Trending Days Prediction of YouTube Trending Videos

Juefei Chen[*]

https://github.com/Juefei-chen/1030pj/tree/master

1 Introduction

YouTube is currently the world's largest video platform. Statistics shows that everyday people watch one billion hours of videos on YouTube; and 500 hours of video are uploaded to YouTube every minute worldwide [1]. It maintains a trending list with videos who: attract a wide range of viewers; are not misleading or cheating; show what's happening around the world; showcase a diversity of creators; and are surprising or novel [2]. In other words, these videos are of high quality and high popularity. This project focus on these videos and is aim to use machine learning methods to explore what features of them attract viewers' attention.

Two datasets are used for this project, which are crawled from YouTube API. One called **YouTube Trending Video Dataset (updated daily)** (U.S. part) is crawled from the YouTube (U.S.) trending list with an open-source code in GitHub, and uploaded to Kaggle daily [3]. In order to explore the influence of channels' information, another **YouTube channels 100,000** dataset from Kaggle [4] is merged, which includes information like channels' followers count, videos count and join date.

Many people on Kaggle have done EDA with these two datasets, some of them further predicted the videos views and likes count [5, 6] or did sentimental analysis [7].

After merging two datasets, some useless variables, like URL links and IDs for marking, are deleted together with the empty variable 'location'. However, except for continuous and categorical variables, the rest variables have some date-type and text variables, where the text variables' values are sentences or paragraph. In order to get use of them, I create two new features 'publish2trend' and 'join2publish' by calculating time differences with date type variables. For text variables, I compute the word frequency of variables 'video_title' and 'tags', then create new features to show whether a video has top 5 most frequent words in each corresponding variable.

Some videos have a couple of records in the dataset, which means they show on the list for days. I count the number of days for each video and use this value as the target variable, which is continuous and this project is a regression problem. In this case, all the other features can be used for prediction and it can also reflect videos popularity. In order to eliminate their time-series properties, for each video, I only keep the first record that it show on the list.

Then, we have the dataset we will use, including 18,069 instances and 23 variables. The variables are shown in Table 1.

Variable	Type	Description	Variable	Type	Description
view_count	int	video's views count	country	str	channel's country(categorical)
likes	int	video's likes count	followers	int	channel's followers count
dislikes	int	video's dislikes count	videos	int	channel's videos count
comment_count	int	video's comments count	join2publish	int	days from channel join to video published
comment_disabled	bool	video allows comment or not	publish2trend	int	days from video published to trending
ratings_disabled	bool	video's allows ratings or not	vtitle_'str'	bool	video's title contains 'str' or not
trending_days	int	days that video on trending list	tag_'str'	bool	video's tag contains 'str' or not
category_name	str	video's category(categorical)			

Table 1: Variables description

[*] E-mail: juefei_chen@brown.edu. Data Science Initiative, Brown University

2 Exploratory Data Analysis

Figure 1 and 2 shows the target variable distribution. It has a long tail, and most trending days are around 6.

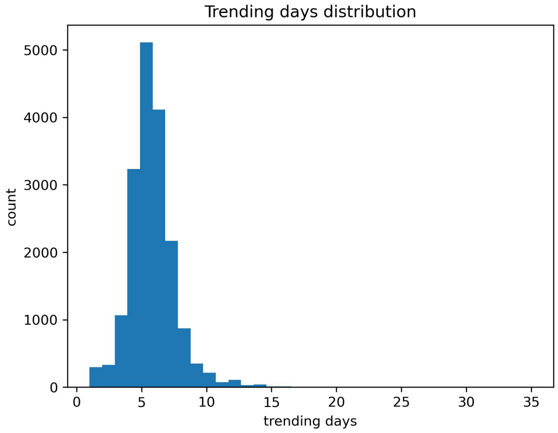

Figure 1: Trending days distribution

count	18069.000000
mean	5.485804
std	2.025103
min	1.000000
25%	4.000000
50%	5.000000
75%	6.000000
max	35.000000

