Youtube Video Title Performance Analytics

Tien Dat Johny Do - 30087967 , Isaiah Lemieux - 30127869,
Gian Adug - 30085990, Oscar Campos - 30057153
University of Calgary - SENG550 Scalable Data Analytics
Date : December 20, 2023

I. PREAMBLE

Contribution Statement : The team has equally distributed the work evenly, where each team member contributed 25% to the overall project. We started with peer programming taking on different approaches such as looking at features. Here Oscar and Gian took on an approach with looking at filtering tags, and Isaiah and Johny took a look at filtering categories within the Youtube dataset. After team meetings, we revised our strategy that was guided by the TA to clean our data, make a bag of words with the tags, and train the model and observe the accuracy, f1-score, and precision scores using a two chosen classifying model and one predictive model.
Declaration : Our group has a mutual agreement of all members working equally and each individual has put an equal amount of work into this project.

Signatures :
Isaiah Lemieux
Gian Adug
Oscar Campos
Tien Dat Johny Do

Github Repository :
https://github.com/OscarCampos98/SENG550ProjectFinal

II. ABSTRACT

The project objective is to investigate how the title of Youtube videos impacts its performance in terms of views, likes, and overall channel activity. The project is designed to conduct analysis on trending video titles by examining key words, title formats, word pairings, and other title metrics in order to gain valuable insight to possible title modifications that can enhance user activity on the video to elevate the channel's performance.

The main outcomes would be conducting title analysis, and finding an optimization strategy that will overall enhance channel performance.

III. INTRODUCTION

The main problem that our group chose to tackle is the one of writing compelling titles for Youtube videos that will ultimately garner the most user activity. Youtube videos are posted on the main page with just a thumbnail and a title, and depending on how captivating the title and thumbnails are, the user will choose to watch to access the video. Our project plans to investigate the trending videos on Youtube and perform analysis on their titles

to understand user activity and optimize video performance.

The importance of this problem would be learning how to capture Youtube user's attention through the title. A trending video will include a captivating title that will contribute to enhancing user engagement, increasing click-through rates, and attract user's attention when compared to the numerous videos a user may see on their homepage. To find the most optimal title would significantly impact the overall video performance in terms of video metrics.

Some notable prior work within the Youtube title analysis space would be looking at keyword analysis tools, search engine optimization(SEO) tools, sentiment analysis on user's engagement (comments/reviews).

Despite some of the existing projects and tools made, our solution aims to understand how titles will perform and create an image of what Youtube titles should encompass by viewing the pattern in titles that lead to more views, comments, and in general user engagement. Some new features our project aims to use is the use of analytics on the tags that Youtube videos use in order to target a certain audience. Our project aims to dive deeper within the categories and tags that videos use in order to garner user engagement. With this approach we will take a look at using the Naive Bayes classifiers to train and test our data. We wanted to explore Binary, Multinomial Naive Bayes, and Linear regression models within this project.

Some of our data analysis questions that were posed during this project would be:

- How does this certain category affect video engagement?
- Do certain titles lead to higher initial engagement?
- Are there patterns in titles in terms of garnering views?

Our proposed implementation would be to create a predictive model using keyword analysis on the Youtube tags category in order to optimize and predict the performance of Youtube video titles. The main findings for this project include revealing specific keywords that significantly impact the performance of a video.

IV. BACKGROUND & RELATED WORK

Some technical background that would be useful to understand for our report would be the libraries that we are using. Our group is using spaCy, an open source library that is used for Natural Language Processing (NLP) within Python. This tool helps us extract words in a language comprehension system. This tool is very fast and has accurate analysis compared to other NLP tools. We will also be using spaCy to remove emojis from the data and keep all alphanumeric characters. WIth spacy due to it being heavy modules, our group removed some of its functionality such as parser in order to get more efficient computation times. This did not change any of the metric results, but changed in computation time.

In addition, Naive Bayes, a supervised machine learning algorithm, will be our classifier that will classify the text.

We choose Naive Bayes due to its very scalable nature and its general use for classifying text.

