Popularity Analysis: Factors Influencing Video Engagement Among Top YouTube Creators

Mason Hu

March 16, 2024

Abstract

I utilized my web scraping skills and data analysis skills learned in JSC370 to this extensive data analysis project of YouTube giants popularity. In this paper, I performed end-to-end data mining and web scraping, and light preliminary data analysis for the YouTube data. This project mainly acts as a stepping stone towards the final project, which involves more intensive data analysis, and is coming in a month. However, I still concluded that subscriber count has a (obviously) positive correlation with view counts; the most viewed videos on average are of the comedy and technology category; the best days to publish videos are Saturdays and Sundays; and that likes are a more reliable indicator to a videos popularity than comments. The datasets are publicly available on GitHub.

Contents

1 Introduction

YouTube has been a platform we cannot live without. Whether we want to follow the sports news or reinforce a concept you were taught in class, or perhaps even watch some recreational skits, we can't manage without YouTube. Many many people has made a living doing YouTube, and today we are going to focus on those who are exceptionally successful on this indispensable platform.

Among the top 1000 YouTube giants, YouTubers have videos with high view counts and low view counts. In this midterm data analysis project, I will determine why some videos created by popular YouTubers with an absurd amount of subscribers sometimes still get low view counts, and what in general are the specific factors that drive viewers to view videos.

My research questions:

- What factors lead to high video view count?
- What in turn lead to low view count, despite the YouTuber having high subscribers count?

To answer this question, I turned to Kaggle for datasets. However, the datasets there are scarce, and most of them only has the basic columns with no insightful attributes. Instead, I used only one column of a Kaggle dataset—the links to the channels of top 1000 YouTubers—and started my web scraping journey towards two other huge datasets that I extracted from the web. Guess what, it was not easy.

Details of web scraping are discussed below.

2 Methodology

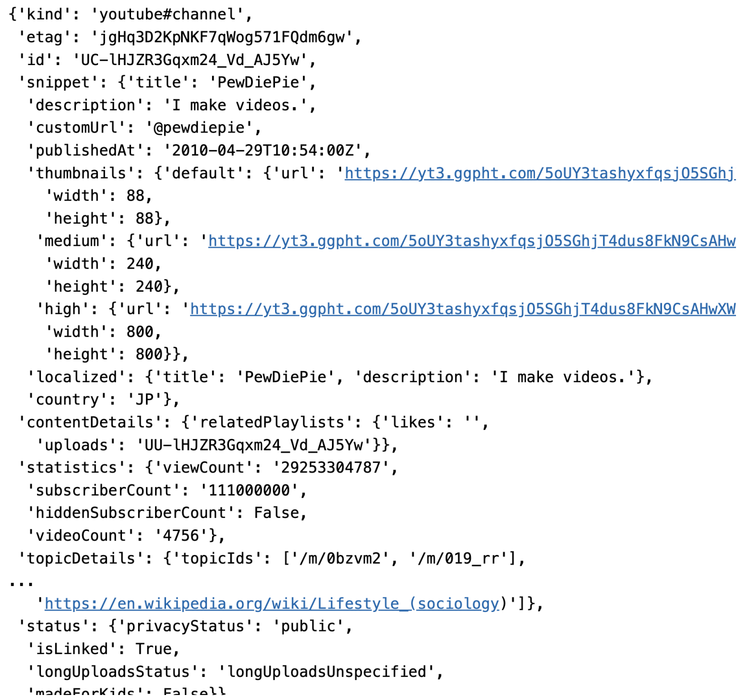
2.1 Data acquiring:

The initial dataset is acquired from Kaggle, linked in references. I only used one column of the data to find out who exactly are the top 1000 YouTubers. This data is more like a entryway to my two main datasets I scraped.

To effectively scrape YouTube, we have to first be able to understand the YouTube Data api. To that end, I spent 3 hours just merely reading the guidelines and references and playing around with some of the api calls. I then made numerous helper functions to extract the statistics from the responses by the requests I made to the api.

