# Overarching

Imports uses common build for consistency and ability to quickly create and conform new notebooks

Scope: This analysis is restricted to videos tagged by YouTube as Music.

# ETL

General: <https://developers.google.com/youtube/v3/docs/>

Categories: <https://gist.github.com/dgp/1b24bf2961521bd75d6c>

This stage is built using the YouTube Data API, v3.

Most YouTube API features are targeted at uploaders, and channel owners in particular. Many features, including the majority of Analytics, are restricted to the owner of a video or channel.

In general, the API references a custom function (build) to construct the query along with authorization. The build type can be stored. Search methods are then appended to this to return information.

I restricted my search to category 10: music. This is a tag that is assigned by YouTube at upload

* search().list
  + Takes some kind of search term as primary input
  + works like typing a query into the youtube search bar, returning a json file that has high-level summary information
  + https://developers.google.com/youtube/v3/docs/search/list
* videos().list
  + Takes unique video id as input
  + gets summary statistics of specific video, along with detailed

Token Costs

* Search: List - 100 tokens

The main search was executed by searching relevant videos for each letter of the alphabet to get as wide a cross reference as reasonable. Each letter returned about 320 results.

Beyond down selecting data at the query, some additional cleaning was done in cleanup. When the model was first done, I had to break halfway through the clean up, so the working model was saved.

Next came interpreting. For example, duration is stored as text in the format PT #H #M #S, with 0 values not stored. For example, 2 hours 23 minutes is PT2H23M, and 1 hour 1 second is PT1H1M.

Some basic feature engineering was done here as well, including adding some integer value date time features, and the conversion of duration, licensedContent, and caption to integers:

* Definition (video quality): 0 for standard def, 1 for high def
* licensed Content (whether the content is claimed under copyright): 0 for none, 1 for yes
* caption (has video caption to display in query): 0 for false, 1 for true

Dropped some unnecessary items

* categoryID: category10 was stored for all videos
* channelID: outside of scope
* dimension: whether video was 2D or 3D. Only 3 or 4 HD videos existed in the set

Duplicates (by vidId) are also dropped. Given the very loose method of searching for videos, I was pleasantly surprised to see that out of about 10,000 records scraped, about 8600 unique videos remained.

Note: This process was repeated when pulling new videos to test the model against.

Not included, unfortunately:

Time series data is heavily restricted. For example, comment and view count by date would have been interesting. Google trends could have served as an approximation, but querying this programmatically requires establishing a specific login protocol. Loosely speaking, it is worth noting that most videos with much more than a million views appear to get the majority of their views in the first month or so after publishing.

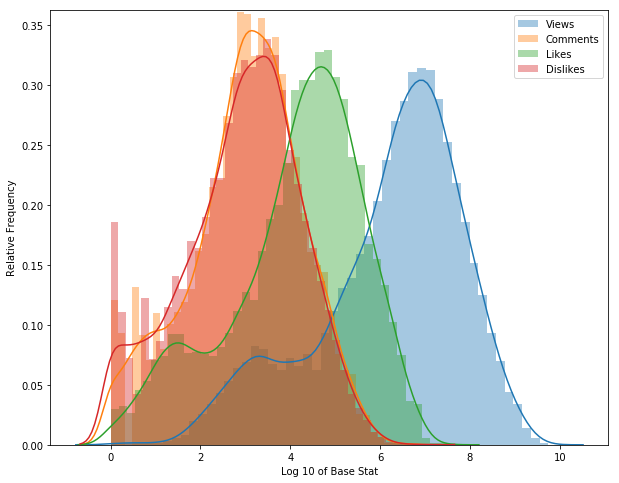
# Exploratory Data Analysis

## Part 1

Some basic facts

* Mean / Median views: 55 M / 3 M

The very first thing I noticed about the data at this point is the spread along orders of magnitude. Since part of the goal of this project involves predicting whether a video is a hit or not, decided to transform the data by taking the base 10 log of the summary statistics to make it easier to interpret results (than using square root or natural log, for example). The next chart is striking.



