YouTube Data Analysis

Howard Hu, Bowen Feng, Jing Lin, Yuchen Zhao

CSSE490/MA415 Machine Learning

22 May 2020

ABSTRACT

People watch popular videos on websites like Youtube every day, but do they ever wonder what kind of video will trend? We are using machine learning skills to deepen the understanding of trending videos by analysis and prediction.

This project aims for discovering the features and reasons behind the popular video based on the analysis of data from Youtube America. In the future people will want to know how far a video can go: how many views will it get? How many comments? Those are real-life concerns attached to economic benefits. We would like to propose a prediction about the features of trending video using the dataset we processed. This would help the producer of the video to analyze their videos and make wiser decisions in future production.

1.INTRODUCTION

As people view videos online every day on websites such as Youtube, it is certainly important for us to know what they are watching and what they will watch. Part of the video has significantly more audiences than the others, and we define those videos as trending videos. This analysis aims to produce a detailed analysis of the features the trending videos have and some predictions about the popularity of a video.

Because we are using a dataset with a large number of samples, processing it to be usable takes us some effort. We certainly spend some time cleaning the dataset and get rid of the useless data, like some video with comments closed, etc. We also need separate datasets to perform different analyses and perform detailed cleaning again on those. We spend some time comparing the fittest algorithms, looking through papers and test running them.

We also spend some time on finding the best algorithm for prediction. We are dealing with a huge dataset with limited features. Although our first plan is to attach popularity to each tag, we find using ridge regression to predict is a decent choice. It provides us the most accurate results made.

Many researchers have done their analysis with the same dataset as we have seen, but most of them put the focus on the relationship between different features - they are researching how the trending videos look like rather than what causes them to trend. We are trying to gain a more thorough understanding of the relationships between features.

The scope of the problem is that we download the Youtube dataset on the Canada region, and train our dataset of the United States in a model to predict the likes of the trending videos in the Canada region. For our continuous target, likes, we use the regression model to predict. We make some assumptions that the thumbnail link, description, and other features are not correlated to the likes. Therefore, we did not train those features in our regression model.

Also in order to ease our work, we considered the features like titles and tags to be unrelated, as we are doing the analysis on the trending video group not specific types of video currently. Some may argue that these do take a significant portion of the final results, but we modified our condition to clean the noises. As in the article "Classification of YouTube Data Based on Sentiment Analysis.", the authors mention that it is necessary to clean the dataset to fit the goal of our own project.

So, we are certain about one hypothesis: the ratio between different features of a video is predictable. Although we can use the data to predict something straight forward like views or likes which is not really convincing - it depends more on the quality and other issues, getting its ratio would be a better choice. In the future, we would want to further develop this project to make the project to be tag-related and title-related, which provides a more persuasive statement about the trending video.

2. LITERATURE REVIEW

Paper: Lim, Ji Young; Kim, Seulki; Kim, Juhang; Lee, Seunghwan. Identifying trends in nursing start-ups using text mining of YouTube content. 2/13/2020, PLoS ONE, Vol. 15 Issue 2, p1-14. 14p. (link)

Review: In this article, the authors illustrate that using YouTube content can analyze top trending videos in different areas. There are three main steps in this analysis. They are text mining, Delphi survey, and comparison. While working on our project, we use a word cloud to display the popular words either on tag or title of the trending video. We do not use the algorithm provided in this paper, however, we do find this paper to be helpful. The method uses to determine the importance of keywords gives us a clue about how we should process our text information.

Paper: Shaila S.G, Prasanna MSM, and Kishore Mohit. Classification of YouTube Data based on Sentiment Analysis. IJERCSE, Vol 5, Issue 6, June 2018. (link)

Review: In this article, the authors create a model that helps them to study the sentiment from the consumer's perspective based on classifying the youtube topics. The linear regression algorithm mentioned has been a very helpful reference. We end up finding many thoughts in the article to be valuable. The idea of cleaning data and removing redundancies is crucial in the data processing stage. After we finish our data processing, we go back and compare our methods with those mentioned in the article. The graphing method is mentioned in the article is probably the most helpful part, as it shows a better way to demonstrate the relationships between different features of the data. The multiple pair of plots is a clear visualization method it provides. Although we are doing deeper analysis, the article still provides a good example of preprocessing the dataset.

