Exploring Topics in the #NBA Twittersphere

BrainStation Data Science Diploma Program Capstone Project - Final Report
by Kevin Sia

Problem Statement

What do NBA fans talk about on Twitter?

Background

Unsupervised machine learning (ML) can facilitate the discovery of opinions or topics in a collection of Tweets (or in any corpus in general). This could have further applications in *supervised* ML projects, since the clusters found through unsupervised ML could be used as the data point labels that are needed for supervised ML. This is useful because it is impractical to manually label every single document in a large corpus. An example of a supervised ML project would be sentiment analysis. There are some published research papers that explore topic discovery, opinion mining and sentiment analysis on Twitter through unsupervised ML.

Data

The corpus used for this project was a collection of Tweets that were acquired through Twitter's standard search API from March 10 to March 17, 2020. I retrieved "recent" Tweets (as opposed to "popular" or "mixed") that contained "#NBA". Note that the NBA announced the suspension of their season on the evening of March 11, soon after it was revealed that a player tested positive for coronavirus.

I collected 46,914 Tweets in total. Each row in this dataset contains information from a single Tweet or Retweet. Of 9 Tweet features from the Twitter API I deemed as being potentially useful, only the following were actually used for any analysis:

- created_at: timestamp of when Tweet was posted
- handle: the handle (username) of the user
- retweet: a binary column where the value is 1 if the Tweet is a Retweet, and 0 if not
- tweet: the text of the Tweet

Data Cleaning, EDA & Preprocessing

The only data cleaning done was converting the columns into the necessary data types, e.g. converting the created_at column to a pandas datetime format. The most important EDA discoveries were 1. most of the accounts that appeared in the dataset are bot-like and frequently post Tweets pertaining to sports betting picks/advice/odds, and 2. that there was, expectedly, a spike in the number of Tweets posted after the NBA announced the suspension of their season on March 11 due to a Utah Jazz player testing positive for the coronavirus.

NLP techniques and scaling were used to preprocess the data. In tokenizing the corpus, I opted to use lemmatization because, with interpretability of the results in mind, lemma forms are easier to understand than stemmed forms. I then used a count vectorizer (instead of a weighted method like a TF-IDF vectorizer) because I was interested in finding only the *popular* topics - words with high term frequencies are what we want to analyze in the end. Extra stop words were added to the vectorizer after analyzing some of the tokens in the first vectorization. Lastly, scaling the data was necessary because I planned to use distance-based ML models. I opted to use the max-abs scaler from the scikit-learn library as it retains data sparsity; vectorization of our corpus resulted in a very sparse matrix.

Modelling

Before building classical unsupervised ML clustering models, I first used latent variable ML for both data exploration and modelling purposes - I employed PCA, t-SNE, and LDA topic modelling. In PCA, the words (features) that comprised each principal component made much interpretable sense. Then, a plot of the 2-D t-SNE-transformed data in showed natural clustering patterns.

LDA topic modelling was the first method used to explore topics/clusters in the dataset. I started with setting the number of LDA components to 40, which is the guess of how many distinct clusters are apparent in the 2-D t-SNE representation of the data. From there, the goal was to iteratively decrease the number of components to see if/how the model would combine some of the 40 clusters. The pyLDAvis library (for generating LDA model visualizations) was used to evaluate topic overlap in two ways: by seeing which topic bubbles overlapped in the intertopic distance map, and by checking the words in different topics to see if they were similar or different. I ended at four iterations and 16 LDA components, where the model's topics were distinct and interpretable. No numerical metric was used to evaluate the models.

The next model I chose to explore was k-means clustering. I first wanted to see how the inertia and the silhouette score varied when increasing k. The plot of inertia vs. k indicated that clusters in the dataset were not tightly-grouped. The silhouette score stayed close to 0 for all values of k, meaning that the models were unable to find clusters that did not overlap. I then opted to start exploring k-means clustering by starting with k = 10, which yielded the best possible silhouette score (though not by much). This first model surprisingly did not pick up on any sports betting cluster, so I incrementally increased k to see how many clusters it took to find a sports betting cluster - that k value was 16. Ultimately, I was not confident in using k-means clustering because of the silhouette score.

The last type of model used was DBSCAN. I kept the min_samples parameter constant at 10 (arbitrary choice) to allow the eps parameter to vary. The first DBSCAN model had eps = 2, which yielded 20 clusters. The goal from here was to change eps in order to obtain less clusters. I ended up iterating twice more, with eps = 2.4 and eps = 2.6. The final model (eps = 2.6) yielded only 9 clusters. In each iteration, there was always one very large cluster, a somewhat large noise "cluster", and all of the other clusters were significantly smaller in comparison. These small clusters ended up being Tweets that came from single accounts, i.e. one cluster was assigned to one account's Tweets. These accounts were usually bot-like,

where the same words or hashtags were used in every one of their Tweets. DBSCAN was thus not entirely useful in discovering topics.

Results

Common topics found across all models included coronavirus (players testing positive, NBA announcing suspension, discussions in other major sports leagues), sports betting (picks, advice, odds), and popular franchises/players (Los Angeles Lakers, LeBron James). The k-means clustering and DBSCAN models possibly interpreted sports betting Tweets as being noise, while LDA topic modelling seemed to define them as a topic(s). Coronavirus aside, I initially hoped to discover what *NBA fans*, not bots, talk about on Twitter, and I was more specifically hoping to find opinions from NBA fans. Thus, bot accounts and sports betting Tweets were unexpected discoveries.

Business Applications

If the dataset is collected when the NBA season is in full effect, it would be much more possible to discover opinions of NBA fans. I would foresee that knowing this information is useful for marketing, and possibly public relations. For marketing, an example would be increasing advertisement activity for tickets to a certain team's basketball games if NBA fans on Twitter are raving about that team, or promoting merchandise of a specific player (e.g. jerseys) if fans are praising that player. In public relations, if for example fans on Twitter are collectively upset about referee decisions, the NBA can recognize this quickly and act upon it.

Next Steps

"Poor performance" in any of the models seems to be more attributed to the dataset and not the models themselves, i.e. the presence of bot Tweets forced the models to pick up on those as their own clusters. It would thus be worthwhile to look into methods of removing Tweets from bots from the dataset, and also filtering out certain topics.

I would like to redo this project with a dataset that is collected during an active NBA season. While it's interesting to explore the topics before and after the season was suspended, my main goal was to analyze Tweets when the NBA season was ongoing.

I would also explore the different parameters in the Twiter standard search API. It's possible that different search terms as well as "mixed" or "popular" Tweets could drastically change the topics in the dataset.

Feature engineering could facilitate computation and analysis. A simple example is adding the word counts of two related features, like a player's name and his nickname.

Lastly, I am curious to know how other unsupervised models like hierarchical clustering and expectation-maximization would perform.