Team Trending

November 16, 2020

Overview

Focus Area: Entertainment Industry

Digital Marketing & Youtube Monetization

- Determine Factors that Influence YouTube Trending Videos
    - Publishing date, like, dislikes, comments.
- Categorise Trending Videos
- Understand Current Trends
- Predict Popularity

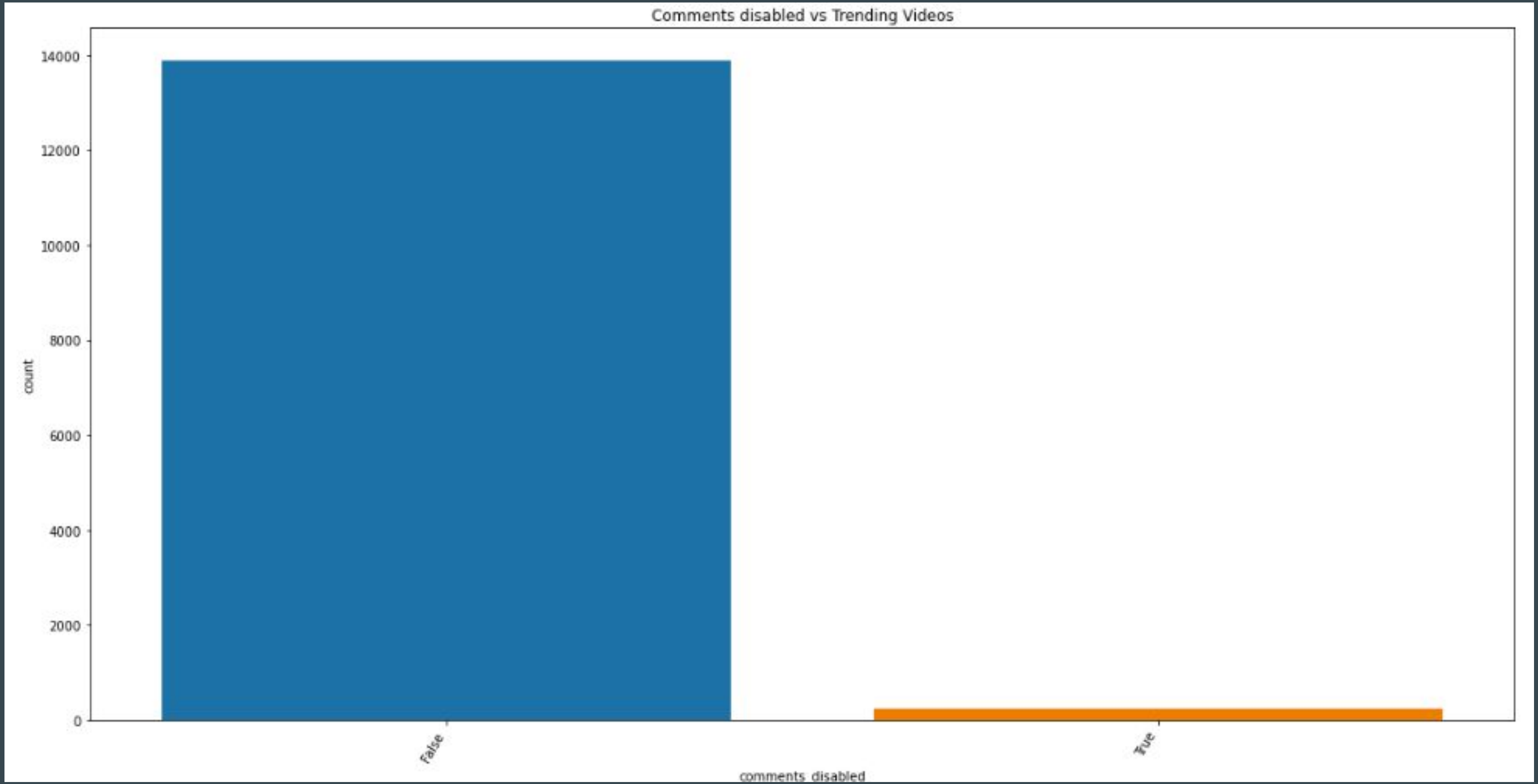
Clustering

...

Clustering

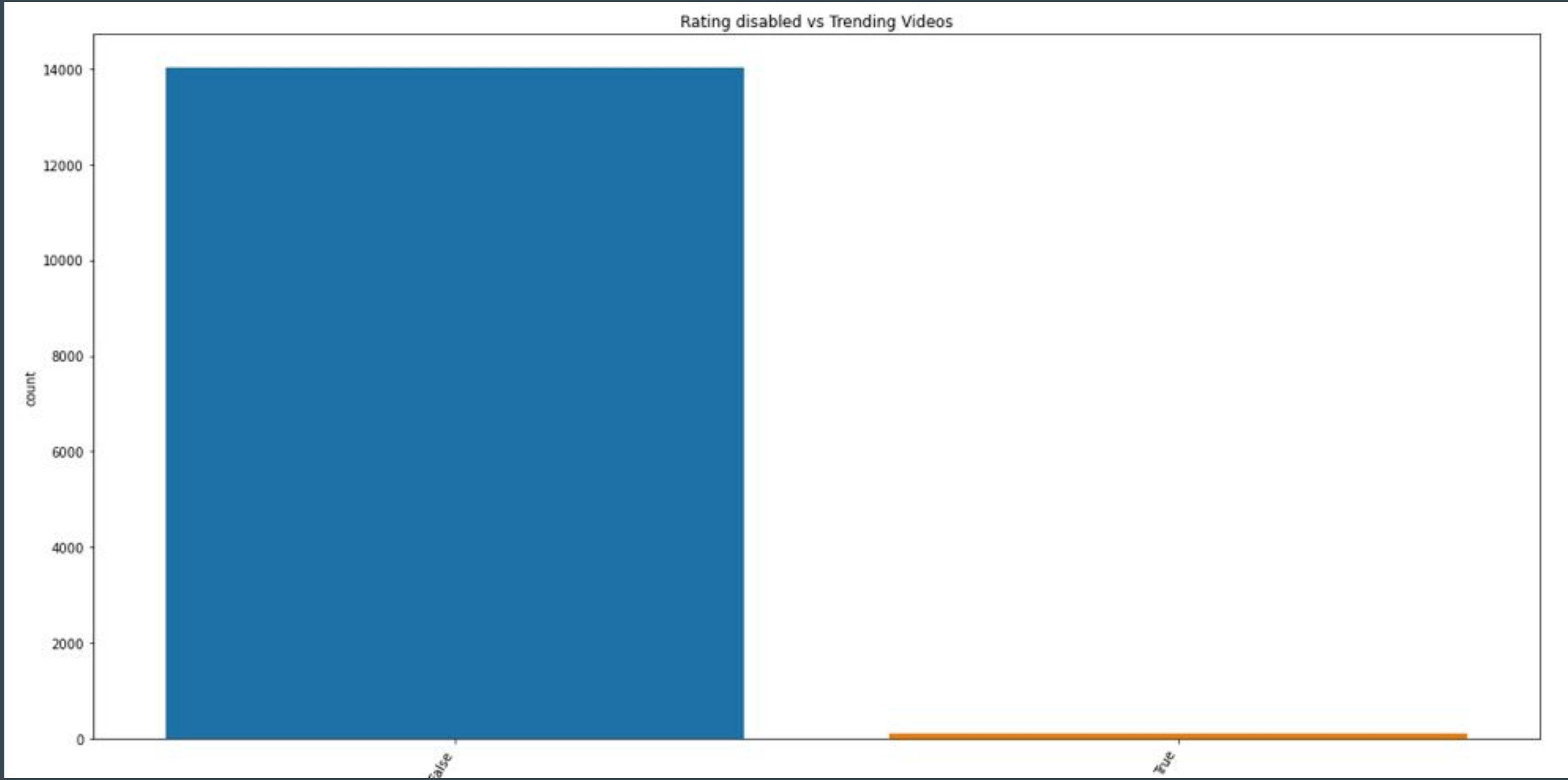
01	Data Cleaning & Data Formatting	• Dropping NaN rows • Mapping the categories to category ID • Deleting unwanted columns after visualization
02	Visualization	• Category vs Trending videos • Top 10 words for each category
03	Feature Engineering	• Extracting tag count for each trending video. • Computing percentage like, dislike and comment with respect to view count. • Observing distribution plots for the same.
04	Clustering	• K-means: Category, percent like, percent dislike, percent comment • DB-Scan: Tag count, percent like, percent dislike, percent comment

