TubeTrends
Final Report

Devansh Grover 2022151 | Manan Aggarwal 2022273 | Shobhit Raj 2022482 | Vashu Chauhan 2022606

Abstract

The success of a YouTube video depends on factors like subscriber count, video tags, category, and length. Using machine learning, we predicted video views, with the Random Forest Regressor outperforming other models. EDA on the dataset revealed interesting patterns and correlations affecting performance. We identified prevalent tags in popular videos and developed a tag suggestion mechanism to increase views. We compared the efficiency of a traditional brute-force algorithm for tag suggestion against our machine learning-based approach, finding the latter to be significantly more time-efficient. An intuitive GUI lets users predict views and receive optimized tag recommendations. This project highlights the potential of machine learning to optimize digital content performance, setting the stage for further innovation in this evolving domain. The code for this project can be found in this GitHub Repository.

1. Introduction

The rapid expansion of digital platforms has established video content, particularly on YouTube, as a dominant medium for driving online interactions and shaping audience engagement. With millions of videos uploaded daily, creators face intense competition, where success is influenced by various factors such as subscriber count, video tags, category, duration, and audience feedback. Among these, view count stands out as the most significant metric, directly impacting both engagement and monetary outcomes. This project, **TubeTrends: Predicting Video Engagement and Performance**, leverages machine learning to predict video success, with a focus on view counts as the primary performance indicator.

2. Literature Survey

1. Machine Learning Models for YouTube Ranking and Views Prediction [1]: This study uses ML and NLP to predict YouTube video view counts and rank content based on trending topics. It employs regression models like XGBoost, Random Forest, and Decision Tree Regressors, with features including title, tags, likes, dislikes, description, and time differences. Preprocessing techniques like correlation analysis and feature engineering enhance model performance. Word2Vec embeddings are used to analyze tags and descriptions for better content alignment with trending topics. Deployed via IBM Watson AutoML, the tool helps creators optimize videos for greater reach. Future work includes thumbnail prediction using CNNs and trend-aware ranking models.

2. YouTube Video Classification based on Title and Description Text [2]: This study focuses on classifying YouTube videos by utilizing NLP techniques applied to video titles and descriptions. Instead of analyzing video content directly, this study uses a text-based approach to categorize videos into six predefined categories. This method emphasizes the efficiency and scalability of text-based classification over complex video content analysis. To convert the textual data into a usable format for machine learning, the authors employ the Bag of Words (BoW) technique using CountVectorizer from scikit-learn. This method represents the corpus as a matrix of token counts, making it suitable for input into machine learning models.

3. Dataset

The dataset was initially downloaded from `Kaggle`.[1] In order to gather more features, this dataset was then updated using API calls to the `YouTube API V3` [3] using a manually written python script. The new features included `thumbnail_link`, `comments_disabled`, `region`, `duration`, `definition`, `ratings_disabled`, `description`, `caption`, `comment_sentiment` and `subscriber_count`. At the time of utilizing and updating this dataset we faced an issue of the channels or

[1] https://www.kaggle.com/datasets/rsrishav/youtube-trending-video-dataset

videos no longer being available in which case we decided to drop these entries. This allowed us to obtain a dataset of about 450K rows. We also encountered problem of some videos not having comments enabled, therefore we were unable to determine the `comment_sentiment`, in which case we used a default value of 0.

`comment_sentiment` was extracted by querying 10 comments randomly from the channel using `vaderSentiment`[2], which is tuned for sentiment analysis on social media. This sentiment analysis was applied on all the comments and then the mean of these was stored as the feature value.

In order to systematically extract and potentially update the tags, we first compiled all the tags used in the videos we had collected, referring to this collection as the **taxonomy of tags**. We then processed the titles and descriptions by **tokenizing the text** using the tokenize function from the `NLTK` module[3]. For each word obtained after tokenization, if it matched a tag from our taxonomy, we assigned it as a tag to the corresponding video. Given that over 1.5 million tags were collected, we narrowed our focus to a subset. For every category, we gathered the 500 most used tags having a word length greater than 3 (to avoid single letters or conjunctions). We then took the intersection of all the tags obtained. This gave us 17 common tags that are used almost everywhere very frequently. We added these to our tags. Subsequently, for all the categories, we took the top 15 most common tags with a length greater than 3, which were not present in the common tags set. This process resulted in a total of 148 tags.