Figure 2: Trending days description

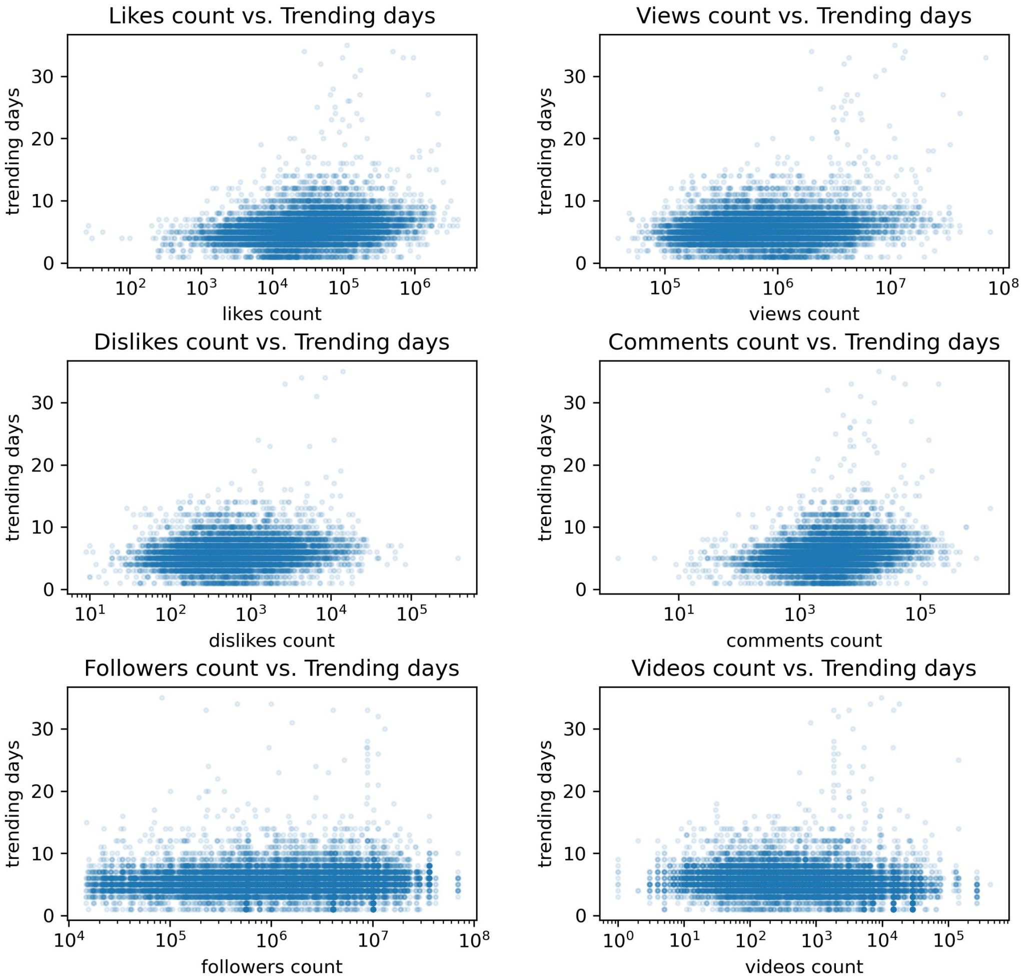

Figure 3: Video info vs. Trending days

The scatter plots of continuous variables show that some of them have slightly positive correlation with trending days, as Figure 3 shows.