Paper: SZABO, GABOR, HUBERMAN, BERNARDO A, Predicting the Popularity of Online Content, Communications of the ACM, Aug2010, Vol. 53 Issue 8, p80-88 (link)

Review: This article is more about the general route we can take when collecting a dataset and predicting the popularity. As we go through the project, we find that the algorithm it contains does not fit our condition. In this case, this article is only useful while we need to consider the purpose and drawback of the project. In other words, it is a little vague about this project.

Paper: Predicting Success of Bollywood Movies Using Machine Learning Techniques(link)

Review: This article provides a completed analysis of the youtube data, but it is not the approach we are using. It is trying to predict the film's success, which is sort of similar to our popularity prediction. The methods it provided about non-numerical data are a good reference, but we did not use it in this project. The paper is certainly not complicated enough to predict something like the popularity of the movie.

3.PROCESS

3.1 Data Source

Our data source is the "Trending YouTube Video Statistics" by Mitchell J. As we all know, YouTube is a famous video website. There are billions of videos posted on Youtube. This dataset provides a daily record of the top trending Youtube videos. It contains five different regions such as the US, GB, and DE. For our project, we focus on the US regions.

https://www.kaggle.com/datasnaek/youtube-new

This provides us the dataset of the information about the trending video in the United States, we choose this as it is complex and is verified by many other engineers working in machine learning.

3.2 Preprocessing

Before processing the data for further analysis, we need to make sure there is no missing value in the dataset that would affect our analysis. So it would be easy to use the heatmap. If there is any missing data in the dataset, the heatmap will color them as light yellow since we use magma as our color scale. It turns out that the missing data only appear under the description column in Figure 1, which will not affect our analysis. Therefore, we keep the dataset the same as we downloaded from Kaggle.

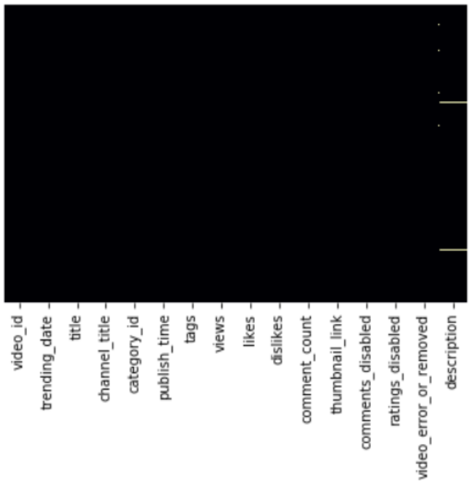
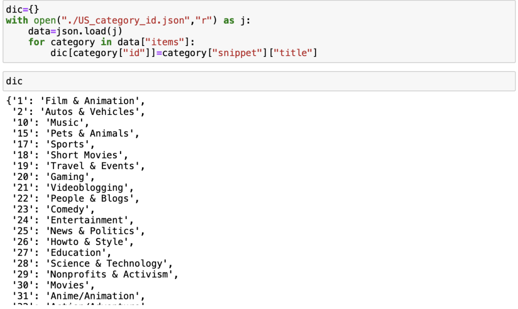


Figure 1. Heatmap plotted for the missing data

However, the dataset provides the category ID but not the description. Thus, to analyze how different categories affect the popularity of the trending video, we need to merge the category description into the dataset. So we merge data from the JSON file to create a new column in our dataset called category. Firstly, we create a dictionary that includes the category id and category name, as shown in Figure 2.



```
dic={}
with open("./US_category_id.json","r") as j:
    data=json.load(j)
    for category in data["items"]:
        dic[category["id"]]=category["snippet"]["title"]
```