Upon performing an Exploratory Data Analysis (EDA) of our dataset, we inferred the following insights:

	categoryId	view_count	likes	dislikes	comment_count	region	duration	definition	comment_sentiment
count	450731.000000	4.507310e+05	4.507310e+05	4.507310e+05	4.507310e+05	450731.000000	450731.000000	450731.000000	450731.000000
mean	19.734764	5.016213e+06	1.256398e+05	3.502586e+02	4.035947e+03	5.877983	1294.606863	0.004486	0.155686
std	6.551173	2.379317e+07	5.177033e+05	6.478964e+03	3.784381e+04	3.114976	3017.209399	0.066828	0.193607
min	1.000000	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000	0.000000	0.000000	-0.911910
25%	17.000000	3.951655e+05	1.039700e+04	0.000000e+00	4.420000e+02	3.000000	220.000000	0.000000	0.001735
50%	22.000000	1.057772e+06	2.823200e+04	0.000000e+00	1.150000e+03	6.000000	668.000000	0.000000	0.122740
75%	24.000000	2.940004e+06	8.188250e+04	1.500000e+02	2.958000e+03	9.000000	1225.000000	0.000000	0.270210
max	29.000000	3.487457e+09	5.196805e+07	3.979409e+06	1.580316e+07	10.000000	86182.000000	1.000000	0.999970

Figure 1. Dataset Description

1. The mean view count is 5 million, but the high standard deviation (around 23.8 million) and a median of 1.05 million indicate a skewed distribution driven by a few viral videos. The maximum view count reaches 3.48 billion, showing exceptional popularity for some videos.

[2] https://github.com/cjhutto/vaderSentiment
[3] https://github.com/nltk/nltk

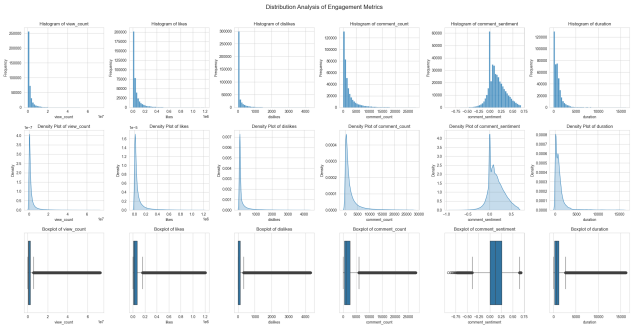

Figure 2. Distribution Analysis

2. The histograms, KDE density plots, and box plots similarly reveal that all the metrics exhibit significant skewness, indicating the presence of extreme values. However, `comment_sentiment` stands out with a more balanced distribution, leaning slightly towards the positive side, suggesting a generally favorable sentiment across video comments.

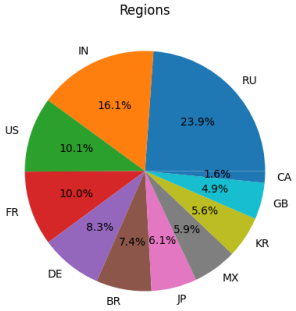

Figure 3. PieChart: regions

3. Russia (24.1%) and India (16.0%) dominate video engagements, suggesting high viewership or content production from these regions. US, France, and Germany, together, account for about 28.4%, indicating significant activity from Western countries.

4. Entertainment videos are the most popular (26.2%), indicating they drive a large portion of engagement. Categories like People & Blogs (13.4%), Gaming (13.1%), and Music (10.8%) also have significant shares, reflecting varied audience preferences.

4. Methodology and Model Details

For training the models, we selected a subset of features that we considered relevant for improving the performance of our regression models. These features include: `categoryId`, `tags`, `comments_disabled`,

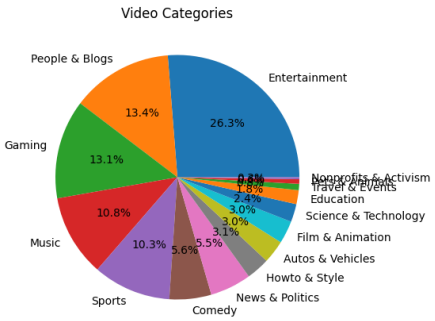

Figure 4. PieChart: category

`region`, `duration`, `definition`, `caption`, and `comment_sentiment`. The target variable for prediction was `view_count`.