After acquiring the top 1000 YouTube channels from the Kaggle dataset and understanding the api, I tried extracting the channel informations by accessing the channels through their channel ID. However, that did not end up working as straightforwardly as I thought. Many channel IDs are invalid in the Kaggle dataset and I had to make supplementary calls using channel usernames as secondary information. In addition, I had to also check if the channel's subscriber count is large because I once encountered a MrBeast channel with only 1960 subscribers.

I ended up scraping 250 valid YouTube channels. But unfortunately, they are in the form shown below:



```
{'kind': 'youtube#channel',
 'etag': 'jgHq3D2KpNKF7qWog571FQdm6gw',
 'id': 'UC-lHJZR3Gqxm24_Vd_AJ5Yw',
 'snippet': {'title': 'PewDiePie',
  'description': 'I make videos.',
  'customUrl': '@pewdiepie',
  'publishedAt': '2010-04-29T10:54:00Z',
  'thumbnails': {'default': {'url': 'https://yt3.ggpht.com/5oUY3tashyxfqsjO5SGhj
    'width': 88,
    'height': 88},
   'medium': {'url': 'https://yt3.ggpht.com/5oUY3tashyxfqsjO5SGhjT4dus8FkN9CsAHw
    'width': 240,
    'height': 240},
   'high': {'url': 'https://yt3.ggpht.com/5oUY3tashyxfqsjO5SGhjT4dus8FkN9CsAHwXW
    'width': 800,
    'height': 800}},
  'localized': {'title': 'PewDiePie', 'description': 'I make videos.'},
  'country': 'JP'},
 'contentDetails': {'relatedPlaylists': {'likes': '',
   'uploads': 'UU-lHJZR3Gqxm24_Vd_AJ5Yw'}},
 'statistics': {'viewCount': '29253304787',
  'subscriberCount': '111000000',
  'hiddenSubscriberCount': False,
  'videoCount': '4756'},
 'topicDetails': {'topicIds': ['/m/0bzvm2', '/m/019_rr'],
...
   'https://en.wikipedia.org/wiki/Lifestyle_(sociology)']},
 'status': {'privacyStatus': 'public',
  'isLinked': True,
  'longUploadsStatus': 'longUploadsUnspecified',
  'madeForKids': False}}
```

Figure 1. JSON Formatted Response

2.2 Cleaning and Wrangling

I haven't encountered JSON formatted strings before, so this is the first time I had to deal with monsters like them. However, it turns out that they are pretty straightforward and tedious rather than perplexing. After knowing how they are structured, a few more helper functions is all it took to turn them into a clean table with the following columns:

Table 1. Columns for Channels data frame

Column	**Count**	**Type**	**Description**
Channel INFO	245	object	JSON formatted response
Id	245	object	Unique identifier for the channel
username	245	object	Username of the channel
handle	245	object	Handle of the channel
subscribers	245	int64	Number of subscribers
views	245	int64	Total number of views
videos	245	int64	Total number of videos
country	197	object	Country the channel is in
description	245	object	Description of the channel
publishedAt	245	datetime64	When the channel was published
topics	244	object	List of topics associated
topicsSuper	244	object	Super categories of topics
topicUrls	244	object	URLs of topics
madeForKids	205	object	True if made for kids
playlistId	245	object	Access to the channels videos
hiddenSubscriber	245	bool	True if subscribers are hidden
isPublic	245	bool	True if public
isLinked	245	bool	True if channel is linked
longUploadsStatus	245	object	Status of long uploads
thumbnailDefault	245	object	URL of the default thumbnail
thumbnailMedium	245	object	URL of the medium-sized thumbnail
thumbnailHigh	245	object	URL of the high-quality thumbnail

The whole process is repeated scrutinously with video data (accessed and scraped through playlist ids). The column descriptions are attached in appendix for better paper browsing experience. In the end, we have a 245*22 data frame—Channels (906 KB), and a 11712*20 data frame—Videos (17.8 MB). This moves us on to the wrangling process.