Data Cleaning

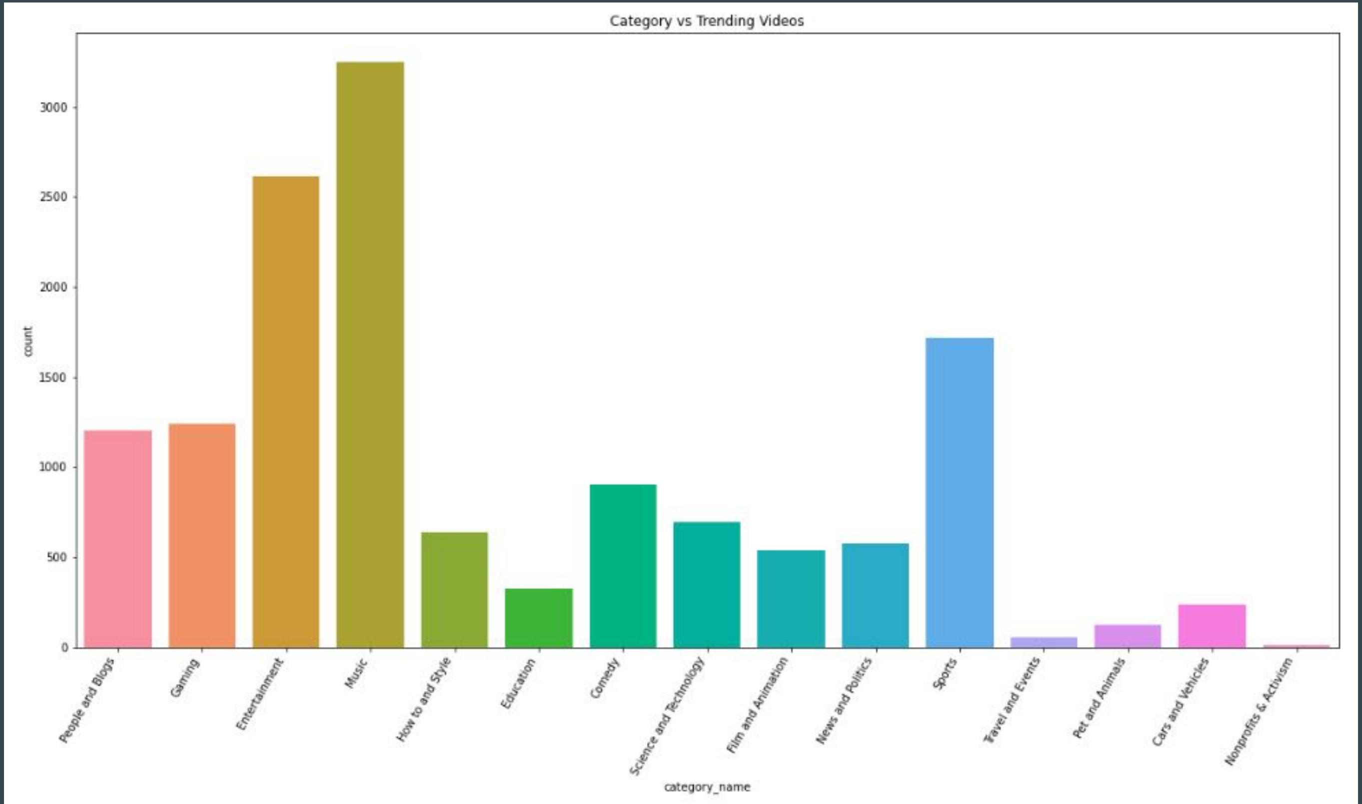


Data Cleaning

Other cleaned columns:
Thumbnail_link, description, title, video_id



Data Visualization: Category vs Trending Videos

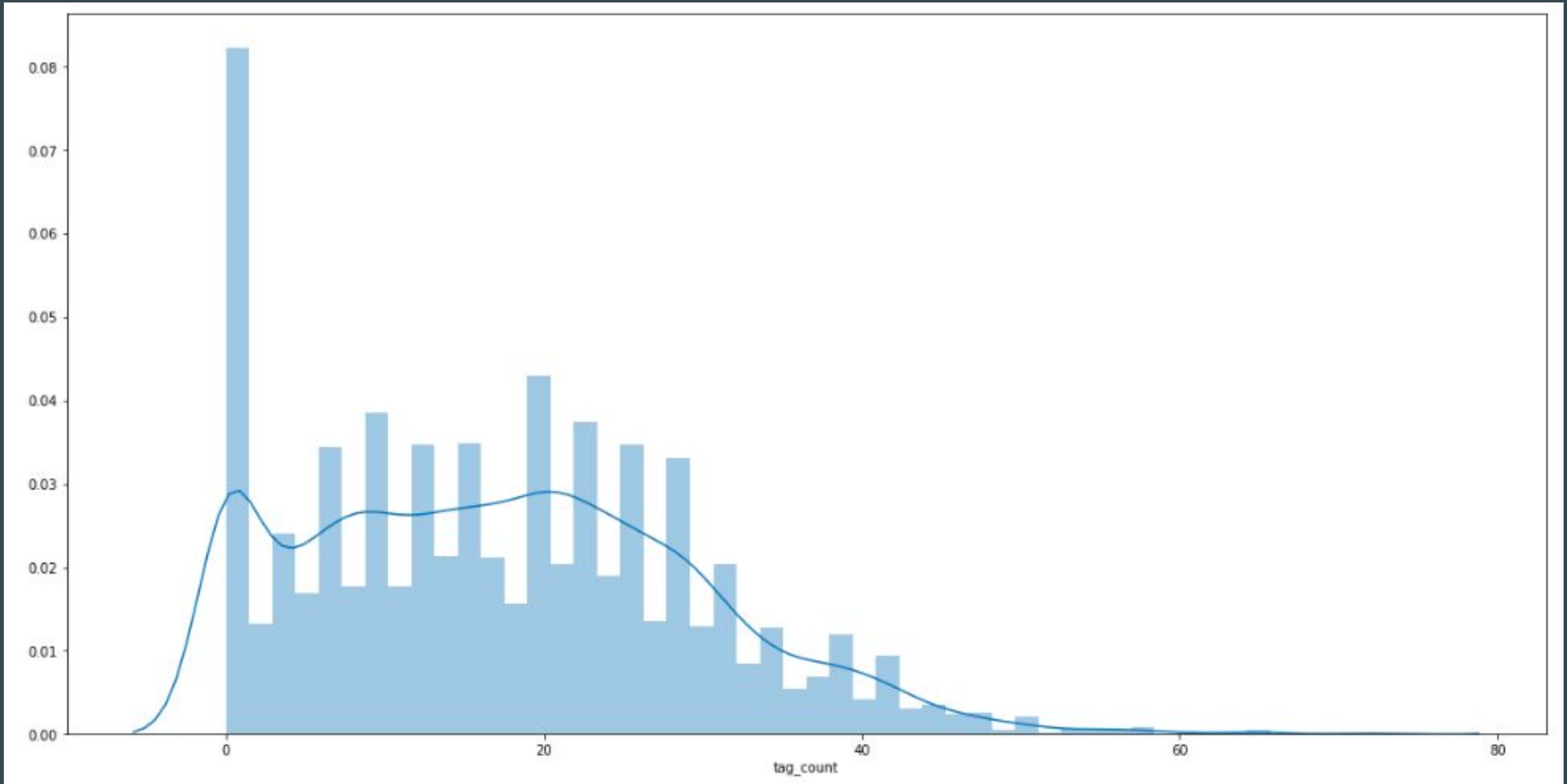


Top 3 Trending Categories:

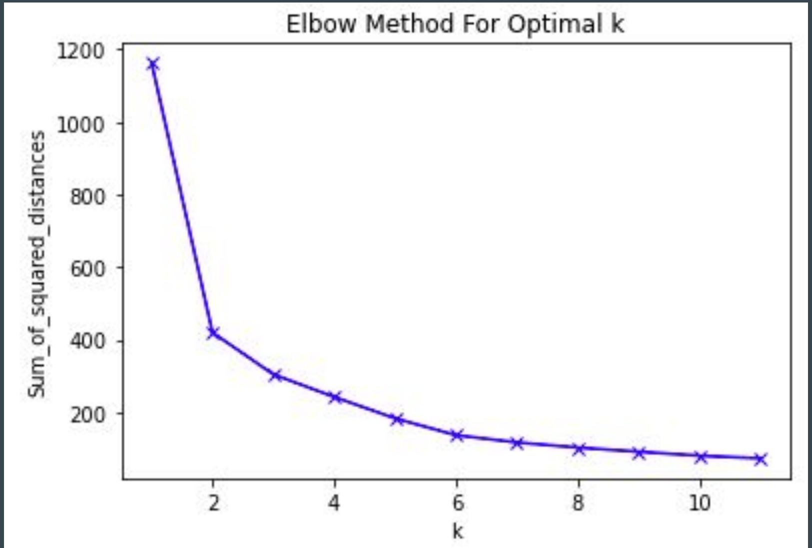
1. Music
2. Entertainment
3. Sports

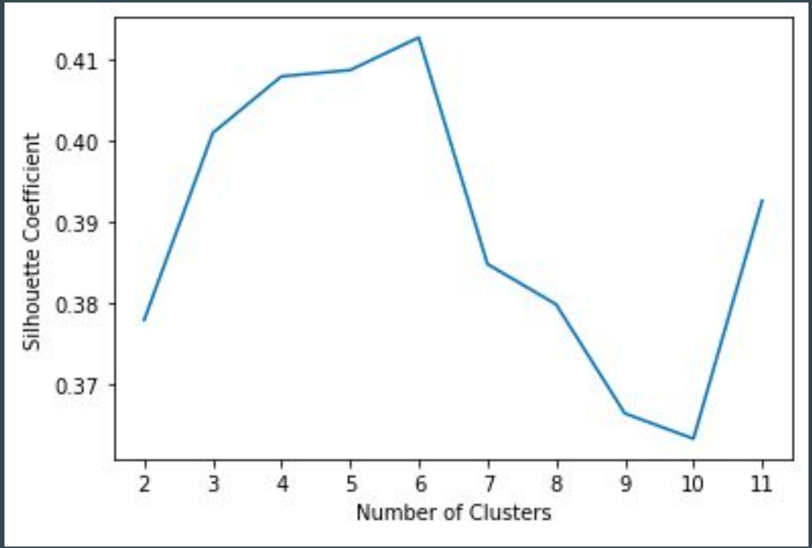
Data Formatting: Distribution of Tag Count

Range: 0 to 73



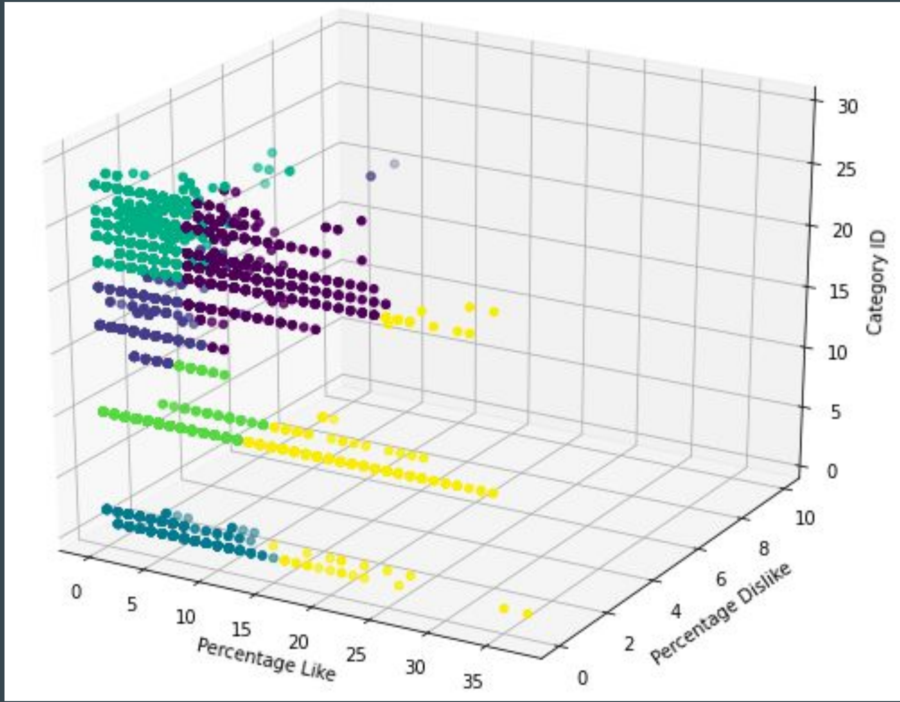
KMeans: Choosing Optimal k

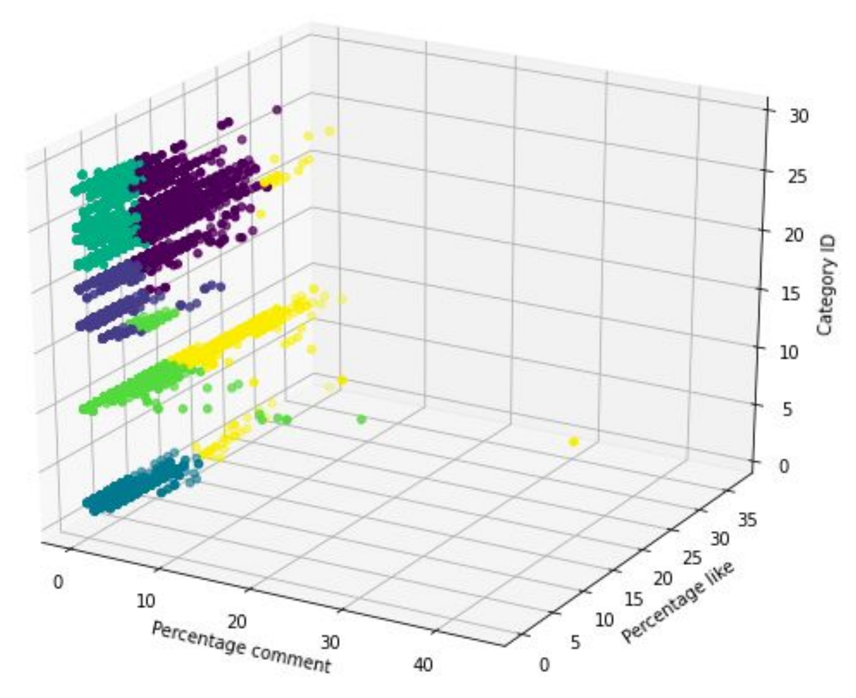




Optimal value for k=6.

KMeans: Clustered Data



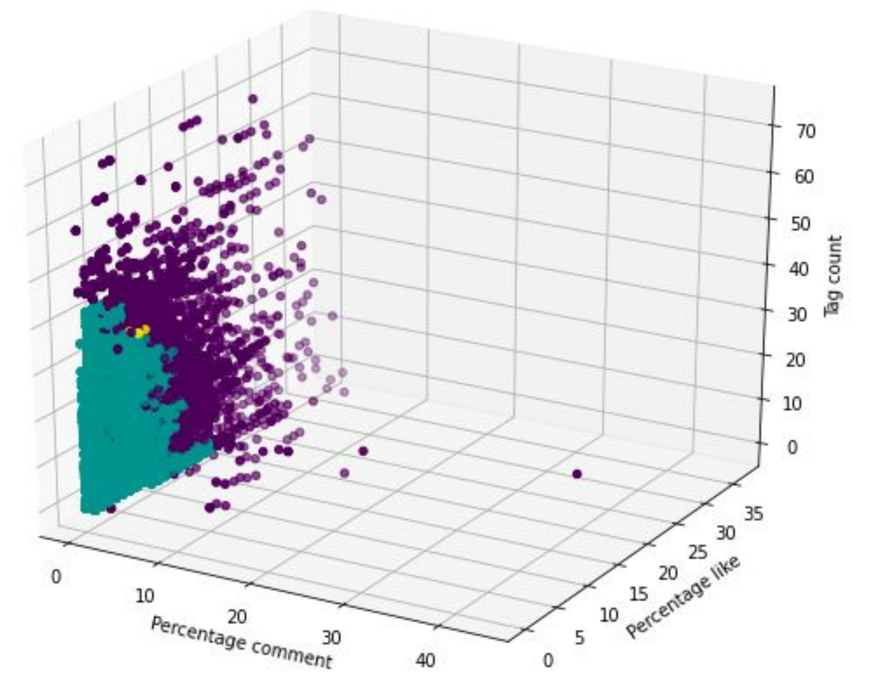


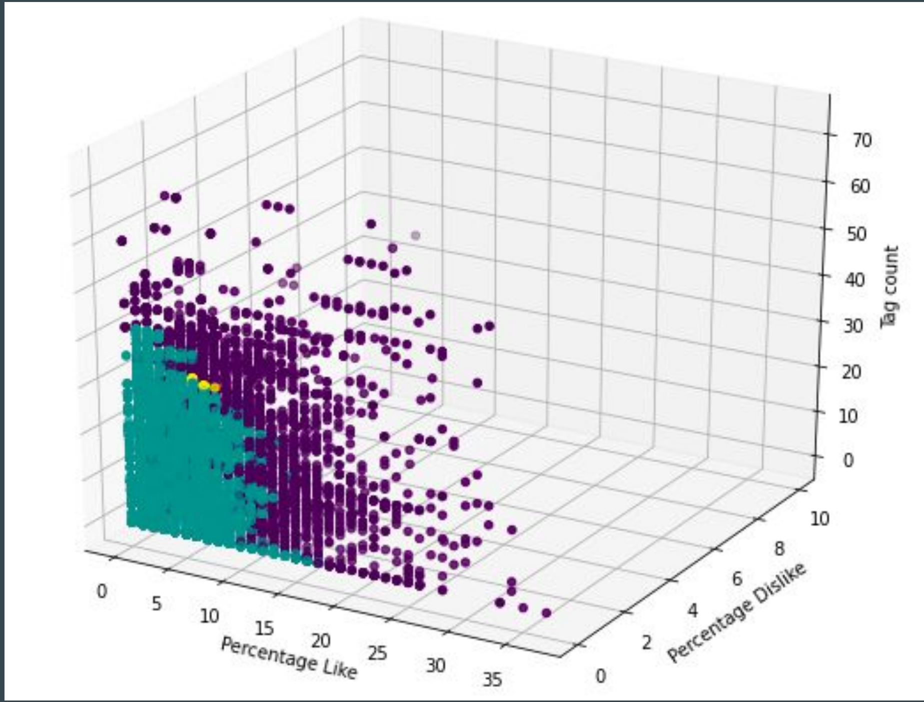
KMeans: Clustered Data

Color	Category ID	Average Percentage Like	Average Percentage Dislike	Average Percentage Comment
Violet	17 to 29	11%	0.02%	0.97%
Dark Blue	15 to 20	3.02%	0.04%	0.3%
Light Green	10, 15	6.27%	0.02%	0.48%
Dark Green	22 to 29	4.0%	0.06%	0.36%
Yellow	1 to 24	18.38%	0.06%	1.56%
Light Blue	1, 2	5.83%	0.01%	0.48%