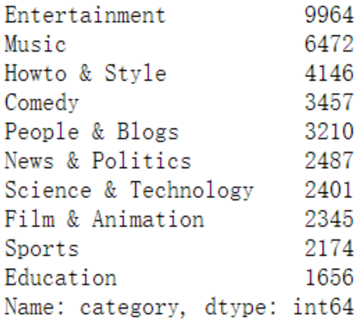
```
dic
```

```
{'1': 'Film & Animation',
 '2': 'Autos & Vehicles',
 '10': 'Music',
 '15': 'Pets & Animals',
 '17': 'Sports',
 '18': 'Short Movies',
 '19': 'Travel & Events',
 '20': 'Gaming',
 '21': 'Videoblogging',
 '22': 'People & Blogs',
 '23': 'Comedy',
 '24': 'Entertainment',
 '25': 'News & Politics',
 '26': 'Howto & Style',
 '27': 'Education',
 '28': 'Science & Technology',
 '29': 'Nonprofits & Activism',
 '30': 'Movies',
 '31': 'Anime/Animation',
 '32': 'Action/Adventure',
```

Figure 2. Mapping data into a dictionary

Secondly, we use the map function to create a new column in the data frame.

After we merged the data, as shown in Figure 3, now we can actually know the top 10 categories of the trending Youtube videos instead of category IDs.



```
Entertainment           9964
Music                   6472
Howto & Style           4146
Comedy                  3457
People & Blogs          3210
News & Politics         2487
Science & Technology    2401
Film & Animation        2345
Sports                  2174
Education               1656
Name: category, dtype: int64
```

Figure 3. The top 10 categories of trending videos

For analyzing the popular tags used in the trending videos, we decide to generate a word cloud, Figure 4, for better visualization:

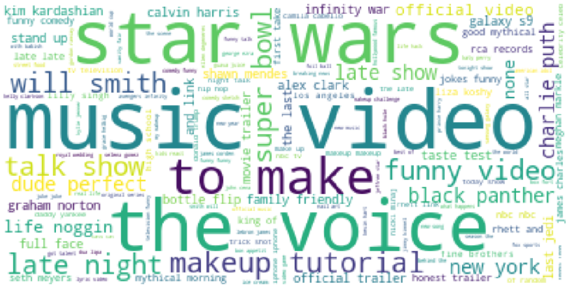


Figure 4. Tags Word Cloud

We can see from figure 4, the tags used most by the trending videos are “Music video”, “Star Wars”, ”The Voice”, “makeup tutorial”, etc. It makes sense because “Music video”, “Star Wars”, ”The Voice”, “makeup tutorial” has a significant influence on people in real life. Most entertainment videos attract people to click and view them. Therefore tags are an important factor of a trending video. As we can see from the word cloud, there are some popular tags selected by the number of views it has. It will certainly affect the popularity of the video.

We also generate a word cloud for the titles of the trending videos:

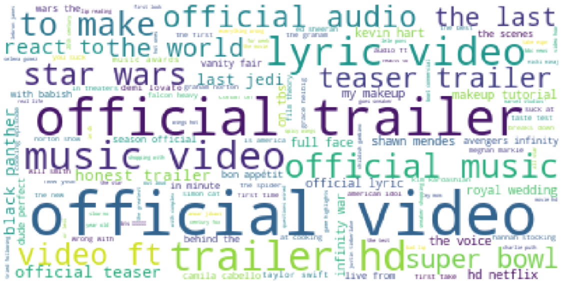


Figure 5. Titles Word Cloud

We can see from Figure 5, the keywords “official”, “trailer”, “music” appear in the word cloud several times. Adding those keywords to the video’s title might be a tip for the YouTubers who would like their videos to become popular. “Official video” makes videos have a strong title which gives people a sense of quality and attracts them to watch the video. A good title will certainly attract more audiences. We have found out the most popular title with a word cloud.

