Unlocking Insights from YouTube Channel and Video Data

Authors: Ethan Morgan(morgan.et@northeastern.edu), Anshul Rao(rao.ans@northeastern.edu)

INTRODUCTION

Motivation and Background

The analysis of YouTube video data is of great significance as it offers an understanding of the preferences, behaviors, and opinions of the platform's users, which in turn can inform business, marketing, and research decisions. For further information, please refer to the IMPACT section.

Previous research on YouTube data has taken various forms, such as identifying the most popular videos or analyzing the sentiment of comments. Other studies have focused on predicting likes based on comment information. However, the datasets used in these studies have varied in terms of the number and types of fields they contain, as well as the time frame and scope of the data. The dataset available on Kaggle for this project is relatively new, which gives it an advantage as it has not been as widely explored or studied in the past.

Dataset

As mentioned above, the dataset has been taken from Kaggle.[1] Fields pertaining to YouTube videos and channels, such as likes, dislikes, subscribers, comments, and more, are included in the dataset that were used for this project. To supplement this data, we utilized the YouTube API to add video and channel titles and descriptions to the dataset.

METHODOLOGY

Low Risk

The majority of our low-risk problems and methods involved EDA and unsupervised techniques in order to more deeply understand this dataset. We did univariate and multivariate analysis of tabular features to better understand our dataset. Additionally, we used principal component analysis (PCA) to identify the key dimensions of the data and determine the number of features needed to explain 95% of the variance in the dataset. We used clustering techniques on tabular features to identify groups of channels and videos with similar content. We applied K-Means, Spectral Clustering, and t-SNE, and compared the resulting clusters. To assess the quality of our clusters, we used the Silhouette score and Calinski-Harabasz Index.

Additionally, we cleaned the textual fields by tokenizing, removing emojis, hyperlinks, punctuations and stopwords (including common words specific to the dataset like ‘please’, ‘subscribe’, ‘video’, ‘channel’, etc.), and lemmatizing (using WordNet). The *nltk* Python library was used to achieve most of the textual preprocessing tasks.

Medium Risk

For medium-risk methodology, we used tabular metadata (posting datetime, channel subscriber number, channel age, etc.) in order to predict the popularity of a new video. We framed the problem as both a regression and classification problem. For regression, we directly used the target variable ‘views/elapsedtime’. Our first model was Linear Regression. There was no missing dataset but there were many outliers. So, we capped the outliers below the 3rd percentile and above the 97th percentile. We also removed any features related to views, elapsed time or channel, encoded categorical features using one-hot encoding, log-transformed right-skewed features and scaled the values. The second model we used for regression was Gradient Boosting. Unlike the linear model, not much data preprocessing was required for this, but we used cross-validation to tune the hyperparameters. The *scikit-learn* library modules were used for regression building the regression models.

For the classification model, we framed the problem as a binary classification where a new target was created called ‘popular’. If ‘views/elapsedtime’ was greater than or equal to the 65th percentile then the video was considered popular else not. Two classifiers were built, one was K-Nearest Neighbors (using *scikit-learn*) and the other was a two-layer neural network (using *PyTorch*). There was only one hidden layer with 12 neurons and ReLU was used as the activation function. The preprocessing done for both of them was the same as the one done for the linear model.

Apart from this, we performed clustering on our textual features, specifically the title, channel description, and the video description. In order to perform clustering on our textual features, we had to convert the fields to numerical values. First, some preprocessing was performed, including removal of non-alphanumeric characters, tokenization, and removal of some stop words. Next, we tried three methods of embedding the fields into vector

space. First, we tried embedding each word using word2vec, and taking the average of each word for a stand-in sentence embedding. Next we tried InferSent, a method of applying pre-trained transformers to represent sentences in vector space. Finally, we tried SentenceBERT, which uses a pre-trained BERT language model to embed sentences in vector space. This last method worked best, so the results will use SentenceBERT.

High Risk

Our high-risk proposition was creating a recommendation system for videos. We intended to build a vector space model using cosine similarity. This proved to be an excellent failure due to the limitations of the dataset, be it the size or coverage of its fields disqualified this idea. The textual data was noisy and incomplete (since the requests to YouTube API were rate-limited) and many video descriptions were in languages other than English. Also, because we do not have subscriber specific info, we could not venture into user-based collaborative filtering or other recommender systems algorithms.