2.2.1 Hashing of topics and super topics

Of course, I also did some data wrangling to achieve this. For example, the topics of the channels were originally provided as topic ids, like "/m/018w8". However, I found this poorly formatted table in 2017 page of YouTube data api that maps each topic id to its topic e.g. "/m/018w8" means basketball. So I copied the raw outlines of this table and stored them in a txt file, which I then read through in Python to produce two neatly structured hashmaps, one that maps ids to categories and one that maps categories to its supercategories.

2.2.2 Duration extraction using Regex

Due to the large limit on video size provided by YouTube, videos can have lengths of hours and even days. The existing `datetime` library does not have a robust bug-free extraction method for the format provided by YouTube. Hence, I wrote regex helpers using the pattern

$$r'PT(?:(d+)H)?(?:(d+)M)?(?:(d+)S)?'$$

And successfully transformed the statistic into integer valued number of seconds.

2.2.3 Log transforms

Since there are a few YouTubers who achieved over 100 million subscribers and most YouTubers have around 20 million, the subscribers distribution is highly skewed. I decided to use a log transform on them. In addition, other variables like view count, likes, comments, and even video duration also follows a similar exponential distribution. Hence, they are all transformed accordingly for a better distribution.

2.3 Data exploration tools

I used seaborn and matplotlib libraries for data visualizations, and I also used scipy, pygam for spline regression analysis. Details are in the immediate next section.

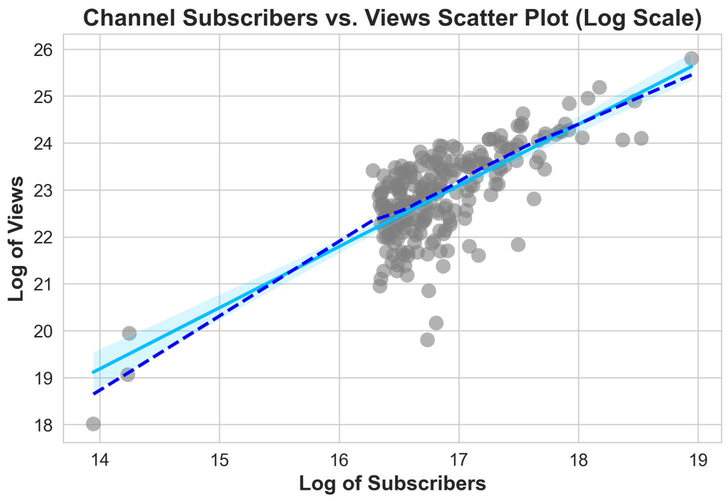
3 Preliminary Results

Exploratory data analysis and preliminary regression has been done. A plethora of plots have been plotted. In this section, I will put forth the most convincing plots that has significance, and omit the unnecessary plots like boxplots of subscriber count.

Below are 4 results I find most powerful and clearest.

3.1 Subscriber count

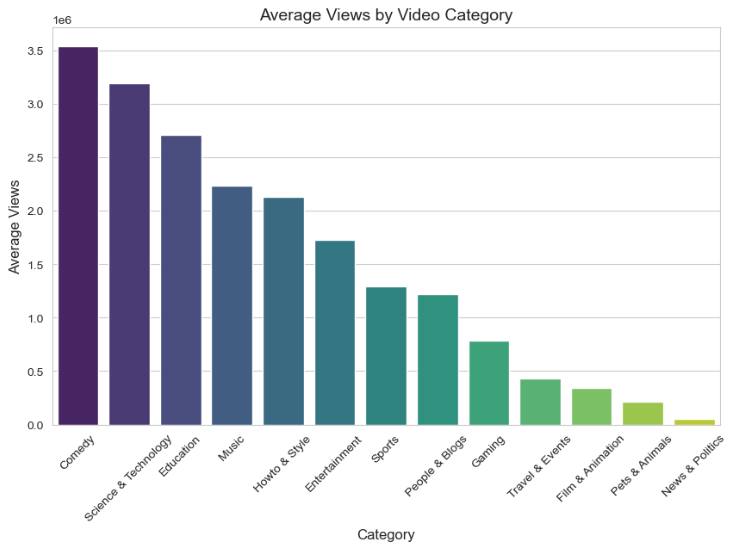
As shown in the below graph, subscriber count and views of a channel is positively relative with a specific log-linear relationship. The dashed blue lines



represent the non-linear smooth curve through the points, and the light blue straight line with error bars is the linear smooth line.