Some basic info here:

|  |  |  |  |
| --- | --- | --- | --- |
| **Log** | Mode | Bump | Mode / Bump |
| Views | 6.96 | 3.25 | 2.14 |
| Likes | 4.40 | 1.60 | 2.75 |
| Dislikes | 2.94 |  |  |
| Comments | 3.13 |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Log Ratio | Views | Likes | Dislikes | Comments |
| Views |  | 1.58 | 2.37 | 2.22 |
| Likes | 0.63 |  | 1.50 | 1.41 |
| Dislikes | 0.42 | 0.67 |  | 0.94 |
| Comments | 0.45 | 0.71 | 1.06 |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Actual** | Mode | Bump | Mode / Bump |
| Views | 9.08E+06 | 1.78E+03 | 5105 |
| Likes | 2.53E+04 | 3.98E+01 | 635 |
| Dislikes | 8.71E+02 |  |  |
| Comments | 1.34E+03 |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Views | Likes | Dislikes | Comments |
| Views |  | 3.6E+02 | 1.0E+04 | 6.8E+03 |
| Likes | 0.28% |  | 2.9E+01 | 1.9E+01 |
| Dislikes | 0.01% | 3.44% |  | 6.5E-01 |
| Comments | 0.01% | 5.31% | 154.17% |  |

Key observations:

* Log likes are a pretty direct function of the view count (63% log to log, or .3% in actual terms)
* The comment count is 50% higher than the dislike count
* Comment / dislikes can be described almost completely by likes, which can be described by views (in log space)
* The Like to View ratio tapers off as views increase because it is described in log space

For example, let’s look at Venom, by Eminem and Despacito, by Louis Fonzi (as of 10/18/18).

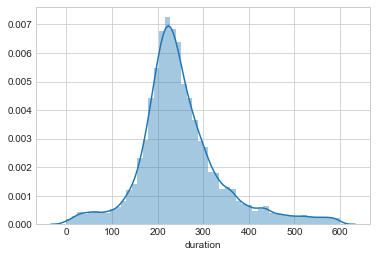
The stats look like this:

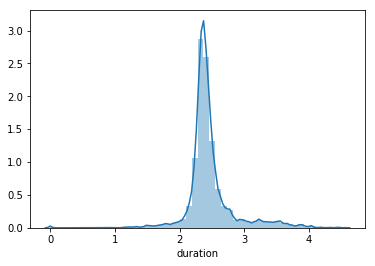
* Venom - Views: 80 M, Likes: 2.9 M, Dislikes: 100k , Comments: 185 k
* Despacito - Views: 5.6 B, Likes: 30M , Dislikes: 3.5 M, Comments: 2.6 M

Despacito has 70 times as many views as Venom, but only 10 times as many likes. In both cases dislikes are within 50% of the comments (it looks like a lot more people dislike Despacito than we would expect - more on that later).

Eminem was set up for failure, however. He could have tripled his views if he had a featuring in artist. Videos with some variant of ‘featuring’ in the title have an average of 129 M, compared to 55 M overall and 43 M for videos without a featuring artist. This works out to an average multiplier of almost exactly 3, which I dub the Pitbull effect. Pitbull has made a career out of being featured in other peoples songs, is incredibly crisp for some reason, and no one is totally sure why it works but it does.

Note: This affect is actually probably larger since I’m not sure exactly how it works in the Spanish language world. Again, like Pitbull.