Light Blue Cluster with Film and Animation, Car and Vehicles categories has least dislike.
Light Green Cluster with Music, Pet and Animals has about average 6% like.

DB Scan: Clustered Data





DB Scan: Clustered Data

Color	Tag Count	Average Percentage Like	Average Percentage Dislike	Average Percentage Comment
Green	0 to 42	4.95%	0.02%	0.37%
Yellow	33 to 34	6.77%	0.0%	0.85%
Violet	0 to 73	11.54%	0.16%	1.28%

Violet Cluster with 0 to 73 tag count has highest average like, dislike and comment percentage.
Yellow Cluster with 33 to 34 tag count has NO dislike.
Green Cluster with 0 to 42 biggest of three groups has least average comment percentage.

Regression

Regression

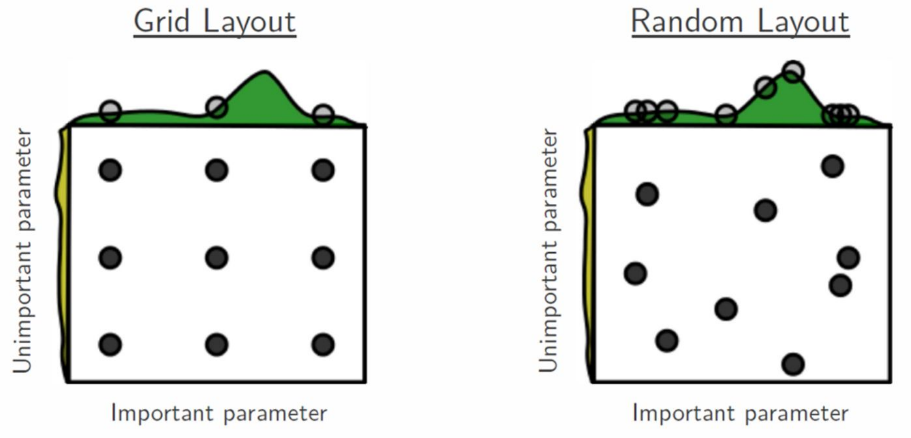
01	**Clean Data**	• Drop inconsequential columns • Convert publish date and trending data to datetime type • Sort dataset by publish date
02	**Preprocessing**	• Encode category ID’s and channel ID’s • Split training and test datasets
03	**Optimize Hyperparameters**	• Define grid of hyperparameters • Use Random Grid method for optimization • Output optimized parameters
04	**Evaluate Models**	• Run model with optimized hyperparameters • Visualize predictions

Hyperparameter Optimization

Hyperparameters: a parameter whose value is used to control the learning process

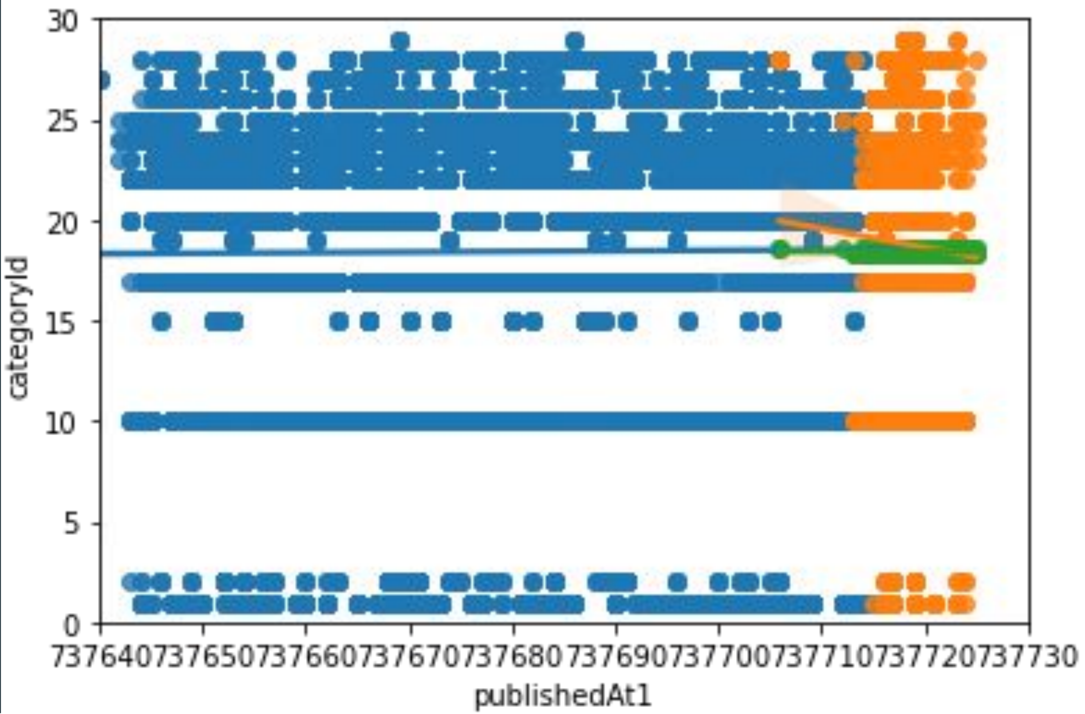
A few basic strategies: grid, random, Bayesian

Used RandomizedSearchCV from sklearn



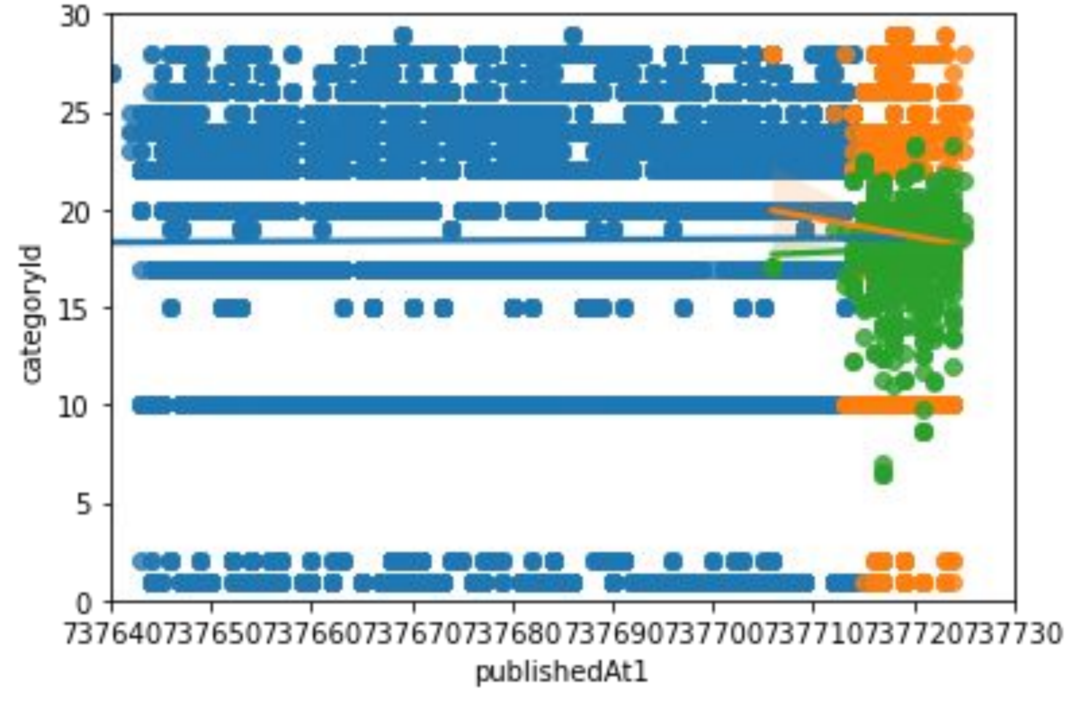
https://www.jeremyjordan.me/hyperparameter-tuning/