RESULTS

EDA

Most of the tabular features are right-skewed. While the videos from 2005 to 2015 are covered in the dataset, most of them are from the period 2011 - 2014. Music, Entertainment and Gaming are the most common video categories in the dataset. Channel view count was strongly positively correlated with subscriber count as can be seen in the correlation heat map in Figure 1 below. On trying to investigate this linear relation further, it seemed like the number of channel views for entertainment is on average higher than the number of views for gaming, for the same number of subscribers, as can be seen in Figure 2 below.

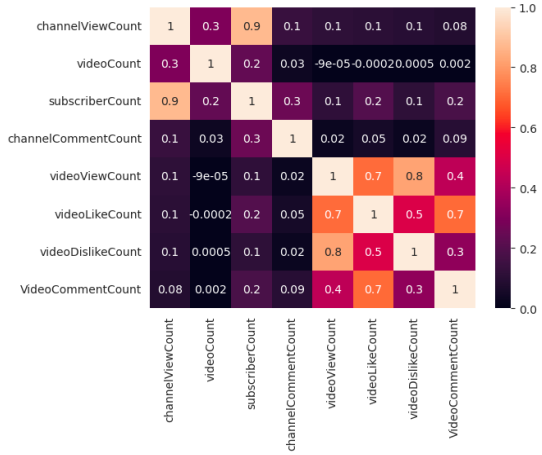

Figure 1

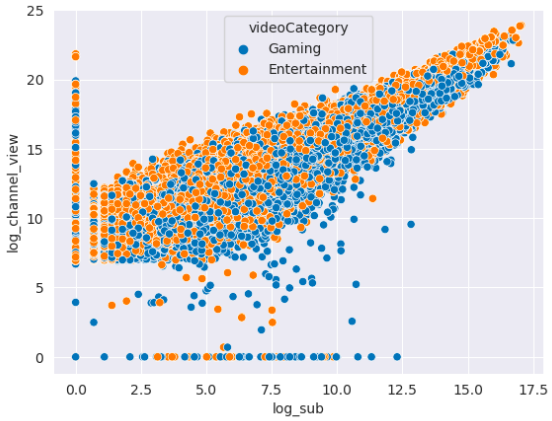

Figure 2

Clustering of Numerical Features

Our final low-risk work was to perform dimensional analysis and clustering using the numerical features in the original dataset. We first removed duplicate instances, trimmed the placeholder values and NaNs, and normalized each column to have a value between 0 and 1. We fed these features into Principal Component Analysis in order to view the dataset in a low-dimensional space. This visualization is shown in Figure 3, along with a Scree and Pareto plot. We noticed that by using only 3 principal components, we were able to explain over 98% of the variance in the dataset.

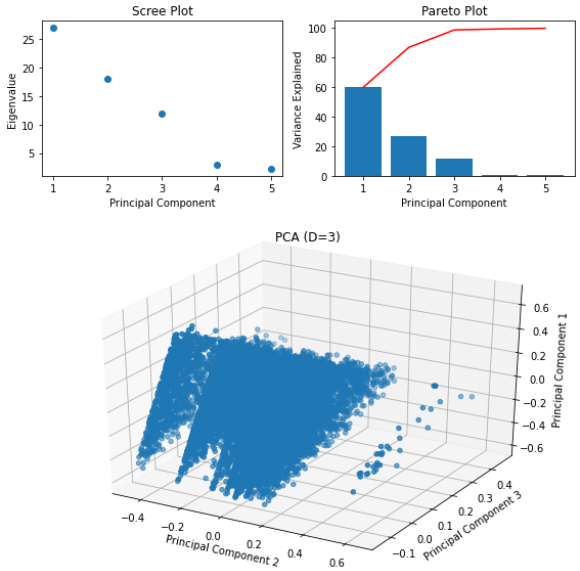

Figure 3

Next, we performed several clustering techniques on this normalized data. Specifically, we attempted K-Means, Gaussian Mixture Models, and Agglomerative Clustering using several different linkages. In Figure 4, we plot the 4 found clusters using K-means on the PCA axes found above, as well as on a t-SNE representation.