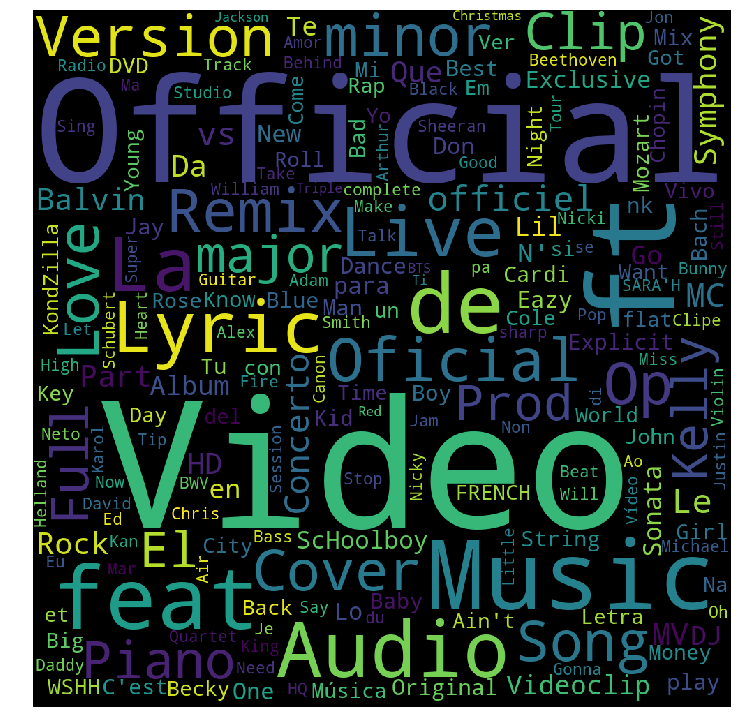




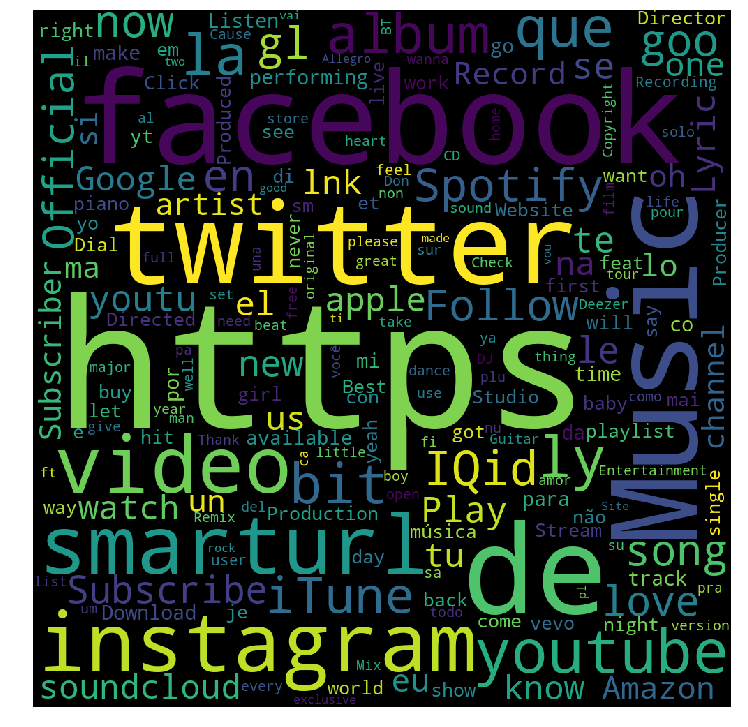
Way more videos over 10 minutes (~3 on the log scale) than expected. Turns out people like to have their study jams on YouTube. 5% of all videos (about 400 total in this set) are over 20 minutes long.

## Some Text Stuff: Title vs. Tags vs. Description

A YouTube video can be posted with 3 text categories, and I wanted to know if there’s any real difference. Lets look.







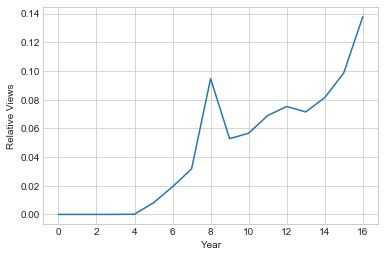
In brief, the title gives a broad description of the video category (lyrics, live, official, cover),

tags contain helpful genre or content information, and the description is full of self-promotional trash.

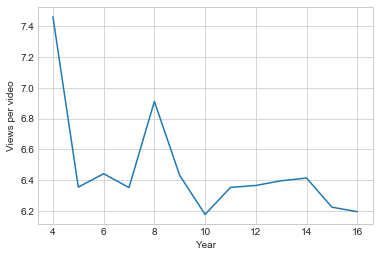
(J Balvin and people named Kelly are popular, love songs as always)

Overall, the sentiment score is pretty similar on average between these, and is unsurprisingly positive. The word bag used for this did not include Spanish, so it may not be totally accurate, although the score should still be informative in relative terms.

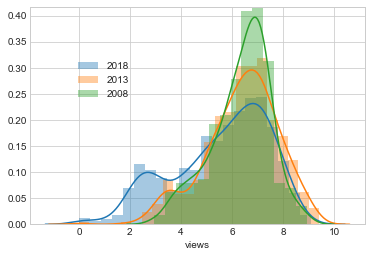
## Time Series



Relative views per year. Mainly reflects increase in videos uploaded per year.