Random Forest Regression




```
RandomForestRegressor(max_depth=5, min_impurity_decrease=0.5555555555555556,
                      min_samples_leaf=2,
                      min_weight_fraction_leaf=0.3333333333333333,
                      n_estimators=5, random_state=42)
```

Gradient Boosting Regression




```
GradientBoostingRegressor(learning_rate=0.1111111111111111, loss='huber',
                          max_depth=5, max_features='auto', min_samples_leaf=2,
                          min_samples_split=10, n_estimators=51,
                          random_state=42)
```

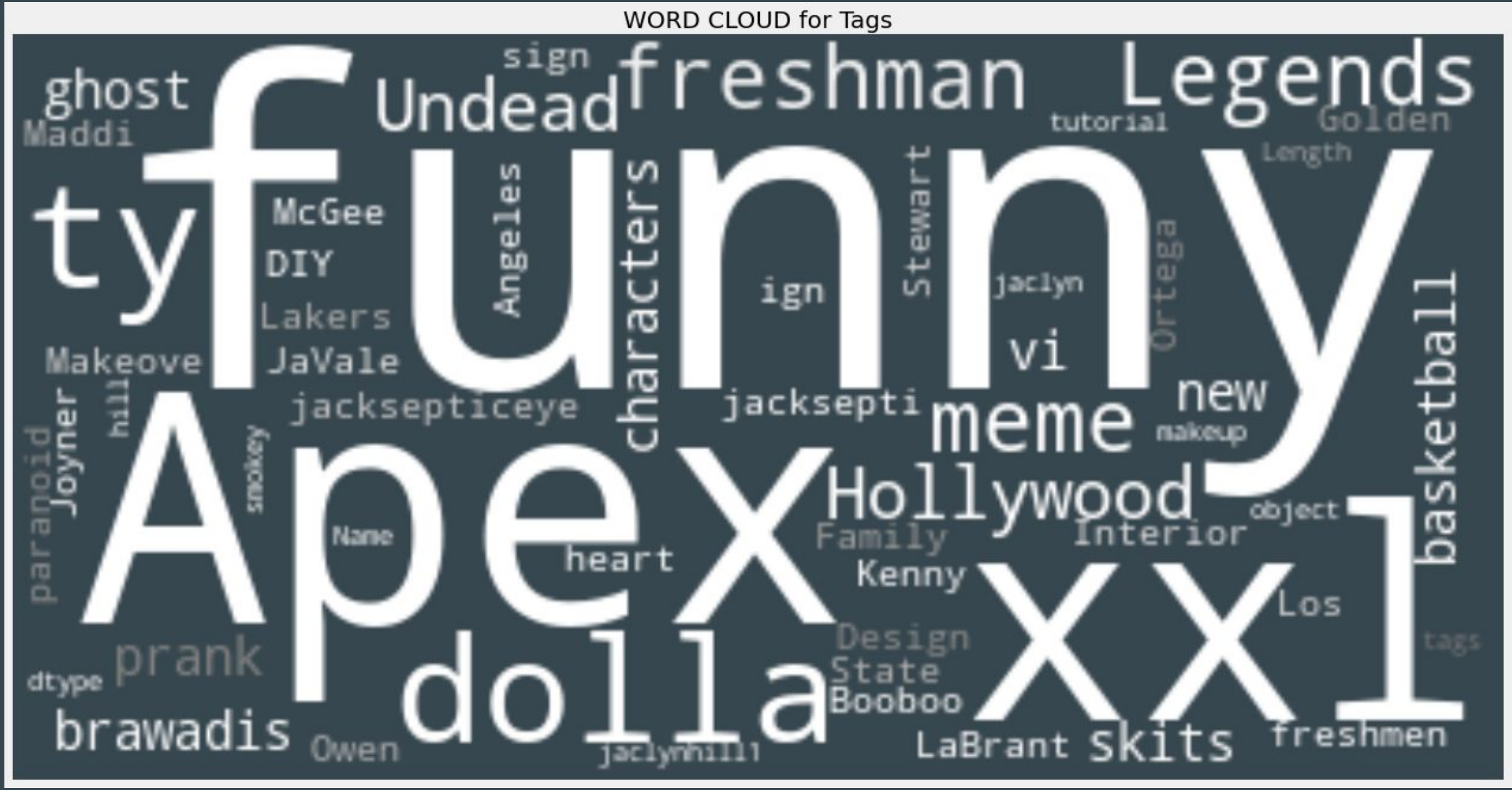
Time Series Forecasting

•••

Time Series Forecasting

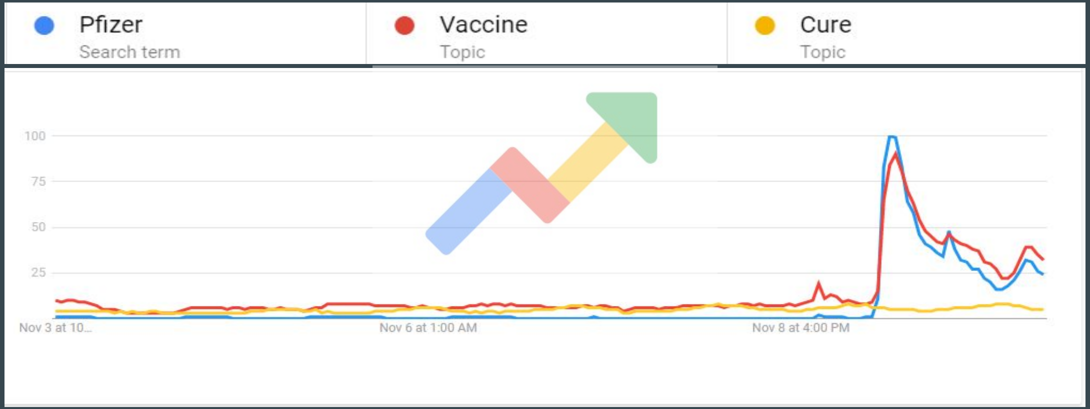
Format and Collect	• Format the YouTube data and collect tags for use with Google Trends
Gather Time Series Data	• Combine Google Trends data and YouTube tags to get a time series dataset
Select a Model	• Select a model that can handle the data provided
Predict Trends	• Use the model to predict trends in searches for tags and test the model using recent trends

Visualization of Popular Tags



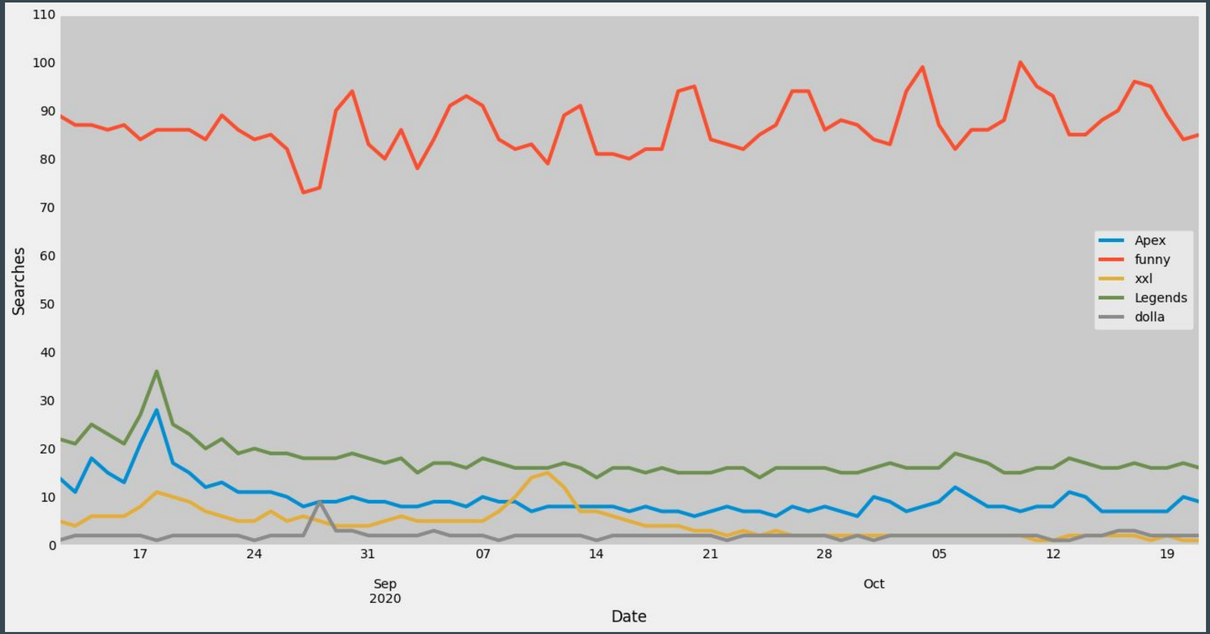
Google Trends API

- An unbiased sample of Google search trends
- Real Time Summaries
  - A random sample of the searches within a timeframe
- Non-Real Time Summaries
  - A random sample of searches from past years
- Normalized Trend Values
  - Magnitudes of Searches
  - Makes high search volume comparable to low search volume timeframes