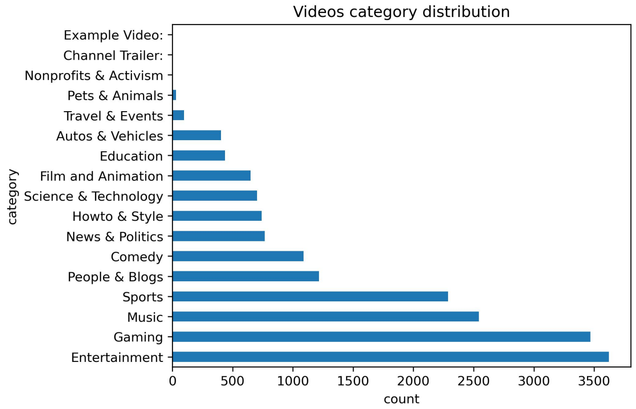

Figure 4: Videos' category distribution

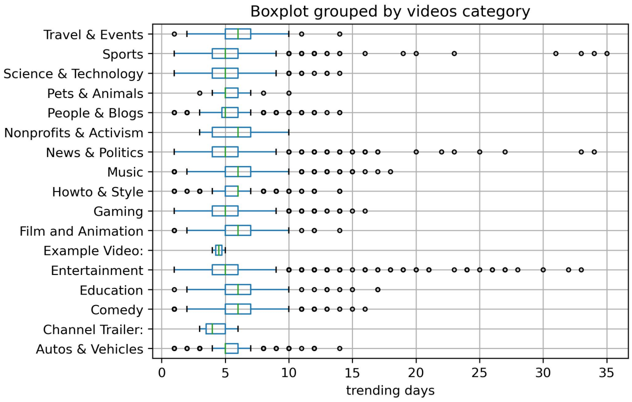

Figure 5: Boxplot grouped by videos' category

Figure 4 is the videos category distribution. We can see people like Entertainment and Gaming videos most, and there are also many people like sports and music. It seems that people tend to watch videos on YouTube more for leisure. However, from Figure 5 we can see that the Entertainment and Gaming videos' likes count are not very high, just the same as most other categories. The music category has relatively high likes count, while sports videos usually gain less likes.

3 Methods

The EDA also tells there are unbalanced distribution of 'comments_disabled' variable. 1.54% videos don't allow comments. This feature may be important to the target variable, so I use stratified KFold on my dataset according to it.

In preprocessing, I use StandardScaler() on each continuous feature because all of them don't have their upper bounds but have a long tail. For categorical features, I use OneHotEncoder() because their categories are unordered.

I write a cross validation pipeline function MLpipe_KFold_RMSE(). It firstly splits the unprocessed data to 'other'-'test' set with 80%-20%, and uses stratified KFold on 'other' with 5 folds. Then it preprocesses the data and performs cross validation, where for missing values, it uses linear regressor to do the imputation (in particular, this step is skipped when using XGBoost).

The 5 algorithms and their hyperparameters to be tuned are shown in Table 2.

Algorithm	hyperparameter	values
Lasso Regression	'alpha'	np.logspace(-7,0,10)
Ridge Regression	'alpha'	np.logspace(-7,0,10)
Random Forest	'max_depth' 'max_features'	[1, 3, 10, 30] [0.25, 0.5, 0.75, 1.0]
K-Nearest Neighbors	'n_neighbors'	[1, 3, 10, 30, 100]
XGBoost	'max_depth' 'learning_rate'	[0.03, 0.1, 0.3, 1] [3, 10, 30, 100]

For each algorithm, the function MLpipe_KFold_RMSE() repeats the whole process above 5 times for different random states. And for each random state, it returns a best model and the corresponding test score and test set, where the test score is RMSE, because it's used for regression problem and it has the same scale with the target variable to be thought as some kind of distance between the predicted values and the true values.

4 Results

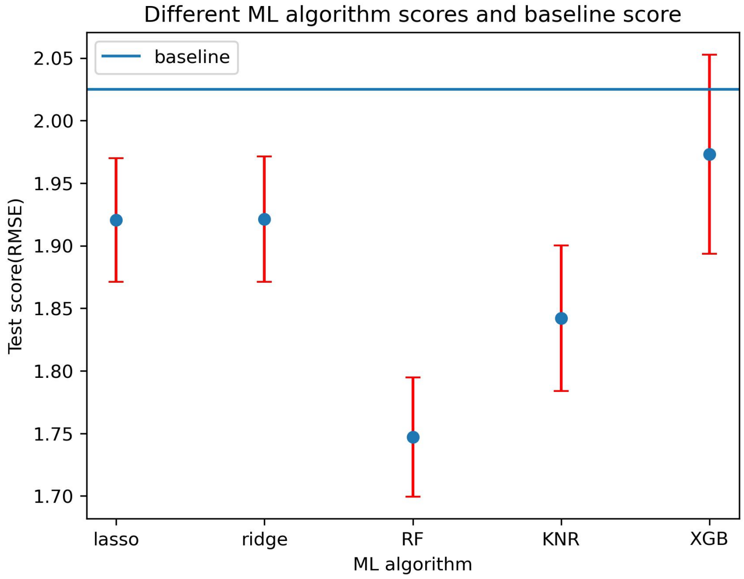

Figure 6: Test scores of 5 algorithms

Figure 6 shows the test scores of the 5 algorithms, where the baseline is the RMSE of target variable and its average. We can see that all the algorithms perform better than the baseline, where the Lasso and Ridge regression have 2 standard deviations above the baseline, and this value is 3 for K-Nearest Neighbors. Unexpectedly, the XGBoost is only less than one std above the baseline, which may be because the appropriate candidate hyperparameter values are not chosen. The Random Forest performs the best, which has nearly 6 stds above the baseline.