3.2 Category

In the below graph, We see that the top three categories with most average views



per videos is Comedy, Science & Technology, and Education. Averging over 2.5 million views per video (the videos are among the top 1000 YouTubers so this makes sense). This is not surprising since nowadays, people crave happiness, but at the same time has to go to school and follow the technology news.

3.3 Publish time of week

As we can see in the graph below, Thogh by only a small margin, Fridays are

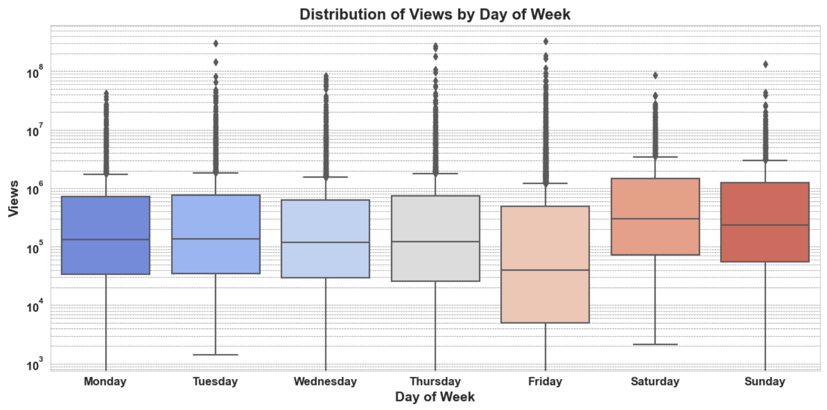


Figure 2. Enter Caption

the worst days to publish a youtube video and Saturdays and Sundays are the best, on average. This is surprising. Do people not have more time to watch videos on a day they get off work on? Maybe peole like to go clubbing on that day instead.

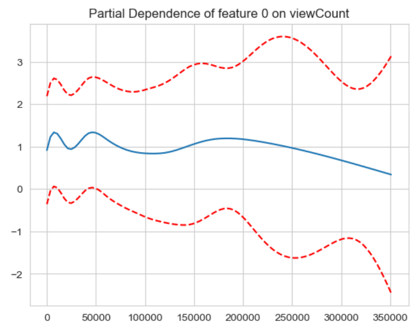
3.4 Views and Comments

As shown in Figure 3 below, two key insights emerge:

Comments: Looking at the number of comments (Feature 0), we see that more comments usually mean more views, at least up to a point. After that point, getting even more comments doesn't always lead to a big increase in views.

Likes: On the other hand, likes (Feature 1) have a strong and steady link to the number of views. The data shows that as likes go up, views tend to sharply rise as well, although the rate of increase slows down after a certain level. Yet, unlike comments, likes consistently relate to more views across the board, making them a more reliable indicator of a video's popularity.

The dashed red lines that frame the main line on the graphs show us the confidence intervals, and they get wider as the number of comments or likes increases.



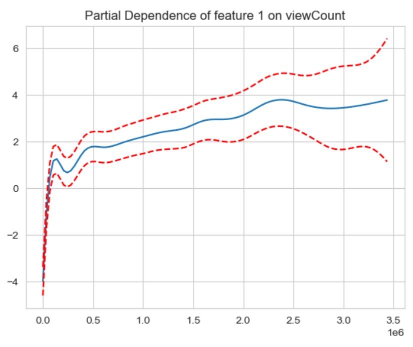


Figure 3. Spline model visualization

This means the more comments or likes we get, the less certain the model is about its predictions, especially when these numbers are much higher than average.

4 Summary

Since results statistics are already mentioned in the last section, lets us devote this section fully to some of the issues and crucial aspects regarding this project.