Next, Multinomial Naive Bayes algorithm is a machine learning method that is mostly used in NLP. It is widely used when features are discrete and represents a count of their respective category.

Linear Regression Model is a machine learning model that makes a relationship between dependent variables and a singular or more independent variable. This model is widely used as a predictive model.

Some strategies used for natural language processing (NLP) within this project would be the use of bags of words and Word2Vec. Bag of words would be used within the classifying models looking at frequencies of words and Word2Vec would be used within the regression model for more complex and word iterations/ frequencies. Both models are techniques to handle NLP.

The use of One-hot encoder was used to encode the categories within the dataset into a numerical format for our regression model.

There are three main categories of notable softwares that have been mentioned that pertain to our project: keyword analysis, SEO tools, and sentiment analysis.

Firstly, with the technique of keyword analysis, user's have used tools in order to generate keywords for titles. This technique is a popular strategy within the marketing industry to understand what viewers are actually searching. Some software projects with this technique are Ahrefs, VidIQ, and Semrush that work specifically with keyword analysis.

Secondly, work within this space using a SEO tool is to find a strategy to get videos to the intended target audience. This method involves investigating captions, transcripts, titles, and tags of a video to understand how to optimize these elements to increase video popularity. There are many tools that help with SEO such as TubeBuddy, Youtube autocomplete, and Sprout Social.

Lastly, sentimental analysis on the video's comment section is another project that is closely related to our project. This tool aims to address if the title pertains to the content of the video and rates videos depending on the polarity of comments.

All of these previous works encompass keyword suggestions in order to optimize the user engagement on Youtube videos.

These projects all review the importance of how the titles are regarded within the Youtube space. These projects gave us our initial project idea, but we wanted to explore different areas of this methodology. Although these projects are very similar to our project, our project aims to look deeper on the tags and categories that user's may use in order to make our bag of words. With this feature we want to explore how well our model can perform in predicting video performances.

V. METHODOLOGY

To begin, the group used Google Colab leveraging pySpark in order to do this project due to its capability to utilize CPU

and GPU resources. This allows our group to reduce runtime and make our systems most efficient with our code running it within batches to increase efficiency. First we installed libraries such as “spaCy”, then we imported some useful libraries such as “pandas” and “re”. Next, we loaded the dataset, which was a youtube CSV file where we started our data preprocessing. The 2022/2023 dataset was exclusively utilized due to the file's size being too large for processing otherwise.

For our data preprocessing step we removed duplicate rows based on titles, and channel titlesand dropped the channelId in order to simplify our table. For emojis, we decided to remove them from the titles since encoding them didn’t give significant improvements. This is done by a function that only looks for alphanumeric characters. After removing the emojis, we filtered the data to only take a look at videos and not other forms of content such as “Youtube Shorts”. Here we took our keywords of “Shorts” in the title, description and channel titles.

In addition, our group added another column of “title language” and a function to detect the language that the video title is in. We only wanted to filter the rows with languages that are not in English just to get the best model we can using the english spaCy text processing.

Next, we tokenized the text using spaCy by removing the stopping words, any punctuation, and lemmatizing the words together. After this, all titles and tags are cleaned and processed in order to start analysis.

Next we converted all words to lowercase and imported “nltk” for more text processing. We print some example tags and titles to get a sense of what data we have now that the data has been transformed and cleaned.

Now we will start with converting the text data into a numerical format for our machine learning model. Firstly, we will use CountVectorizer in order to extract text and vectorize the text into a sparse matrix using term frequency as a feature. We then assigned the target variable being the categoryID column and converted it to an array for reshaping and comparing purposes.

Lastly, to test our classifying model, we imported “Train_test_split” from the Scikit-learn library in order to use our machine learning model for classification. Our training and testing split encompassed 80% of the data for training and 20% for testing. With this, we ran the Naive Bayes classifier from the GaussianNB model, and Multinomial Naive Bayes

For our linear regression model, we would combine our features within a single vector column and use pySpark’s linear regression as our predictive model. This dataset we split 70% to train, and 30% to test.