As a result, I show the result of the best Random Forest model in Figure 7. We can see that some of data points are correctly predicted, but there are still many predicted values that deviate from the true values.

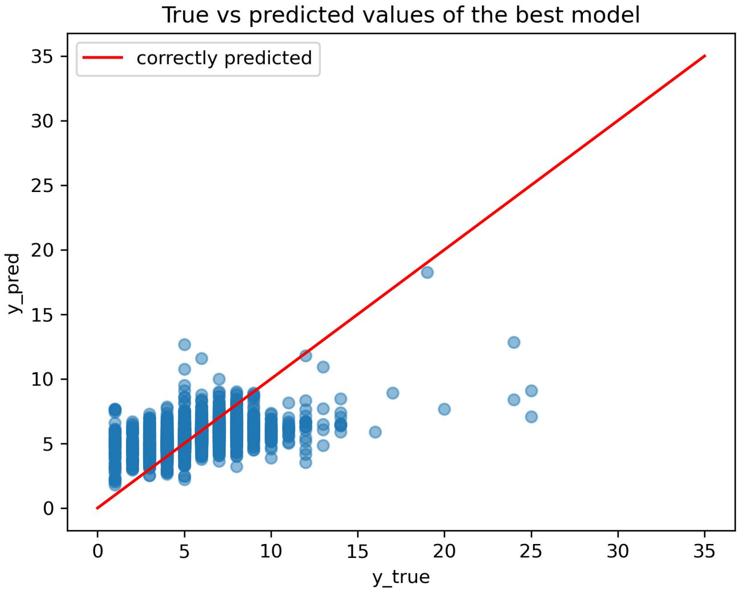

Figure 7: True values vs. Predicted values

I further explore the global feature importance of this model. Since it's a Random Forest model, I use permutation importance, mean decrease in impurity, and SHAP global importance respectively. Although they have some differences in their results, some features still appear in all of them. From Figure 8-10, we can see the views count, likes count, comments count and days from publish to trend are relatively more important.