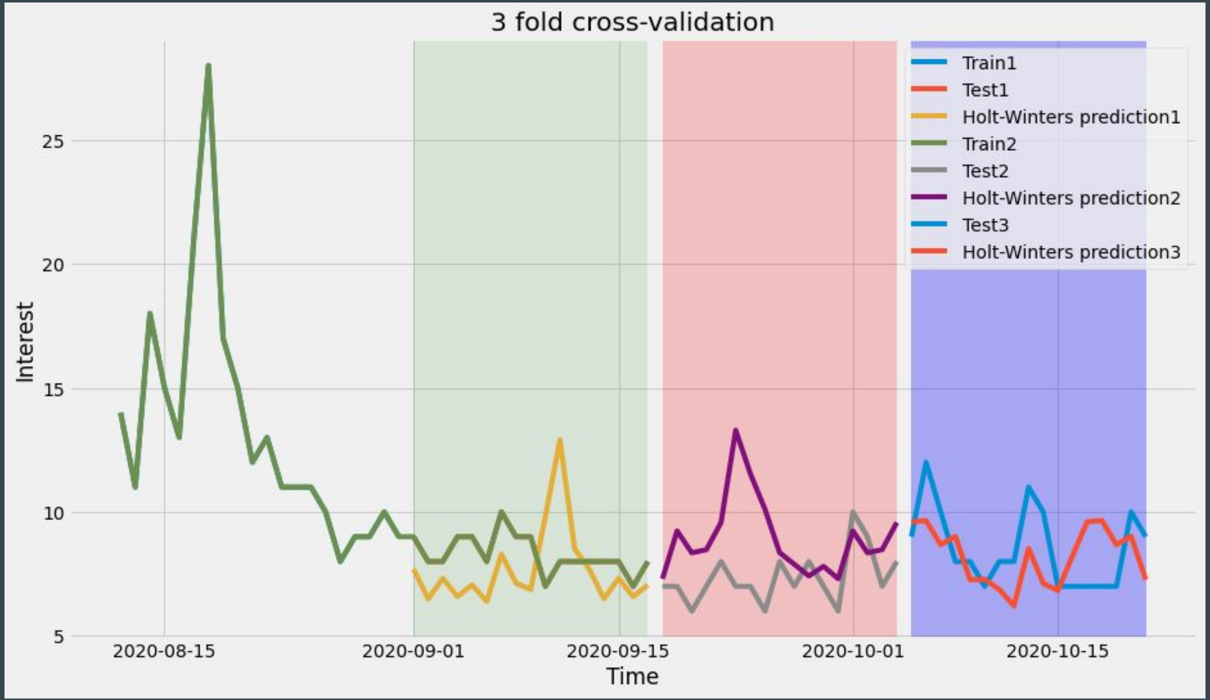
Google Trends Data for Popular YouTube Tags

- General tags stay popular
  - Funny is currently at 90% of its previous peak popularity
- Useful in determining related tags
  - Apex and Legends share similar trends in popularity
- Normalized data allows us to compare different tags simultaneously



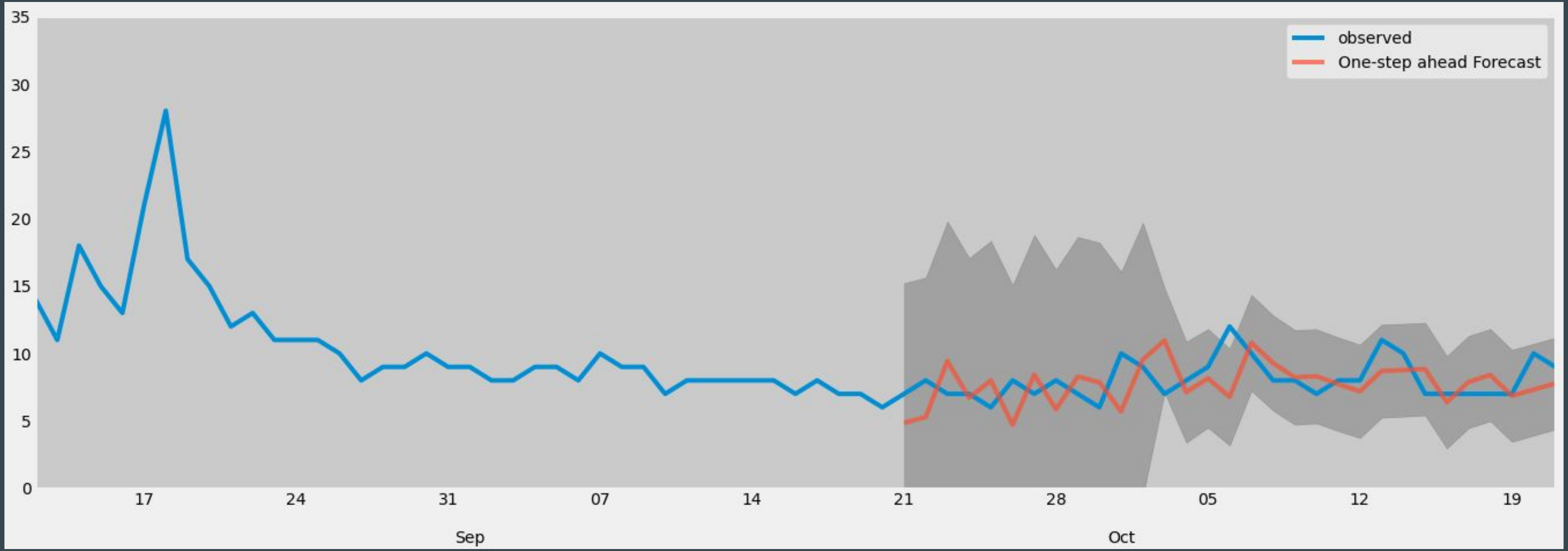
Holt Winters Seasonal Method with Cross Validation

- Holt Winters
  - Time series decomposition shows seasonality
  - Exponential smoothing method
- Seasonality
  - Daily data
- Nested Cross Validation
  - Divided the dataset into three training and testing pairs
  - Small dataset made it difficult to have high accuracy at the third fold