To streamline the process, we utilized `LazyPredict`[4], a Python library that allows quick training and comparison of multiple machine learning models. In total, we trained 42 regression models using this library. The purpose of using LazyPredict was to rapidly evaluate a variety of models and select the top performers based on metrics such as Root Mean Squared Error (RMSE) and R-squared (R^2).

Subsequently, we focused on predicting the optimal tags that would maximize the views. The following algorithm was applied for tag prediction:

Algorithm 1 Tag Prediction Algorithm

```
changes ← "no change": prediction of views on current data
for each tag in data:
    if data[tag] ≠ 1:
        data[tag] ← 1
        changes[tag] ← Prediction of views on the current data
max_change ← key of changes which gives maximum views
if max_change == "no change":
    return
data[max_change] ← 1
Rerun the algorithm with updated data
```

This algorithm provides a sequence of tags that are best suited for maximizing views. We also applied max depth pruning to the Tag Prediction Algorithm, with a default depth value $d = 10$, which gives a time complexity of $O(n \cdot d)$, where n is the number of tags considered.

Next, we implemented a second approach based on an algorithmic model. To predict the best tag combinations that increase views, we needed to consider the subscriber count as it greatly influences the view count. Initially, we plotted the relationship between view count and subscriber count, but the dataset was noisy. To address this, we binned the data based on subscriber count and then plotted the median view count for each bin. This resulted in a logarithmic curve, and we applied logarithmic regression to fit the best curve to this plot. By doing so, we were able to normalize the view counts with respect to subscriber count, allowing us to focus on tag-related factors.

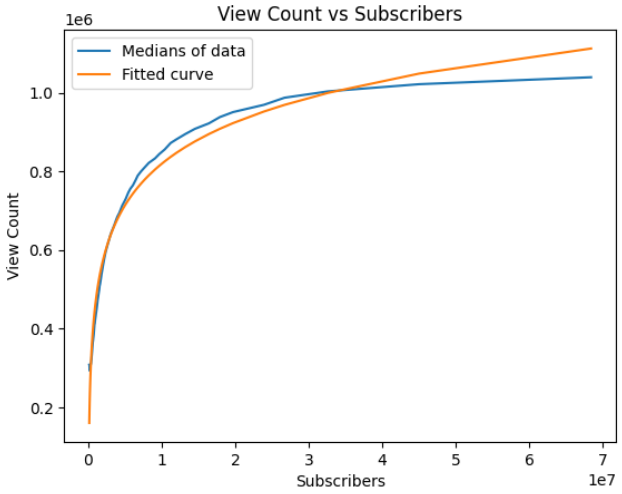

Figure 5. Dataset Description

The following algorithm was used to find the best tags for each combination:

Algorithm 2 Combination-Based Tag Analysis

```
Next_tags ← hashmap
for each combination in all combinations of tags:
    Changes ← Initialize to have "no tag": 0
    for each tag in tags:
        if tag not in combination:
            Subset ← subset of dataset with combination & tag
            Increase ← actual - expected
            Inc_sum ← sum(Increase)
            Changes[tag] ← Inc_sum
Next_tags[combination] ← key of changes which gives max increase
```

For each combination of tags, we iteratively added the next best tag based on the 'Next_tags' hashmap and reran the process until no further improvement was found. This results in a time complexity of $O(n \cdot 2^n)$.

Thus, both algorithms generate tags, with the machine learning-based approach having a lower time complexity due to its use of a greedy algorithm, while the algorithmic approach, based on the dataset, has a higher complexity but may provide more accurate results. In a manner similar to Maximum Likelihood Estimation (MLE), we generated tags that maximize video views by iteratively predict-

[4] https://github.com/shankarpandala/lazypredict

ing the most impactful tags and selecting the combination that yields the highest increase in views. This approach uses a greedy algorithm to assess and prioritize tag combinations, optimizing view count through a sequence of decisions based on model predictions and dataset analysis. The assumption in Algorithm 2 is that any increase in views is solely attributed to the tags.