2008 - the golden era. Interestingly, average views per video has stayed above comfortably above 1 million despite the increase in total videos uploaded per year over time. Mainly, this is due to the videos in the set being more broadly distributed, and having more low end answers:



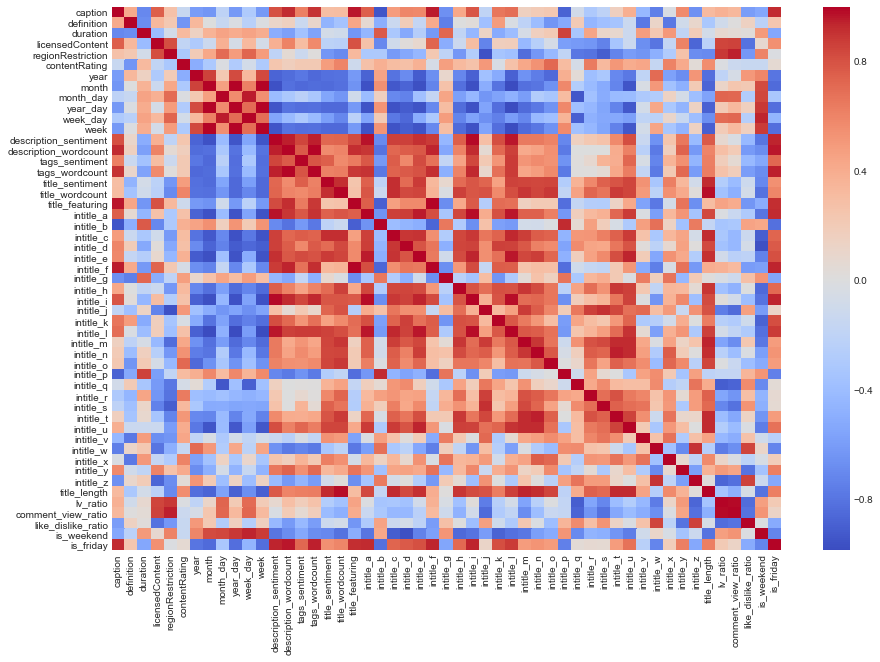
The cause of the low end bump is more apparent here. The search query method used to scrape the videos was far from perfect, as I was looking for individual letters by relevance. I believe YouTube uses views and upload date when determining relevance, and so the low count videos generally have some single letter in the title somewhere and are more recent. For example:

|  |  |
| --- | --- |
| **Views** | **Title** |
| 240 | B tone - LEAVE STORY Official Video1 |
| 159 | WHY HOLD BACK - Trunk Boiz member B\*Janky |
| 112 | Jonathan C. Meier - Savage (Audio) |
| 58 | HeartLand Didgeridoo - key of C - (4701) |
| 157 | Genomineerden gouden C 2017 |
| 26 | DELAIN /SING TO ME /c/ MARCO HIETALA 6/ en MAL... |
| 87 | Tram,Bix and Lang - For no reason at all in C ... |
| 270 | Catalogue d'Emojis Teaser |

## Part 2

In this phase, I looked at all my features to see which ones might be providing confusing statistics to the model. Refer to the Excel file in /background/supplemental for the full table. Basically, I went through all the variables to see which had distinctive means between each view class to see which might have real explanatory power.

In addition, after doing some down select, I Iooked at cross correlation to see if there was some overlap.



Some items of interest here are the correlation of a lot of variables over time, presumably because people interact with youtube differently now. For example, title lengths have gotten shorter, presumably as more people learned how to use tags. Similarly, sentiment goes with length, since sentiment is calculated as the sum of the sentiment of each individual word.

This suggests we need to focus on models robust to feature correlation, as well as removing most of the time series and some of the text analysis features.

# Feature Engineering

# Model

Ratio explanatory power -

with open('ss\_title\_tag.pkl', 'wb') as file:

pickle.dump(ss, file)

lr = LogisticRegression()

lr.fit(X\_train\_sc, y\_train)

with open('log\_reg.pkl', 'wb') as file:

pickle.dump(lr, file)

gb\_reg = GradientBoostingRegressor()

gb\_reg.fit(X\_train\_sc, y\_train)

with open('gb\_regr.pkl', 'wb') as file:

pickle.dump(gb\_reg, file)

preds = lr.predict(X\_test\_sc)

cm = confusion\_matrix(y\_test, preds)

pd.DataFrame(cm, columns=range(2,10), index=range(2,10))

from sklearn.metrics import r2\_score,confusion\_matrix, roc\_auc\_score, roc\_curve

sns.heatmap(pd.DataFrame(cm, columns=range(2,10), index=range(2,10)))