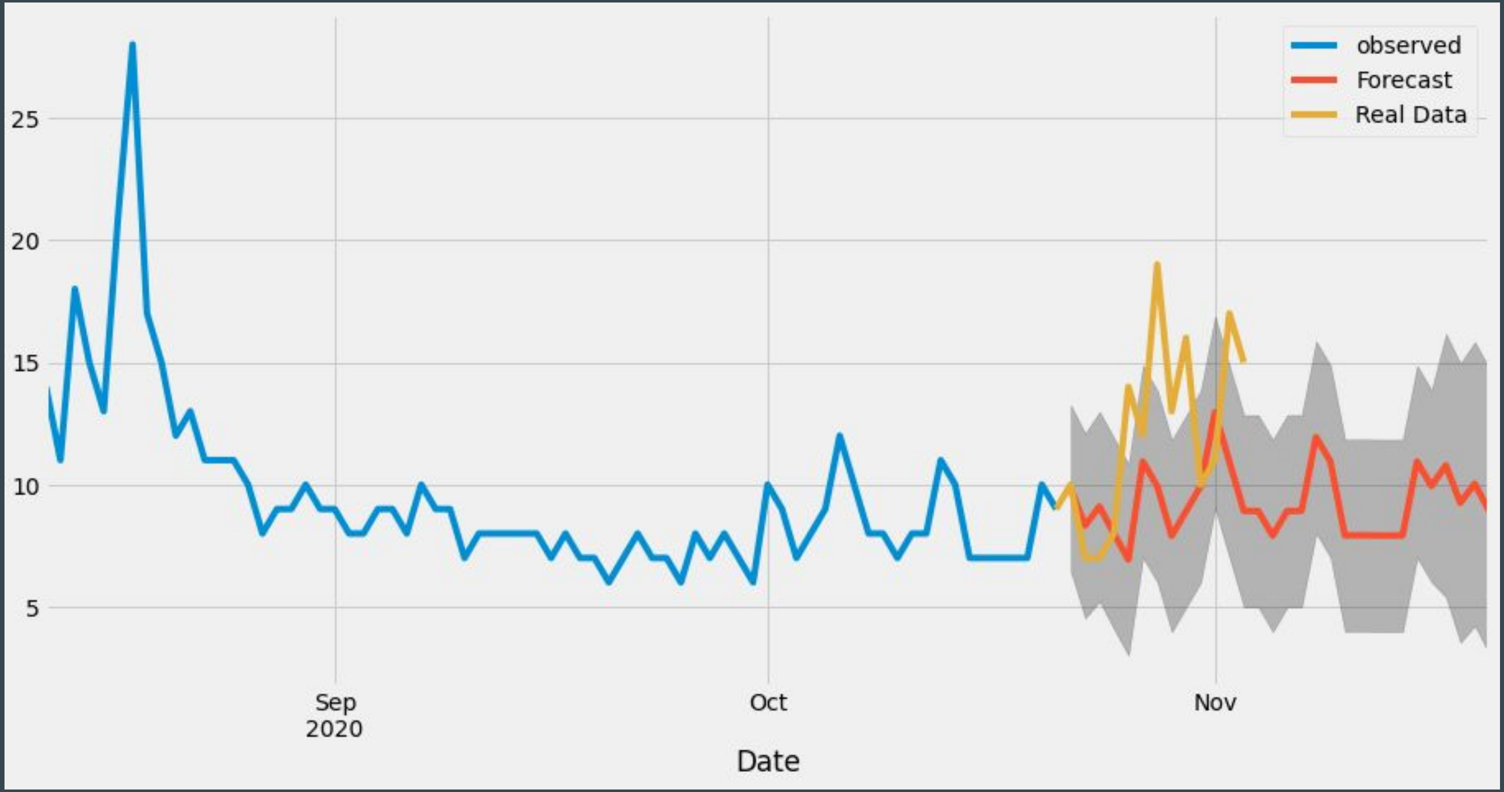


SARMA Model

- Optimize hyper-parameters
  - Coefficient for seasonal and non-seasonal terms
  - Determine seasonality
  - Minimize AIC
- Predict the behavior of the time series data
  - Use One-step ahead forecast to evaluate performance of the model
  - Use optimize hyperparameters to forecast the next month of views



SARMA Forecast for November On APEX



Anomaly Detection

•••

01 **Cleaning**

- Drop non-helpful features
- Split dates into machine readable text
- Split tags into lists

02 **Feature Engineering**

- Derive useful engagement metrics
- Derive useful temporal metrics
- Attempt to find items differentially correlated to views

03 **Model Selection**

- Evaluate feasibility of the various approaches
- Establish viability of chosen approach
- Select algorithm

04 **Training & Testing**

- Drop non-helpful features
- Split dates into machine readable text
- Split tags into lists

Subconcern

How do you define an anomaly when all you have is trending data - which by definition is anomalous?

Possible solutions:

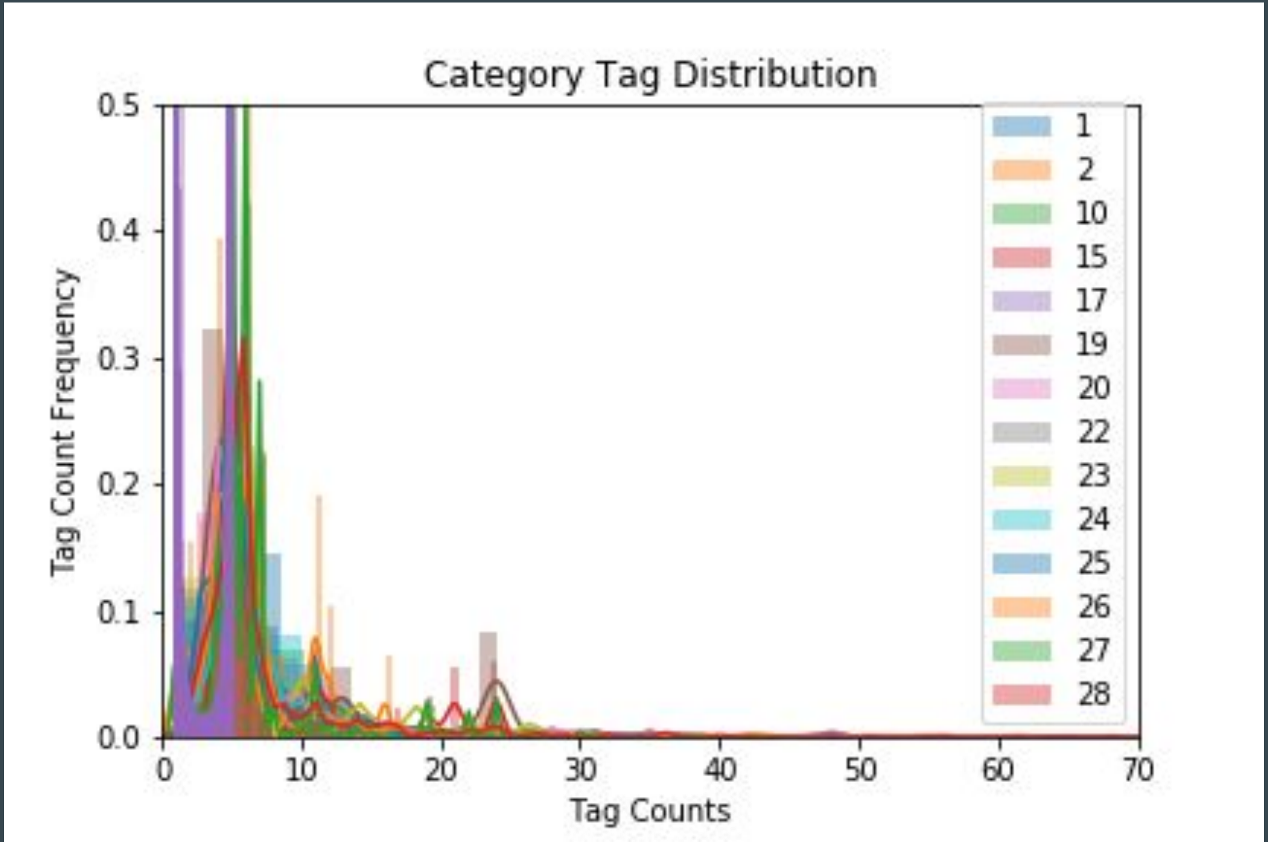
1)Find thresholds to anomaly status - ie anything 2 stdevs under mean isn't trending

2)Find an external metric as a reference - ie PyTrends interest_over_time counts

3) Use each category as an internal standard - Define anomaly by outside status

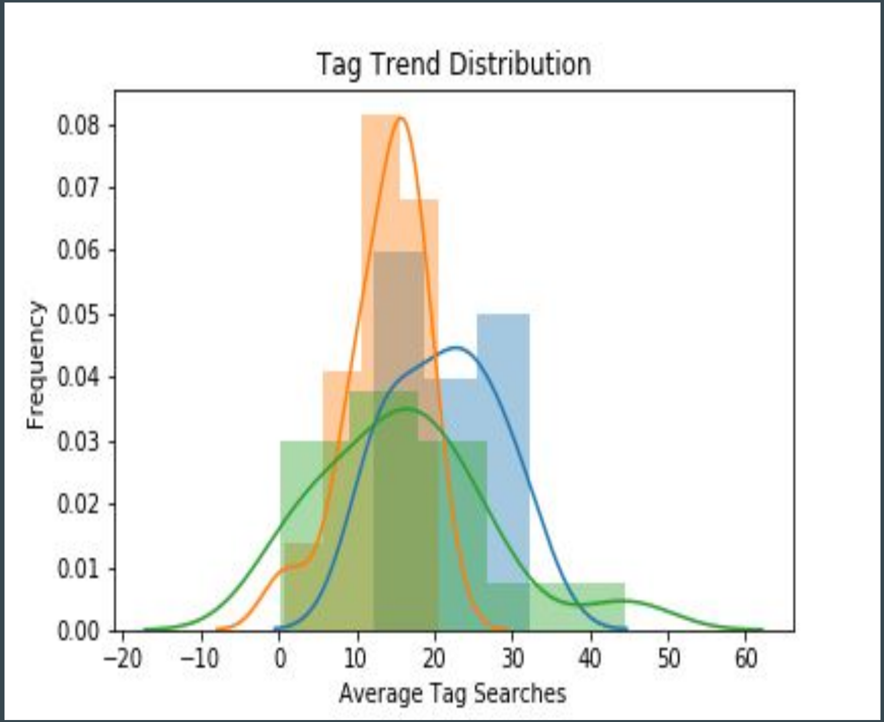
Tag Approach

- Discretize Tags by Category ID
- Separate frequently used tags from infrequently used tags
- Plug these in PyTrend to generate a normalized interest level over time per category
- Detect outliers relative to the baseline interest for that category
- Predict trends via spikes in relevant tags per view