Before we apply a real algorithm into the dataset, we would like to analyze the relationship between features of the trending video. We use the pandas’ corr() function to find out the correlations among Views, Likes, Dislikes, and Comments:

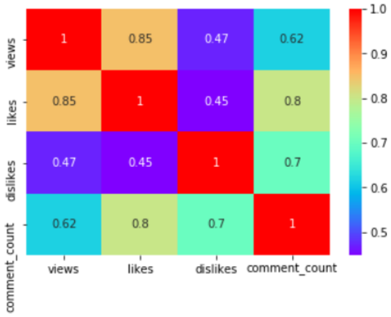


Figure 6. Heatmap for correlations of views, likes, dislikes, and comment_count

From the correlation heatmap, figure 6, we can see that views and likes are correlated with 85%. We can conjecture that more views of a video bring more likes. Likes and the number of comments are correlated with 80%. We would say the number of likes of a video has a positive relationship with its number of comments. Dislikes and the number of comments are correlated with 70%. Although the number of dislikes has a positive relationship with its number of comments, its relationship is weaker than the relationship between likes and comments. We can say that whether people like or dislike a video, it does not affect the decision of people who will or will not leave a comment under the video.

We also plot three bar graphs to study the ratio between the two features and which category has the highest such ratio. For the features of likes and views, in figure 7, the category with the highest likes-views ratio is Nonprofits and Activism, which makes sense since people would like to engage in nonprofits activities.

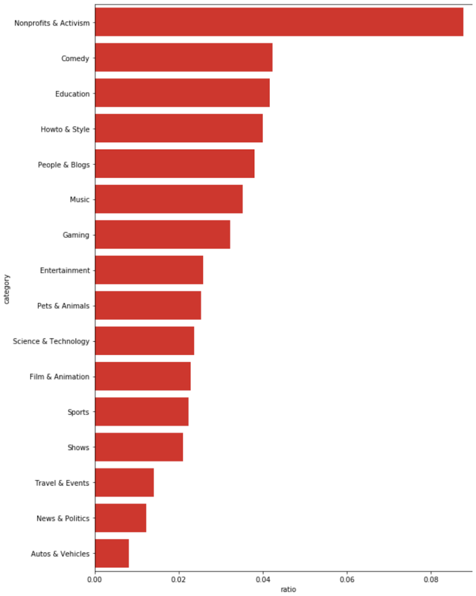


Figure 7. Barplot of the categories based on a likes-views ratio from highest to lowest.

For the features of likes and comments, in figure 8, the category has the highest likes-comments ratio is Shows, which is reasonable since people would like to leave comments for their liked shows,

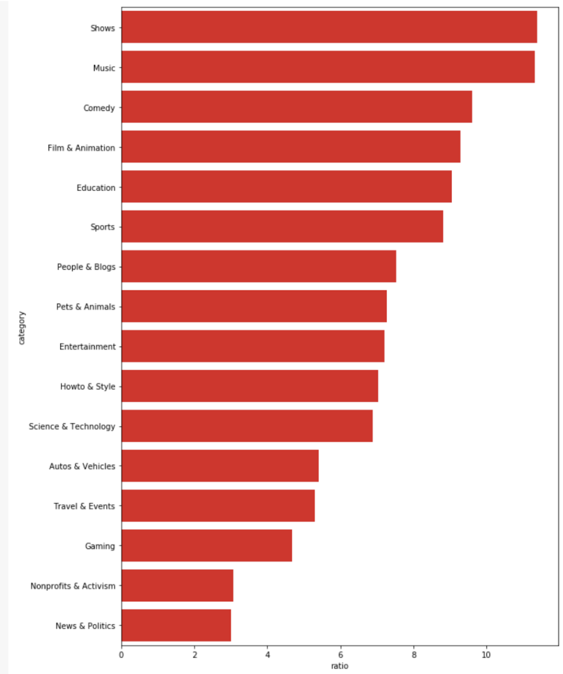


Figure 8. Barplot of Categories based on a likes-comments ratio from highest to lowest

For the features of dislikes and comments, in figure 9, we found that the category with the highest dislikes-comment ratio is News & Politics which is understandable since people usually want to express their thoughts when they see disliked news or politics, and leaving a comment is also a way to receive options from other people.