We then classified and predicted how well the model will do by calculating the accuracy, f1-score, and precision metrics for the Naive Bayes. The metrics for the linear regression model is calculated through root mean squared error (RMSE), Mean Absolute Error (MAE), and R squared (R^2).

VI. RESULTS

Here are the results from gathering the calculations from using the Naive Bayes classifier.

Metrics for Naive Bayes	**Scores**
Accuracy	*0.55945*
F1-score	*0.39867*
Precision	*0.38497*

Figure 1 : Metric Table from Naive Bayes Classifier

With an accuracy of 0.55945, F1-score of 0.39867, and precision of 0.38497 there is room for improvement and that our predictive model has moderate capabilities. The accuracy is still below a threshold of what we would like to predict. Although the accuracy is below our standard, it still has the capability to predict moderate results. The F1 score indicates an imbalance of performance in precision and recall. Our group needs to find a better balance between the trade-off between precision and recall. As for precision, it is positive, but still contains limitations. Some ideas to enhance our model's performance would be to explore other algorithms, change features, or fine tuning our parameters to enhance our metrics.

With the other model of Multinomial Naive Bayes, here are the results from gathering the calculations :

Metrics for Multinomial Naive Bayes	**Scores**
Accuracy	*0.80057*
F1-score	*0.70230*
Precision	*0.71031*

Figure 2 : Metric Table from Multinomial Naive Bayes

The multinomial naive bayes model yields results for accuracy at 0.80, F1-score at 0.70, and Precision at 0.71 which are significantly higher than the binary Naive Bayes model we used previously. With accuracy being 0.80 it indicates that this model has great capacity to predict correctly. With an F1-score of 0.70 it gives a better balance between precision and recall. We found a great threshold to balance both these two classes. Lastly, with precision being 0.71, it gives our model a very positive prediction capacity. In general, this model yields high metric results for each category and makes it more of a reliable model.

Furthermore, here are the results from doing Linear Regression Model :

Metrics for Linear Regression Model	**Scores**
Root Mean Squared Error (RMSE)	0.01711
Mean Absolute Error (MAE)	0.00592
R-Square (R^2)	0.62026

Figure 3 : Metric Table from Linear Regression Model

With a root mean squared error of 0.00171 it indicates better accuracy and in this case it has very low error when predicting on the model in terms of view

count. With the mean absolute error of 0.00592 being low it indicates very good performance and indicates that the model can predict decently close to predicting actual values on average. Finally the r-squared value was 0.6202 which indicates that around 62% of the view counts could be deemed through the features used within the model. Overall, this linear regression model performed well with high quality metrics. Below we will take a look at some of the diagrams that we have made in regards to this model. Here we will have Actual vs Predicted Views dot plot, 3 cumulative explained variance diagrams, and a correlation matrix.

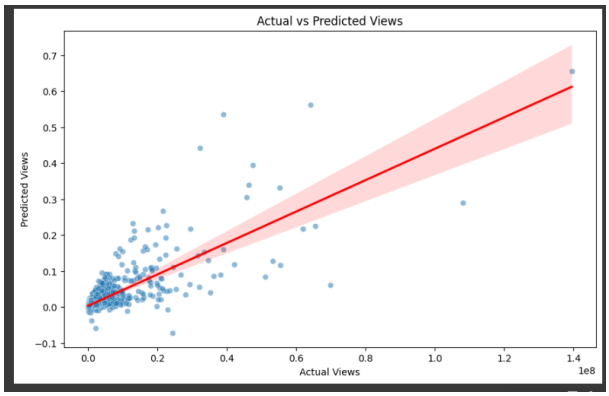

Figure 4 : Actual views plots against predicted views

The graph above shows a diagram plot of Actual views vs Predicted Views. The closer the points are to this fitted line shows how well it will perform and the further it is would predict poorly on how the model did. Most data points strayed moderately close with a couple variable points. Here it gives us a good idea of how well our model performed with the predicted and cross validating actual views.

