was clear that adjustments to the data could yield better results. Even if our predictor variables did work, having only 220 samples was limiting. It remains to be seen whether this method of determining a video’s like ratio based on comment sentiment could be a functional model or not. All things considered, the sentiment scores performed alright, but there seems to be other qualitative factors that contribute to a comment section’s sentiment other than the overall quality of a video.

*Regression analysis*

We then designed a regression model on video’s like/dislike ratio. The attributes we used are the generated features of the comments. Here are the predictor features.

1. Word vectors: we transform comment sentences to word tokens (after lemmatization and removing stop words). Then we used pre-trained Word2Vec models transforming each word in the token list to fixed length vectors. Then we find the average vector of the words in a sentence together as the feature of the sentence. E.g.

* Original sentence: 'This was weirdly interesting and informative'
* Word tokens: ['was', 'weirdly', 'interesting', 'informative']
* Word vectors: [0.00691067, 0.39791665, -0.23997347, -0.31321, …, 0.58683]

1. Word sentiments: we generate the sentiment polarity and scores of each sentence, using NLTK’s sentiment analysis basic model. e.g.

* original sentence: 'This was weirdly interesting and informative'
* generated sentiments: [{'label': 'POSITIVE', 'score': 0.9998146891593933}]

1. Other attributes: sentence length (by word), like count (per comment)

In theory, these predictor variables make sense in predicting a video’s like/dislike ratio. Although a generally-liked video can possibly have negative comments, those negative comments might not receive a large number of likes. Only the comments that are liked most have the critical impact on how the video is liked in common sense.Then we did regression on normed like/dislike ratio of the video, using those predictor features. We tried 4 models:

Support vector machines/neural networks/Adaboost Regressor/Xgboost Regressor

The parameters we are tuning are: **SVM:** C value (for regularization); **Neural Networks:** network structures, learning rates, number of iterations; **Adaboost regressor:** base classifier parameters (e.g. max depth of decision tree, minimal split, minimum leaf samples), number of iterations, learning rates; **Xgboost regressor:** number of iterations, learning rates. The results of best parameter settings (shown by RMSE score)：SVM: 0.506; NN: 0.473; Adaboost: 0.491; Xgboost: 0.453. Comparing these 4 models, Xgboost has best performance. And we incorporated an ensemble technique, using the average prediction of the 4 models, and reached a final RMSE as 0.449. Finally, we aggregated the video like/dislike ratio prediction by aggregating all the prediction value by single comments of that video and finding average. The we used Min-Max scaler on both original like-dislike ratio and predicted like-dislike ratio, and we got the final 2 maps on our best models.

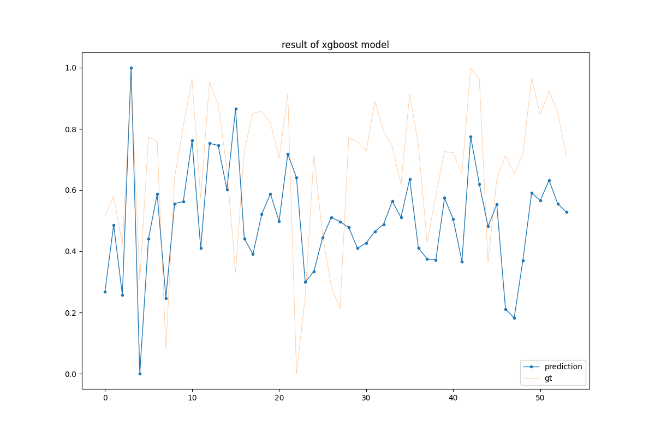
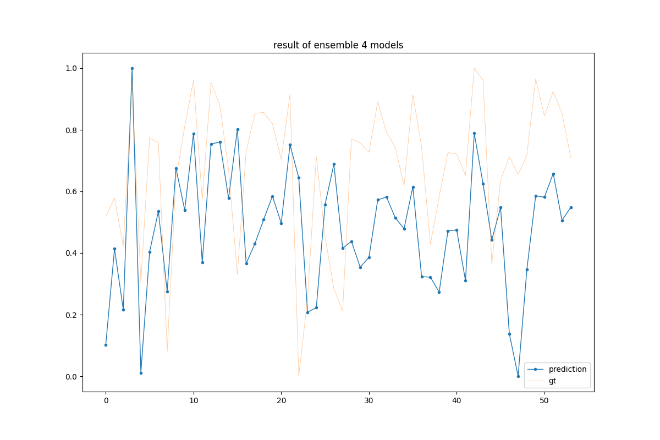


Figure. The left is the result vs. ground truth of Xgboost model, and the right one is the result of ensemble of 4 models.

From the result figure we can see that, although the prediction model can’t get very close the ground truth data, it can capture the major trends (high or low) of the ground truth. This ability of the prediction model is sensitive to small units, making it a usable model in predicting how a video is liked.

**Conclusion and Results**

Our overall results showed some very promising patterns. However, we have identified some approaches that could have produced better results using different data and different analysis techniques. Firstly, the data collection technique could be performed on an easier to control group of videos. Instead of spreading across videos from many different channels and video types with ranging comment sincerity, we could focus on one channel at a time as a group of researchers at the 2019 ICOMITEE conference showed to be very successful [1]. However, this could prove to be difficult to scale out and generalize to all videos in actual application and would require a large amount of data from a corpus that currently doesn’t exist. Additionally, the data could be more carefully cleaned of spam and non useful comments. This is a massive hurdle that many other researchers have found to taint results [3]. We only explored some ways of deleting irrelevant comments such as using the first comment in each thread and only using comments that received likes. If one could devise a method to more carefully select comments that actually provide information on a video’s true sentiment there could be a reduction in size as well as an increase in accuracy. Lastly, creating a corpus of comments beyond 240 videos to the thousands could significantly aid in true sentiment interpretation. We weren’t able to improve performance and trim videos with few comments or try undersampling the data because of this limitation. At the end of the day YouTube has many mechanisms other than the dislike button to disincentivize harmful videos. These methods are built into their algorithms and into the functionality of blocking and reporting harmful content. Still, there is a massive amount of unstructured text data that if utilized properly, could provide invaluable insights to YouTube and its users.

**References**

1) A. N. Muhammad, S. Bukhori and P. Pandunata, "Sentiment Analysis of Positive and Negative of YouTube Comments Using Naïve Bayes – Support Vector Machine (NBSVM) Classifier," 2019 International Conference on Computer Science, Information Technology, and Electrical Engineering (ICOMITEE), Jember, Indonesia, 2019, pp. 199-205, [[link](https://ieeexplore.ieee.org/abstract/document/8920923)]

2) P. Chandra and U. Prasad, "Classification of Emojis using Artificial Neural Network and Natural Language Processing," 2021 8th International Conference on Computing for Sustainable Global Development (INDIACom), New Delhi, India, 2021, pp. 205-212. [[link](https://ieeexplore.ieee.org/abstract/document/9441099)]

3) I. Lorentz and G. Singh, “Sentiment Analysis on YouTube Comments to Predict YouTube Video Like Proportions,” Dissertation, 2021. [[link](https://www.diva-portal.org/smash/record.jsf?pid=diva2%3A1593439&dswid=7825)]