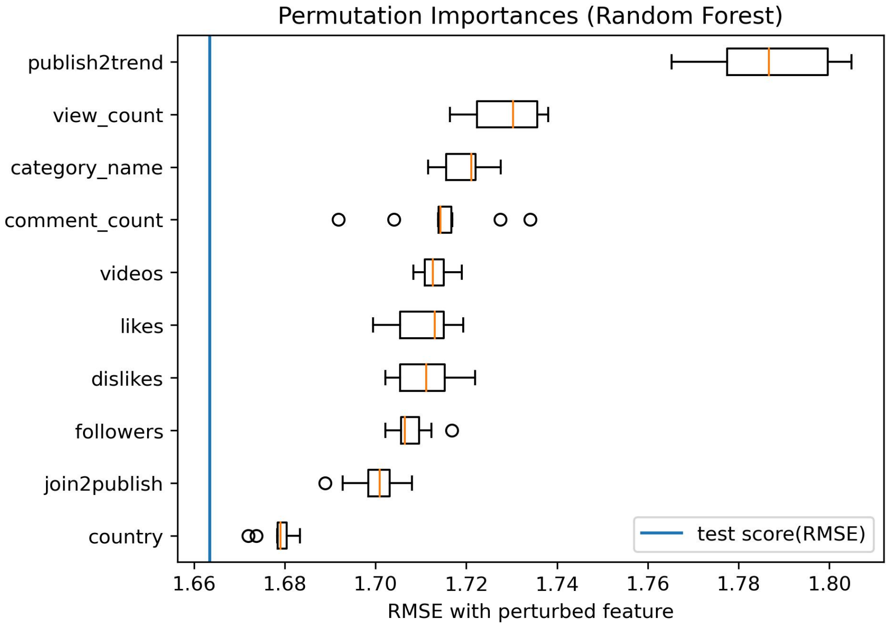

Figure 8: permutation importance

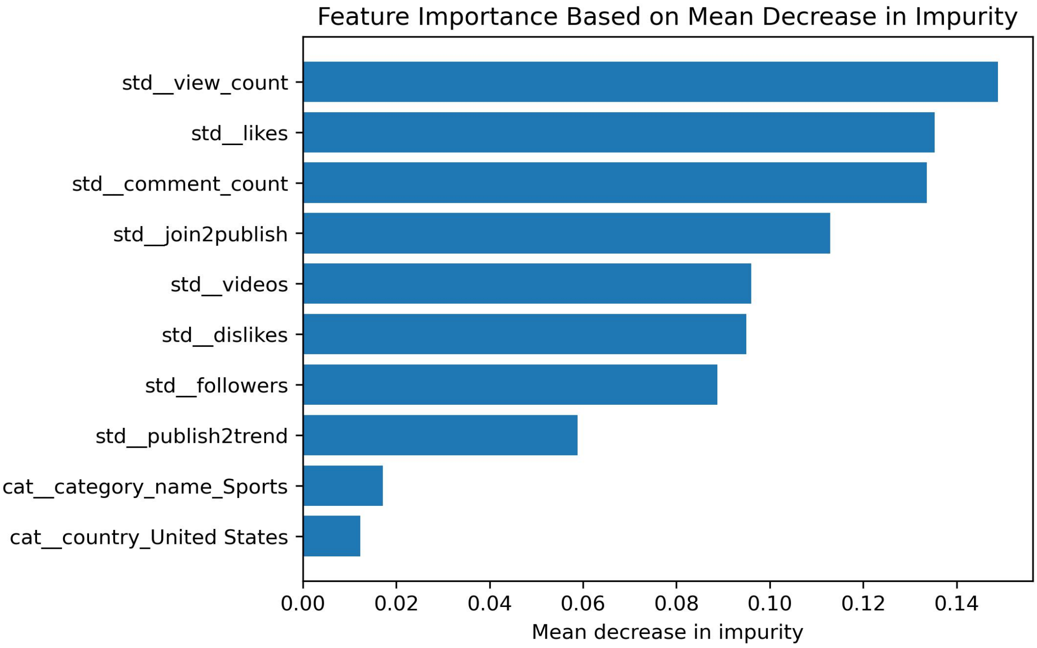

Figure 9: Mean decrease in impurity

Finally, I randomly choose several data points to do the local feature importance analysis. Figure 11 shows 3 of them with index=[1, 100, 3000]. Different features have different contributions for each data point. For the first

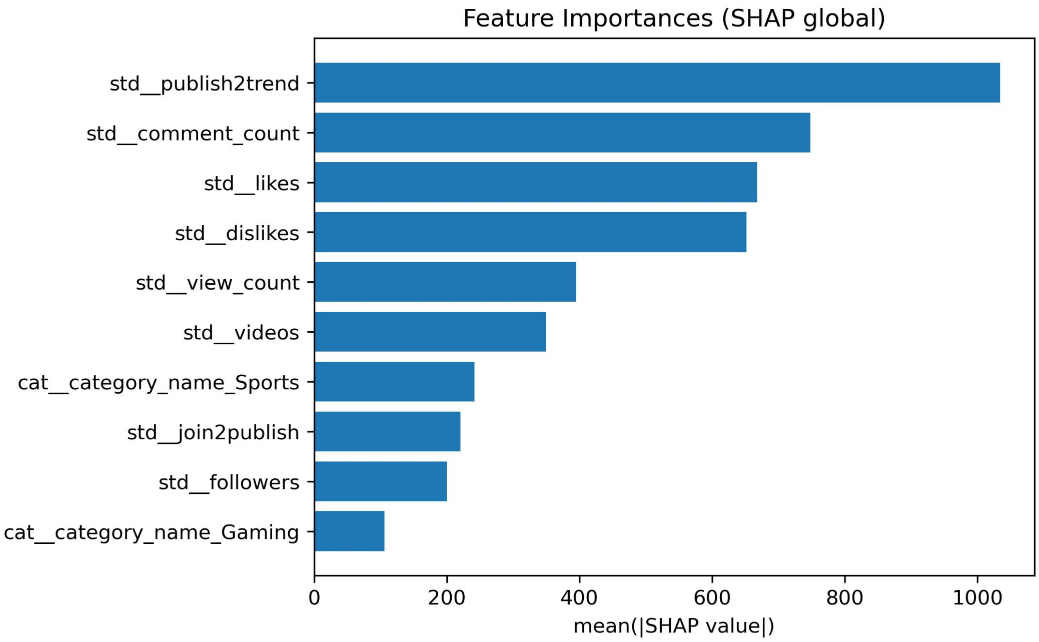

Figure 10: SHAP global feature importance

data point, the dislikes count has the largest positive contribution, while comments count has the largest negative contribution. But for the 100th and 3000th data point, this result is the opposite.