Tags: Redux - Tag Frequency By Category

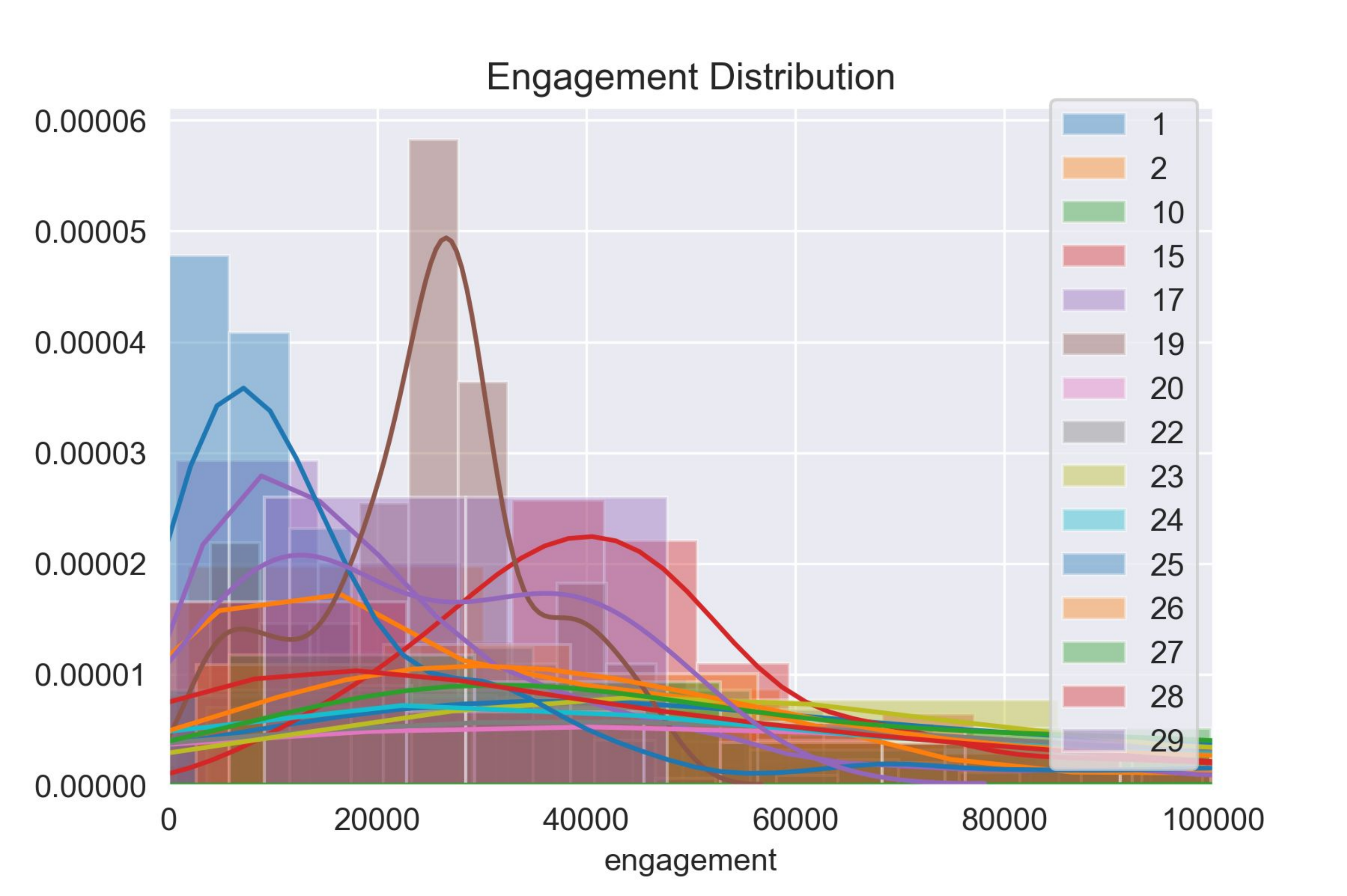
Tags: Redux - Tag Interest By Category

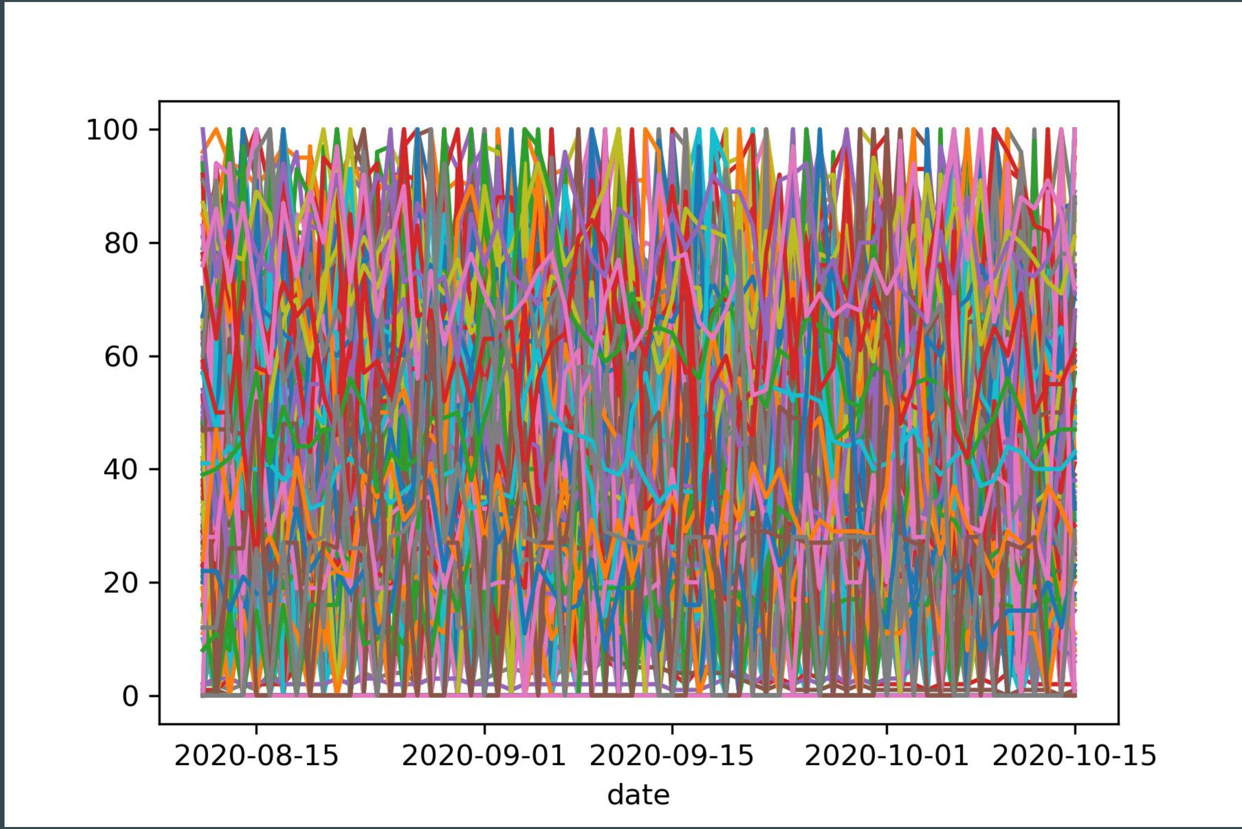


	Category	High Count Tags	Mid Count Tags	Low Count Tags	High StDev	Mid StDev	Low StDev
0	1.0	31.244949	4.121212	11.459596	29.416300	5.903679	11.243240
1	2.0	28.989899	25.172727	15.353535	31.705346	33.028106	24.234733
2	10.0	21.679293	14.293939	17.424242	34.521319	30.027783	33.722826
3	15.0	17.777778	16.730303	17.224747	34.124631	30.882338	31.070474
4	17.0	26.068182	0.303030	10.462121	21.706219	0.677596	16.405734
5	19.0	12.085859	16.278788	20.469697	27.648525	35.341047	27.662088
6	20.0	20.828283	18.060606	9.931818	29.085373	28.524322	14.601309
7	22.0	25.686869	14.115152	0.825758	37.576085	28.098252	1.942180
8	23.0	32.164141	21.081818	15.315657	35.323227	35.190580	25.160105
9	24.0	13.325758	21.927273	17.833333	19.166090	32.796233	24.956686
10	25.0	12.489899	2.478788	17.469697	19.504452	4.604448	27.476147
11	26.0	17.234848	9.175758	14.924242	34.570696	14.379277	35.742625
12	27.0	22.952020	27.912121	7.797980	30.744180	35.565689	18.671314
13	28.0	24.050505	8.284848	13.025253	33.023030	14.412601	31.712439
14	29.0	13.373737	44.636364	15.527778	26.777093	41.166785	31.795521

```
ResponseError: The request failed: Google returned a response with code 429.
```

Sublesson: Google does not take kindly to DOSing, accidently or not


Engagement Distribution
0.00006
0.00005
0.00004
0.00003
0.00002
0.00001
0.00000
0
20000
40000
60000
80000
100000
engagement
1
2
10
15
17
19
20
22
23
24
25
26
27
28
29



Subsampled Channel Trends

The Team

Matt Stalcup	Madhumithra SK	Emily Ruth Mikeska	Adam Podgorny
Time Series Forecasting	Clustering	Regression	Anomaly Detection