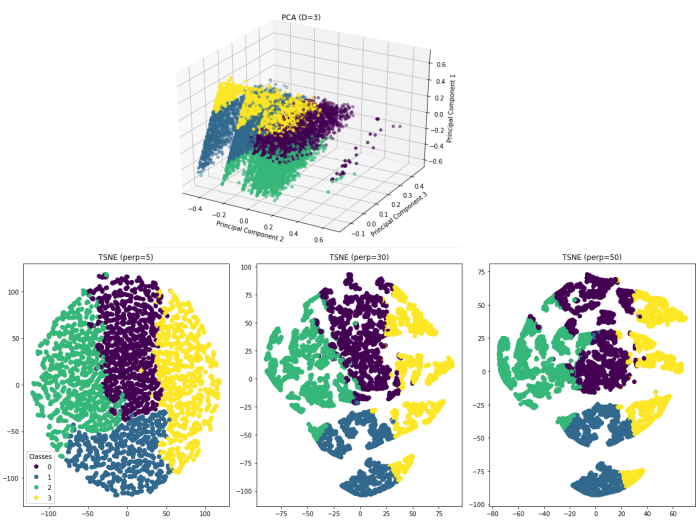

Figure 4

Predicting Popularity: Regression

The linear model gave an RMSE value of 45. On investigating further, it was concluded that the linearity assumptions were violated. For example, if we look at the Q-Q plot of residuals in Figure 5 below, we can see that there is no linearity and hence the residuals are not normally distributed.

The Gradient Boosting model performed better and gave an RMSE score of 18. On tuning the hyperparameter like 'max_depth' the score improved to 16. The *videoDislikeCount* was the most important feature in this model as can be seen in Figure 6 below.

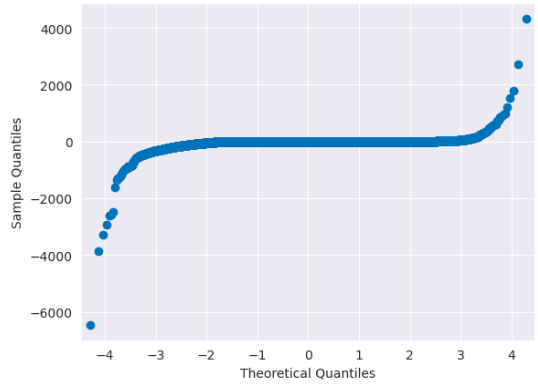

Figure 5

Predicting Popularity: Classification

For the binary classification case, the first classifier K-Nearest Neighbors gave an accuracy of 78 percent and F1-score of 74 percent. Note that since this model is computationally expensive, it was fit on a smaller random subsample of the dataset.

The two-layer neural network gave an accuracy of 87 percent and F1-score of 79 percent. Refer to the table below for a detailed overview of the scores.

Figure 7 shows the confusion matrix of the KNN model. As can be seen, we have higher precision than recall meaning that out of all the predicted popular videos, 87 percent are actually popular. On the other hand, out of all actual popular videos only 65 percent were predicted to be popular.

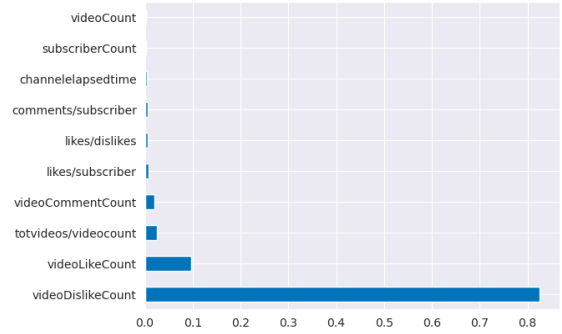

Figure 6

Model	Accuracy	Precision	Recall	F1-Score
KNN	0.78	0.87	0.65	0.74
NN	0.87	0.80	0.84	0.79