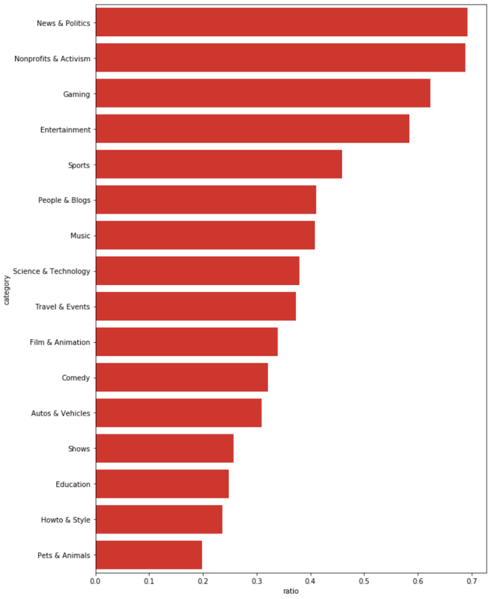


Figure 9. Barplot of categories based on a dislikes-comments ratio from highest to lowest

Then we want to analyze factors influencing the popularity of the video

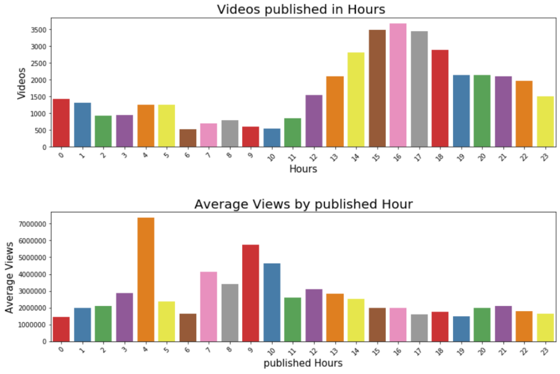


Figure 10: Barplots based on the Hour

We format the publish time into the date, hour, minute, and second. Then, we plot two bar graphs as shown in figure 10, the first graph shows how many videos published in the specific hour. As we can see from the graph, the videos published from 3 pm to 5 pm tended to become the Youtube trending video. The second graph shows the average views based on the published house. We see from the graph that the video published at 4 am has the most views on average.

3.2 Feature extraction or engineering

Before using different models to train the dataset, we would like to find out which features have a reasonable correlation with our target likes, which is one of the factors that can show the popularity of the YouTube videos. We use the function corr() from the panda's package and heatmap from seaborn open source, we obtain the graph as shown in figure 11.

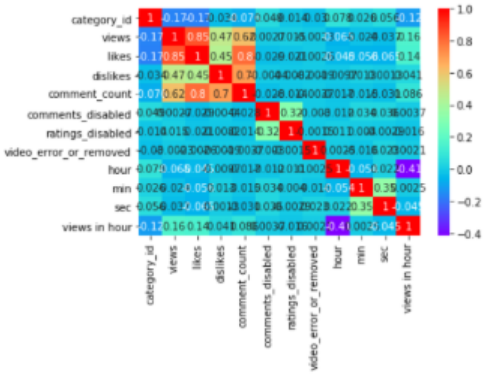
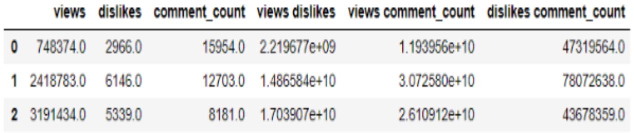


Figure 11. Heatmap for correlations of all features

By filtering out the correlation that is lower than 0.2, we only have three features left, views, dislikes, and comment_counts. Therefore, to improve the model accuracy or to obtain the most out of the three features, we decide to do feature engineering. The new dataset after feature engineering as shown in figure 12.

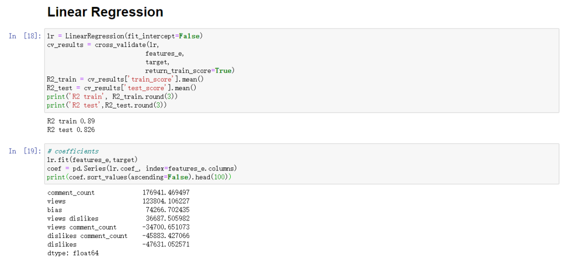


	views	dislikes	comment_count	views dislikes	views comment_count	dislikes comment_count
0	748374.0	2966.0	15954.0	2.219677e+09	1.193956e+10	47319564.0
1	2418783.0	6146.0	12703.0	1.486584e+10	3.072580e+10	78072638.0
2	3191434.0	5339.0	8181.0	1.703907e+10	2.610912e+10	43678359.0

Figure 12: Features after feature engineering

3.3 Classifier/regressor and tuning

Linear Regression and K-Nearest Neighbors(KNN) regressors are used in the analysis of predicating like rate and the features most likely affect the like or dislike rate.