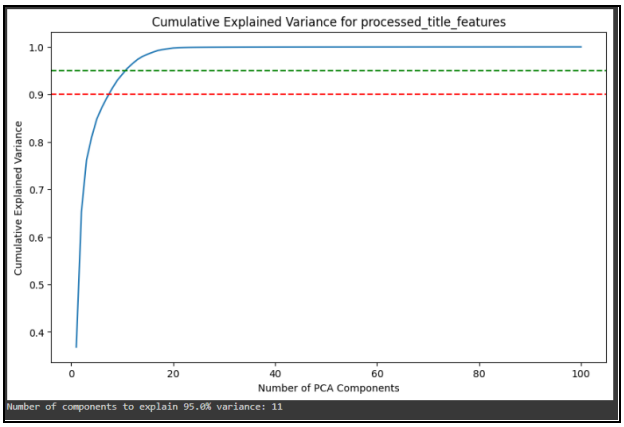

Figure 5 : Cumulative Explained Variance for Processed Title Features

For the cumulative explained variance for processed titles features it performed well due in indicating that the initial components captures a significant feature within the data. The plot then gradually increases which is a good sign to a good diagram. Within this processed title feature, the balance between trade-offs between its features.

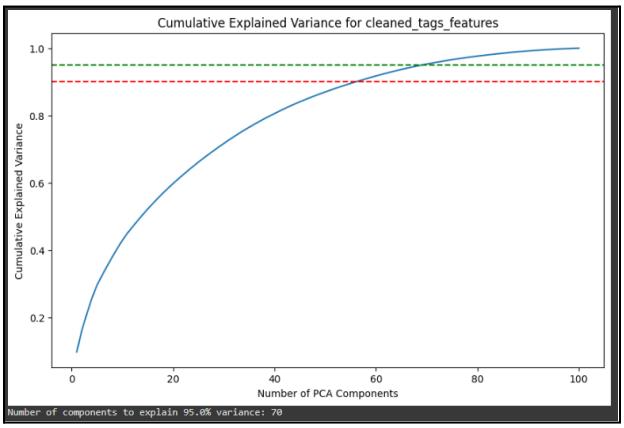

Figure 6 : Cumulative Explained Variance for Cleaned Tag Features

For the cumulative explained variance for processed cleaned tags features, it didn't perform as well due to the not rapidly rise within the plot curve. Based on

this diagram, the cleaned tag features did not have a good trade-offs between its features

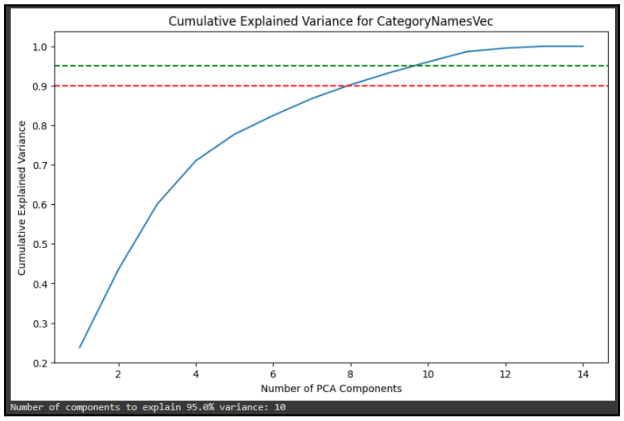

Figure 7 : Cumulative Explained Variance for Category Names Vector

For the cumulative explained variance for category names vector, it didn't perform well due to similar reasons in the diagram of cleaned tags. It did not possess a rapid rise within the plot curve at the start meaning that it does not capture the variance in the data from this feature curve.