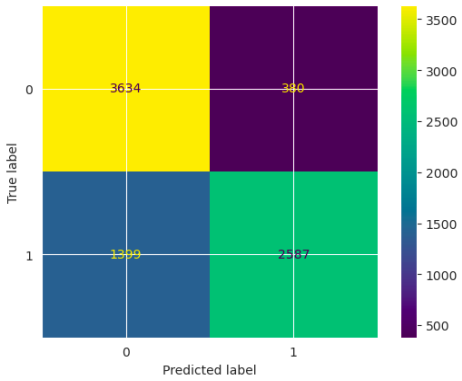

Figure 7

Clustering Using Textual Features

We adapted the clustering code from above to support high-dimensional embedding data, and additionally added some of the numerical features back in. Again, K-Means worked best on this modified dataset. We tried to fit from 2 clusters up to 8 clusters. The best results are shown in Figure 8. After fitting the clusters, we applied word clouds to the corpus found in each individual cluster in order to find similarities within each cluster. Some examples pertaining to the clusters found are also visualized below in Figure 8.

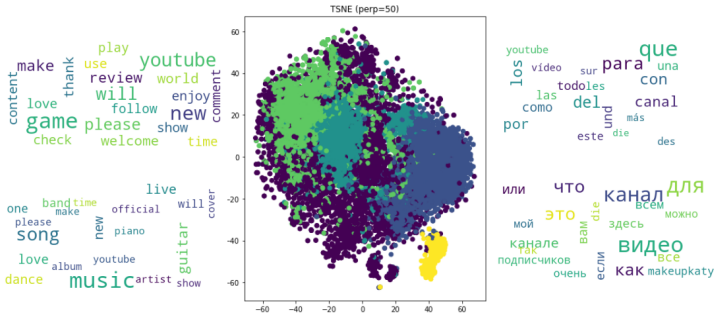

Figure 8

IMPACT

1. Marketing: Companies and advertisers can use data on the types of videos that are popular on YouTube to better target their advertising and promotional efforts.
2. Content creation: YouTube content creators can leverage the model's ability to predict popularity to guide their content creation and increase their chances of success.
3. Social and cultural trends: Analysis of YouTube video data can provide a glimpse into the social and cultural trends of the platform's users, which can be useful for researchers and academics studying the dynamics of online communities.
4. Product development: YouTube as a company can use the findings and insights to improve its product such as recommendation algorithm, video quality, etc.

CONCLUSION

Throughout the course of the project, we successfully achieved the major checkpoints that we had planned and estimated to complete. We were able to construct and evaluate models to forecast popularity, as well as perform clustering and interpret the resulting data. As a result, we have gained a thorough understanding of the dataset and extracted valuable insights from it.

For example, some interesting results were found from the textual clustering efforts. In Figure 8, we found that one cluster (purple) was just background noise with general Youtube-related terms. However, the other 4 clusters were illuminating. First, two of the clusters related to music and gaming, which are two of the largest topics on YouTube. The other two clusters highlighted the Spanish and Russian communities on YouTube. Tweaking the hyperparameters found other types of groups, for instance, the German community and food-related content. Given more time, we would like to spend more time teasing out the different communities. This could prove a useful tool for automatic tagging of video topics and categories.

There were some issues with the dataset that prevented us from completing the high-risk objective of building a recommendation system. First, the API for extracting our textual fields was rate-limited to 8000 requests per day. Given that our dataset had over 500,000 videos, many of which had been deleted since the dataset was created, we were not able to populate the entire dataset in time. Second, we found that many of the videos were not in English, which posed a few problems when trying to compare descriptions that were semantically similar but in different languages. Given a few weeks to finish downloading all of the textual columns and further develop the recommendation algorithm, we would likely be able to finish the high-risk objective.

Overall, despite the limited timeframe of this project, we made substantial progress and accomplished noteworthy results. As previously stated, there remains ample opportunity to delve further and improve the outcomes.

REFERENCES

1. Kaggle: YouTube Videos and Channels Metadata, https://www.kaggle.com/datasets/thedevastator/revealing-insights-from-youtube-video-and-channe
2. Code: Jupyter Notebooks, https://drive.google.com/drive/folders/11ovYyLhCSDa72W_v-DtC3JbCp9RWqNvY?usp=share_link
3. SentenceBERT https://www.sbert.net/
4. InferSent https://github.com/facebookresearch/InferSent
5. YouTube Data API https://developers.google.com/youtube/v3/docs/?apix=true