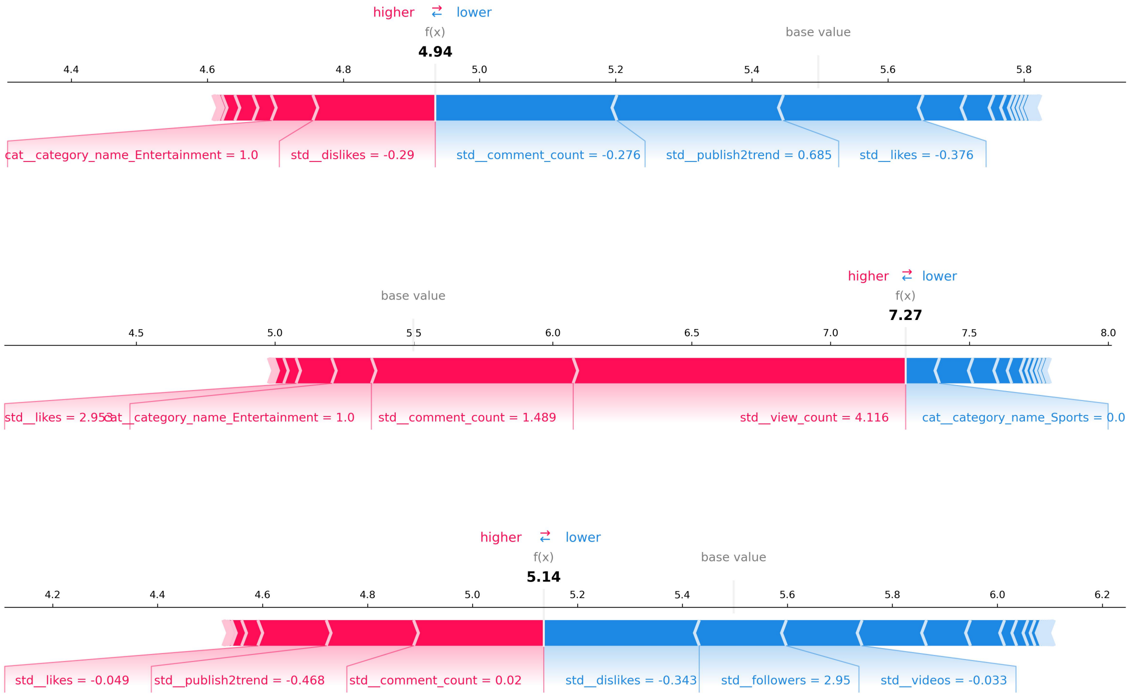

Figure 11: SHAP global feature importance

5 Outlook

This project is still kind of rough. From the above results, even the best model performs not very accurately and the predicted values are deviated. There are many ways to improve the model. I can try more candidate hyperparameter values and tune more hyperparameters to get a more precise model, especially for XGBoost. And I can try some other algorithms like SVM regression, which is given up because of the long running time. I can also use the date type variable or the text variable in some other ways. For example, the video's publishing time period can be taken into account. It is intuitive that if a video is published on the weekend, it will get higher attention. Even I can use some deep learning methods to build neural networks to directly analyse those text variables. The dataset can be extended to include videos who don't show on the trending list, and then we can use this dataset to better analyse which videos can be trending and which can not.

References

[1] M. Mohsin, 10 YouTube stats every marketer should know in 2022 [infographic], https://www.oberlo.com/blog/youtube-statistics.

[2] Trending on YouTube, https://support.google.com/youtube/answer.7239739

[3] YouTube Trending Video Dataset (updated daily), updated by R. Sharma, https://www.kaggle.com/datasets/rsrishav/youtube-trending-video-dataset.

[4] YouTube channels 100000, updated by I. Babikov, https://www.kaggle.com/datasets/babikov/youtube-channels-100000.

[5] R. Anand, Youtube-View,Like&Comment prediction, https://www.kaggle.com/code/rahulanand0070/youtube-view-like-comment-prediction.

[6] H. Mehta, youtube likes prediction, https://www.kaggle.com/code/hetulmehta/youtube-likes-prediction.

[7] W. Laknaoui, Sentiment Analysis USA(Step by step— Emojis—Tags), https://www.kaggle.com/code/wardalaknaoui1/sentiment-analysis-usa-step-by-step-emojis-tags.