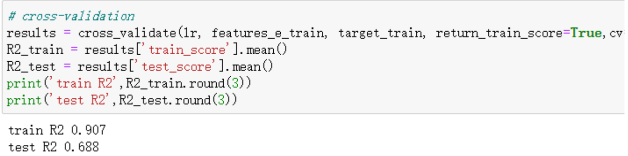


Linear Regression

```
In [18]: lr = LinearRegression(fit_intercept=False)
         cv_results = cross_validate(lr,
                                     features_e,
                                     target,
                                     return_train_score=True)
         R2_train = cv_results['train_score'].mean()
         R2_test = cv_results['test_score'].mean()
         print('R2 train', R2_train.round(3))
         print('R2 test',R2_test.round(3))

         R2 train 0.89
         R2 test 0.826

In [19]: # coefficients
         lr.fit(features_e,target)
         coef = pd.Series(lr.coef_, index=features_e.columns)
         print(coef.sort_values(ascending=False).head(100))

         comment_count             176941.469497
         views                     123804.106227
         bias                       74266.702435
         views dislikes             36687.505982
         views comment_count       -34700.651073
         dislikes comment_count    -45883.427066
         dislikes                  -47631.052571
         dtype: float64
```

Figure 13: Linear Regression

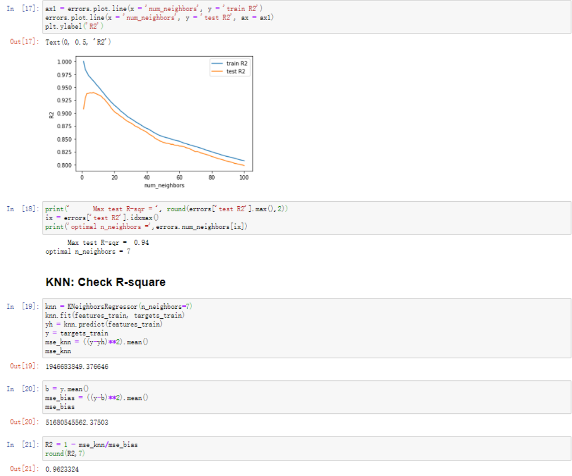
From the Linear Regression model, in figure 13, R-squared is 0.826 and the most important feature affecting people's viewing is comment_count.



```
# cross-validation
results = cross_validate(lr, features_e_train, target_train, return_train_score=True, cv
R2_train = results['train_score'].mean()
R2_test = results['test_score'].mean()
print('train R2',R2_train.round(3))
print('test R2',R2_test.round(3))

train R2 0.907
test R2 0.688
```

Figure 17. Cross-validation(lr)

As shown in figure 17, we obtain the train R-squared 0.907 and test R-squared 0.688, which shows an indication of overfitting data. Therefore, to avoid overfitting the data, we did the Ridge Regression and Lasso Regression. And the R-square from ridge regression is a little bit better than the Lasso regression (see table 1).



```
In [17]: ax1 = errors.plot.line(x = 'num_neighbors', y = 'train R2')
         errors.plot.line(x = 'num_neighbors', y = 'test R2', ax = ax1)
         plt.ylabel('R2')

Out[17]: Text(0, 0.5, 'R2')
```

```
In [18]: print('      Max test R-sqr = ', round(errors['test R2'].max(),2))
         ix = errors['test R2'].idxmax()
         print('optimal n_neighbors =',errors.num_neighbors[ix])

               Max test R-sqr =  0.94
         optimal n_neighbors = 7
```