4.1 Difficulties and Limitations

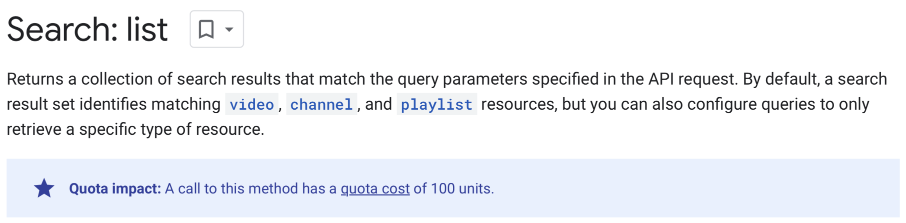
4.1.1 Still Limited data

Channel IDs are hard to retrieve. The Kaggle dataset, as mentioned above, has possibly expired channel ids, making it hard to retrieve the channel information. Handles are also not available on any datasets online, so we cannot retrieve the info using handles. In addition, usernames are usually not useful since they are poorly formatted in the dataset, and we have to match the true username exactly. There are also no -list-top-1000-youtubers methods in the api, which would have made things so much easier.

The two ways to work around this that I came up with are: extracting the channel ids manually on Google Chrome page source (obviously impractical), or using search.list method in api.

Using the search.list method, we can derive a list of 10 prospective responses to the keyword "Mr Beast" and extract the one with type channel and subscribers count over 10 million.

However, A call to this method has a quota cost of 100 units instead of 1 as



Search: list

Returns a collection of search results that match the query parameters specified in the API request. By default, a search result set identifies matching `video`, `channel`, and `playlist` resources, but you can also configure queries to only retrieve a specific type of resource.

Quota impact: A call to this method has a quota cost of 100 units.

other calls do, which uses up out the daily limit (10,000) in no time.

This difficulty limited my channels dataset to only around 250 videos.

4.1.2 Description Quality

The descriptions of videos are hard to read, filled with external links and sponsor advertisements, and filled with repetitive and desperate requests for people to subscribe. This makes it hard for us to do natural language computing and text mining on them. However, with enough time and fine grained mining, I believe in the final project there would be breakthroughs in this aspect of the project.

4.2 Advantages and Future Work

4.2.1 Web scraping paid off

The dataset I spent my whole week scraping is definitely the pinnacle of this project. The two datasets I produced are reliable—straight from the YouTube api with actual working channel IDs, up-to-date—has videos uploaded even today (Mar 15th), insightful—holds descriptions of channels and other important informations unlike those found online, and potential—discussed in this next sections.

As the by-product of this project, I also produced optimal and well documented functions for scraping YouTube, for accelerated future project deployments.

4.2.2 Using CNNs and thumbnails to predict popularity

Thumbnails are not used in this project. However, they yield great potential in classifying popularities of videos and channels. In the final project, I would gladly build a CNN predictor that somehow predicts the view count of the videos based on the thumbnails, which are in the dataset and ready to be accessed. This might seem dumb, but it is definitely a potential field to explore.

4.2.3 Using transformers to investigate topics and/or descriptions

One other thing we did not use are the topic urls. They link to the wikipedia page of topics of interest. This might be super meta, but what if we can predict based on the wikipedia page. This, including further analysis of video/channel descriptions are highly potential future directions worth giving a try.

4.2.4 Integrating SQL

There are tons of meta information like comments, lists of tags and supertags. What if we can expand the database and fully connect every information of a youtube video. This cannot be done without SQL and a big chunk of the rest of my semester. However, if we can accomplish that, it would be huge for making categorical analysis and decision tree models.

4.2.5 Spatial-temporal Analysis

In this final project, if we are asked for interactive plot, one great idea is to analyze those data geospatially. We already have the countries of each channel, and we also have the times a video is published and a channel is published. We can then merge other supplementary datasets that produce latitude and longitude information for example, to produce creative interactive plots, and do geo-temporal analysis on.

References

Kaggle dataset: https://www.kaggle.com/datasets/syedjaferk/top-1000-youtubers-cleaned?rvi=1