Introduction

In this project, I hoped to compile a list of keywords and hashtags that one could use on Twitter to garner more engagement for tweets that are on the topic of HIV and AIDS. A lot of organizations that work with HIV+ populations are either non-profits or charity/volunteer-based orgs and cannot afford to hire social media managers, so I hoped that I could provide an algorithm or model to help optimize tweets. Exposure often directly equates to cash influx for places like these.

However, after over 50,000 tweets pulled across many sessions, several rounds of data cleaning, feature engineering, and 5 different models, I was unable to find an underlying pattern in the data that a machine learning model could grasp onto.

Data collection

Steps:

- Create a pre-query based on generic terms related to the topic.
- Collect tweets using this pre-query to determine which terms should be in the main query.
- Collect main dataset using main query, save tweets in jsons.

Problem: building the query will determine the nature of the dataset. It's important to have a broad range of tweets, but that they are still reliably on the topic of HIV/AIDs.
Solution: perform an initial API search using fairly generic terms in the query. Pull the tweets of the user IDs that appear most commonly in this search, and collect the terms and hashtags they use most. Build the test query out of these terms.

Problem: tweets pulled from the same timeframe (the API pulls chronologically) will contain many repeats of the same tweet in the form of retweets.
Solution: build a loop in the collection code to change the start date every X number of pulls.

Problem: there are limits in the amount of tweets you can pull using a free Twitter API account.
Solution: I created a second account, and also waited out the month time limit so that I could pull more tweets.

Data Cleaning

Steps:

- Find retweets, identify original tweet ID, and remove all duplicates.
- Remove non-English tweets (a few snuck in despite a language filter)
- Remove stopwords and punctuation from tweets, separate out hashtags and mentions

Problem: there are many duplicates in the form of retweets. A retweet gets its own unique tweet ID, even though the text of the tweet is the same. This makes it harder to know which is the original tweet.
Solution: Digging through the documentation of both Tweepy and the Twitter API showed that the tweet json for retweets contains the information of the original tweet.

Data Exploration

Steps:

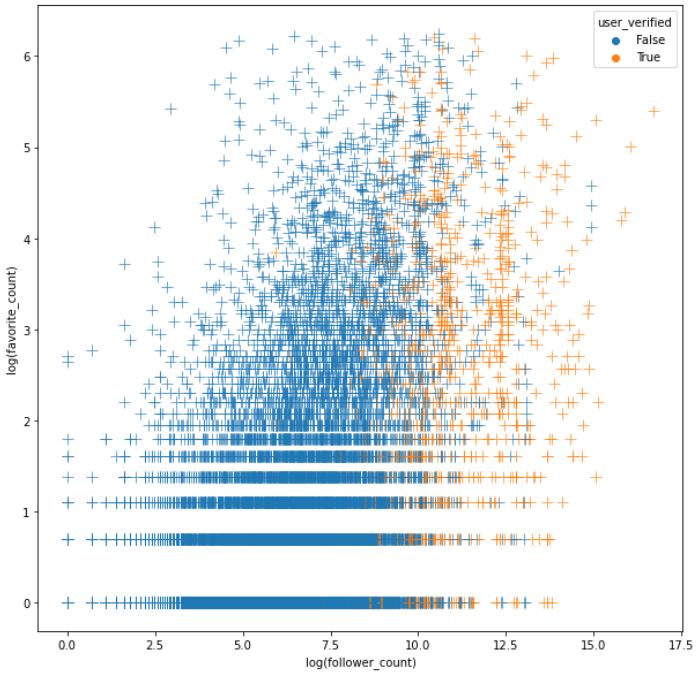
- Remove outliers.
- Look at general correlations and distributions.
- Examine relationship between follower count and tweet engagement.

Problem: Viral tweets -- virality is almost always a function of luck rather than the content of the tweet.
Solution: Remove outlier tweets, defined as any tweets at or above the 99th percentile in interactions.

Problem: There are four different engagement metrics: favorites, replies, retweets, quote retweets.
Solution: They are all highly correlated with each other, so I consolidated them into one metric.

Problem: Tweets by users who are verified and/or have a high follower count might get higher engagement not because of the content of their tweets, but just by sheer exposure to higher numbers.
Solution: Statistical testing showed that while follower count does not actually correlate to engagement, a user's verified status significantly affects the engagement rates.

Figure: Favorite count is generally agnostic to follower count



The scatter plot of log(favorites) vs log(follower count) shows a slight positive correlation (r squared = 0.19) between the two. Generally speaking, verified accounts tend to have a higher follower count. However, favorite count for a given tweet seems to be mostly agnostic to follower count.

Data Preprocessing

Steps:

- Generate categories for the target variable.
- Some final cleaning steps.
- Train/test split of the data.
- Transform data using Tfidf vectorizer.
- Reduce dimensionality of data using NMF.

Problem: Target variable (engagement) is in discrete ranges, and has a lot of zeroes. It's not good for regression models.
Solution: create categories so that categorical models can be used.

Problem: Creating text features using Tfidf results in far too many features. PCA is not good when you need to know what features are the ones that are most important.
Solution: NMF also reduces dimensionality, but allows you to recreate the features for better interpretability.

Modeling

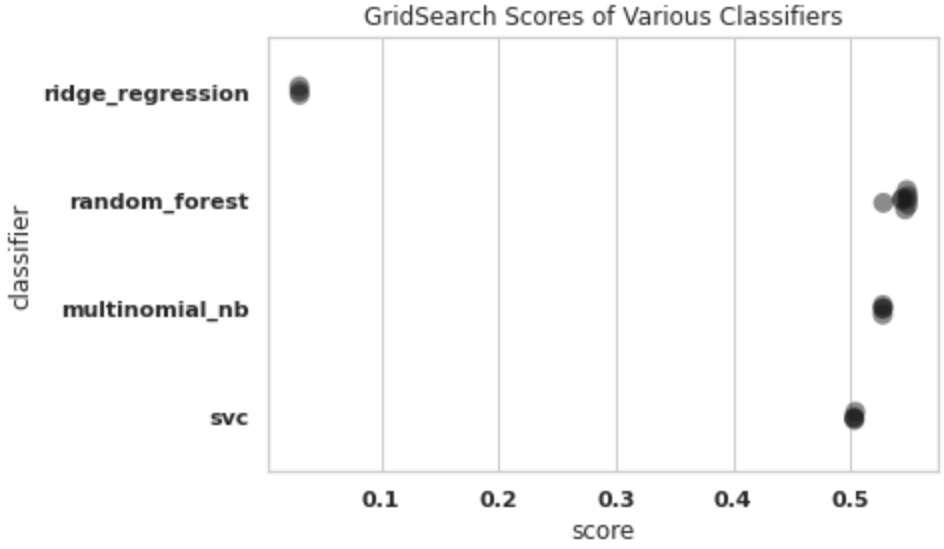
Models tested:

- SVC
- Multinomial NB
- Random Forest Classification
- Kmeans clustering
- Gaussian Mixture clustering
- Ridge Regression

Problem: All of the models had extremely poor performance out of the box.
Solution: Used GridSearchCV to attempt to optimize hyperparameters.

Problem: Even with GridSearch and hyperparameter tuning, I was unable to get a model to perform above 55% accuracy.
Solution: None, really.

Figure: Accuracy scores of a subset of the models I tested and attempted to optimize.



Future Directions

I think this project requires something more powerful than classification algorithms. If I were to do this again, I would approach it more like building a chat bot. Using deep learning to figure out what components make a “believable” tweet on the topic would, by definition, extract the most salient parts of the conversations.