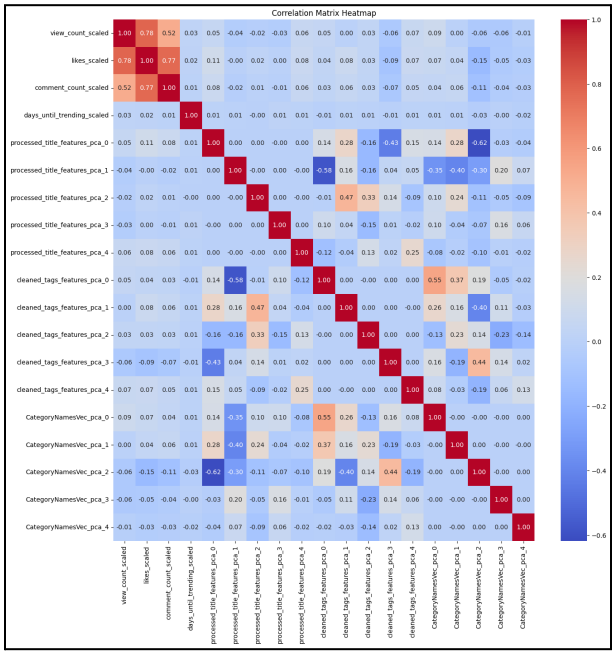

Figure 8 : Correlation Matrix between features/columns

With our correlation matrix it shows the correlation coefficient between the features within our project. Here with the heatmap, it shows a strong relationship with the darker/denser colors and a weaker relationship with lighter colors highlighted on the scale on the right. This matrix shows different patterns within the matrix and shows the relationship of features within the dataset.

Comparing first the Naive Bayes models, the multinomial yields better results in each aspect of accuracy, f-1 score, and precision. Our group found that using Multinomial Naive Bayes helps us gain better results due to it classifying using multiple categories instead of just the binary classification. Multinomial Naive Bayes is able to handle more classes and does better considering the weights of occurrences compared to a binary model. We find that multinomial naive bayes perform better on datasets that include text-based classification and its ability to retain information regarding other classes of features.

Now comparing the naive bayes models and the linear regression model. Both models have different objectives where naive bayes want to categorize videos and linear regression estimates the view count. Due to this fact, depending on the use case, we would use Multinomial Naive Bayes to categorize classes, and we would use Linear Regression Model in order to be our predictive model. Both these models performed well within their respective areas where Naive Bayes could accurately classify

videos with very high metric scores that consider trade-offs, and Linear regression model had an accurate ability to estimate view counts of Youtube videos.

As for future work on this model, some ways ours group ideas that could improve this model would be including different strategies such as exploring word embeddings to capture the relationship between words, more fine tuning of the hyperparameters in order to maximize performance, filter and include more data of the dataset, incorporate the behavior of sentence structure, and lastly if we could experiment with different cross-validation techniques to ensure no imbalance within the dataset.

VII. CONCLUSION

Our model implements keyword analysis using the tag categories to predict the video's engagement level. The process of data preprocessing, data cleaning, title analysis, and predictive analysis were all used in order to complete this project and garner results. Our model uses many tools such as Natural Language Processing in order to extract words that our model will vectorize using a vectorizer. The dataset will then get trained and tested to predict the accuracy, f1-score, and precision of the model. With an F1-score of 0.39 it is an imbalance between precision and recall that our group wants to grind to a threshold of at least 0.7 in order to get a better balance between the parameters of precision and recall.

Moreover, after testing a new model of Multinomial Naive Bayes, the metric yielded accuracy of 0.80, f1-score of 0.70, and precision of 0.71 which performed better than the binary naive bayes model. Our group found that we needed a model that had the capacity of classifying multiple categories instead of just a binary classification. The more categories our model was trained on, our results increased.

Lastly, with the linear regression model it performed well in predicting the views of Youtube videos with a root mean square error of 0.0171, a mean absolute error of 0.00592, and a r-square score of 0.6202 which reflects good results to estimate views counts compared to actual views. This tackles the use case of finding and predicting the amount of views of a youtube video.

In conclusion, we were able to make two classifying models for Youtube titles to understand their user engagement in respect to categorizing the tags of a Youtube video and make one predictive model for Youtube video views in order to understand the metrics that goes into making a Youtube title.