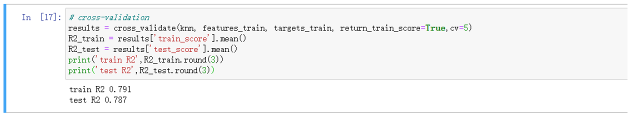
KNN: Check R-square

```
In [19]: knn = KNeighborsRegressor(n_neighbors=7)
         knn.fit(features_train, targets_train)
         yh = knn.predict(features_train)
         y = targets_train
         mse_knn = ((y-yh)**2).mean()
         mse_knn

Out[19]: 1946683849.376646

In [20]: b = y.mean()
         mse_bias = ((y-b)**2).mean()
         mse_bias

Out[20]: 51680545562.37503

In [21]: R2 = 1 - mse_knn/mse_bias
         round(R2, 7)

Out[21]: 0.9623324
```

Figure 14: K-Nearest Neighbors Classification

Obviously, by comparing the result from figure 13 and figure 14, the result of R-squared predicted by KNN is much better than that of Linear Regression. The accuracy rate with the KNN method is 0.962, by using seven neighbors.

```
9          10   0.951330   0.935472
```

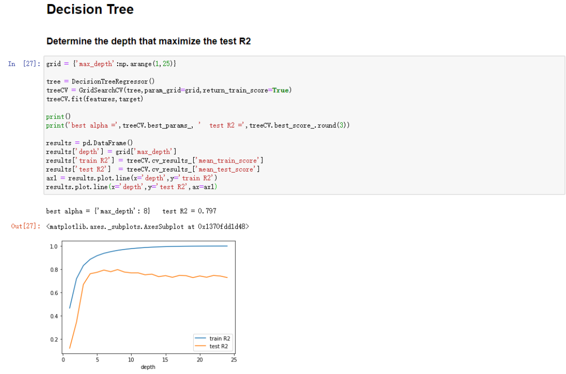


```
In [17]: # cross-validation
         results = cross_validate(knn, features_train, targets_train, return_train_score=True,cv=5)
         R2_train = results['train_score'].mean()
         R2_test = results['test_score'].mean()
         print('train R2',R2_train.round(3))
         print('test R2',R2_test.round(3))

         train R2 0.791
         test R2 0.787
```

Figure 18: Cross-validation(KNN)

As shown in Figure 18, we obtain the train R-squared 0.791 and test R-squared 0.787. Those two R-squared numbers are close enough that we can neglect the difference between them. So, there is not an overfitting or underfitting problem for KNN Regression.

Other than KNN and Linear Regression, decision trees are also used for predicting the liked rate.



Decision Tree

Determine the depth that maximize the test R2

```
In [27]: grid = {'max_depth':np.arange(1,25)}

         tree = DecisionTreeRegressor()
         treeCV = GridSearchCV(tree,param_grid=grid,return_train_score=True)
         treeCV.fit(features,target)

         print()
         print('best alpha =',treeCV.best_params_, '  test R2 =',treeCV.best_score_.round(3))

         results = pd.DataFrame()
         results['depth'] = grid['max_depth']
         results['train R2'] = treeCV.cv_results_['mean_train_score']
         results['test R2']  = treeCV.cv_results_['mean_test_score']
         ax1 = results.plot.line(x='depth',y='train R2')
         results.plot.line(x='depth',y='test R2',ax=ax1)

         best alpha = {'max_depth': 8}   test R2 = 0.797

Out[27]: <matplotlib.axes._subplots.AxesSubplot at 0x1370fdd1d48>
```

Figure 15: Decision Trees

The accuracy from Decision Tree is the lowest among the three predicting methods. The accuracy of predicting like the rate is 0.797 with depth eight, as shown in figure 15.

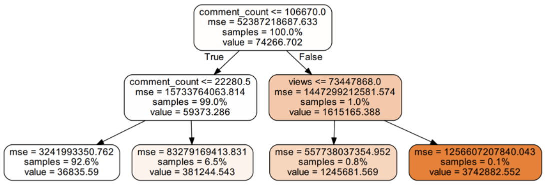


Figure 16: Decision Tree with Depth 2

From figure 16, comment_count is the root and is the most important element affecting people's favor of videos. This meets the prediction of Linear Regression that comment_count affects the like rate more than other features.