5. Result and Analysis

A total of 42 regression models were trained, and their performance was evaluated using the Root Mean Squared Error (RMSE) and R-squared (R^2) metrics. The top five performing models, in terms of the RMSE metric, were: RandomForestRegressor, BaggingRegressor, ExtraTreesRegressor, GradientBoostingRegressor, and LGBMRegressor. The best performing model, **RandomForestRegressor**, achieved an R^2 score of 0.618, which indicates that approximately 61.8% of the variance in the target variable (*view_count*) can be explained by the model. The close values of adjusted R^2 and R^2 suggest minimal penalty for the number of predictors in this model. Below is a summary of the top five models' performance metrics:

Model	Adj. R^2	R^2	RMSE
RandomForest	0.6180	0.6182	15525132.5
Bagging	0.6159	0.6160	15568830.2
ExtraTrees	0.6139	0.6140	15609041.0
GradientBoosting	0.5699	0.5701	16473013.7
LGBM	0.5551	0.5553	16755405.1

We also ran GridSearchCV on the Random Forest Regressor but were unable to increase R^2.
We evaluated the efficiency of our model in comparison to the ground truth algorithm. Due to time constraints, we created a smaller dataset comprising 15 tags selected from our original dataset. The efficiency was calculated using the following formula:

$$\text{Efficiency} = \frac{|S_g - S_l|}{|S_g|}$$

Where:

- S_g: Total tags predicted by the ground truth algorithm.
- S_l: Total tags predicted by the machine learning approach.

We computed the average efficiency across the first 50 samples in our dataset, resulting in an efficiency value of **89.31%**. This suggests that the machine learning-based approach outperforms the ground truth algorithm significantly in terms of speed while still maintaining a reasonably good efficiency.

Additionally, we developed a graphical user interface (GUI) using the `tkinter` library. The GUI provides a practical demonstration of the algorithm's input and output, illustrating how the model operates in real-world applications.

6. Conclusion

6.1. Learnings and Key Findings From The Project

Through this project, we gained insights into working with Google Cloud APIs, utilizing YouTube API efficiently, and managing API quotas. We explored sentiment analysis of social media text, extended a tag taxonomy. Random Forest Regressor achieved a R^2 score of 0.618, demonstrating that the model can explain a substantial portion of the variance in video views. This result highlights the importance of features such as category, region, and sentiment in influencing view counts.

A key contribution of this project was the development of a machine learning-based tag suggestion mechanism. Our approach offered a significantly more time-efficient solution for predicting the best tags to maximize video views while also maintaining reasonable efficiency. While both methods provided useful tag combinations, the machine learning-based approach proved to be more scalable, offering a good balance between efficiency and predictive accuracy.

6.2. Limitations

While the Random Forest Regressor showed promising results, the model only achieves a moderate fit and thus still leaves room for improvement. GridSearchCV on the Random Forest Regressor was unable to increase the R^2, suggesting that the current hyperparameter settings or the features used might already be optimized. The hyperparamter turning and model training also had a very hard limitation of the available compute power, since the training and validation steps were proving to require large amounts of ram, which we are currently not in possession of.

6.3. Future Work

Use of neural networks in general and CNNs in particular to predict video performance based on thumbnails. Additionally, generative AI and LLMs could be used to suggest not just tags for the video but also generate video titles, video description and other metrics that could improve video performance. Such DL techniques can also be utilized to characterize the entire transcript of the video in order to suggest improvements and marketing of any video.

6.4. Member's Contribution

1. Devansh Grover: Literature Review, Model Training, Tag Taxonomy
2. Manan Aggarwal: Develop/Gather Dataset, Model Training, EDA, Tag Expansion Algorithm
3. Shobhit Raj: Literature Review, EDA, Tag Expansion Algorithm
4. Vashu: Dataset Pre-processing, Result & Analysis, GUI

References

[1] Vandit Gupta, Akshit Diwan, Chaitanya Chadha, Ashish Khanna, and Deepak Gupta, *Machine Learning enabled models for YouTube Ranking Mechanism and Views Prediction*, arXiv preprint arXiv:2211.11528, 2022. Available at https://arxiv.org/abs/2211.11528.

[2] Gurjyot Singh Kalra, Ramandeep Singh Kathuria, and Amit Kumar, "YouTube Video Classification based on Title and Description Text," in *Proceedings of the 2019 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS)*, 2019, pp. 74-79. DOI: 10.1109/ICCCIS48478.2019.8974514.

[3] YouTube API V3. Available at https://developers.google.com/youtube/v3/docs.