3.4 Post-processing

Since our results from Linear Regression, KNN, and decision trees are straightforward to interpret, we don't need an additional section to describe.

4. EXPERIMENTAL SETUP AND RESULTS

We use several regression algorithms in our data in order to predict like numbers such as linear regression, KNN regression, and decision trees (see table 1).

Table 1. The R-squared result from three models.

Algorithms	**Accuracy (Test error)**
Linear Regression	0.826
Ridge Regression	0.836
Lasso Regression	0.83
KNN regression(n_neighbors = 7)	0.94
Decision tree (max_depth = 8)	0.797

We conjecture that KNN regression with 7 neighbors has the best accuracy for this dataset. The most surprising is the accuracy of the decision tree is even worse than linear regression. We would say the decision tree model is not a good choice for this situation.

Based on the three models we train, we found that Linear Regression has the R-squared of prediction is 0.826, KNN regression has the test R-squared equal to 0.962, Decision Tree has R-squared equal to 0.797 with depth 8. According to the R-squared, we believe that KNN regression fits our data the best to predict the likes based on the features, views, comment_count, and dislikes. The most significant feature among them based on three models is comment_count. It seems that people who leave comments for the trending video will also give a thumbs up for that video, and we can say that the number of comments is crucial when analyzing the popularity of the video.

5. DISCUSSION

We found out the KNN regression algorithm has the best accuracy for this dataset. However, we have limited train features which only include views, dislikes, comment_count, and so on. Our system may underfit the data because we may miss some important features such as videos post date, but KNN regression has a good performance based on current given features.

6. CONCLUSIONS AND FUTURE WORK

The key findings of our project are that we find the most important element affecting the audience's favor. Based on the United States Youtube trending videos dataset, people's favorite category is entertainment. Another interesting finding is whether people like or dislike a video, they are willing to give a comment to express their idea. For the tags, We found that most trending videos have the tags of "Music video", "Star Wars", "The Voice", "makeup tutorial", etc. For the title, the most trending videos have the keywords, "official", "trailer", and "music", in the title of the video. From the result of KNN and Decision Tree, we find out that comment_count can affect people's favor most. According to this finding, video producers can do some improvements based on comment_count to raise their products' popularity. The weakness of our research is we only did research on Youtube trending videos in the United States. If we want to know more about the popularity of the videos of entire human beings, research on the U.S. trending videos is obviously not enough. If we have another week or two, we will work on the videos in a different area, and do the comparison and try to find the reason behind the difference or similarity. And we only create word clouds for the tags and title. There is definitely more we can discover from those two features. We might analyze the tags and titles further to investigate the relationship between different tags and titles with popularity.

If we have a whole year or more to work on it, we might merge the dataset from all over the world, and do the comparison to find out the similarity and differences of all human beings and the popular topics that people are interested in. The major change would be the model we use to predict the target since we have not trained all the models we learned from the class, such as random forest, bootstrap, and gradient boosted trees, and etc. And based on the different targets we might research on, we will change our model.

7. KEY CHALLENGES AND LESSONS LEARNED

Data preparation is critical for our prediction analysis. Before we apply algorithms in the dataset, we need to merge some essential data from the JSON file to CSV file. Some group members are unfamiliar with JSON files, so we spend more time than we expected.

In the future, we will implement the lessons we learned from this project. When we will do prediction analysis, we will always try different methods, compare them, and interpret them. Some methods are suitable for a typical data set. We need to address the reason behind their fitness, with respect to the real-world interpretation of the results.

8. REFERENCES

[1] Lim, Ji Young et al. "Identifying Trends In Nursing Start-Ups Using Text Mining Of Youtube Content". *PLOS ONE*, vol 15, no. 2, 2020, p. e0226329. *Public Library Of Science (Plos)*, doi:10.1371/journal.pone.0226329.

[2] S.G, Shaila, et al. “Classification of YouTube Data Based on Sentiment Analysis.” *IJERCSE*, vol. 5, no. 6, June 2018.