Christine Lee

Professor Li

MIS 3640-01

November 4, 2017

Overview:

Given the recent victory of the Toronto Raptors, I decided to analyze the tweets containing the Raptors to get a sense of the audience’s mood towards the team. I predicted that the response would’ve been positive, since they have are ranked the best in their regional conference. The Raptors have only lost two games, which means they are most likely to advance to the playoffs.

Sentiment analysis is very important because it allows the analyst to identify and categorize opinions expressed through text. If a business wanted to see how their brand was performing in the market, through sentiment analysis they can see get insight into what their customers want and how they feel about their products or services.

After I used “#WeTheNorth” to find 200 tweets, with the installation of Tweepy and TextBlog I was able to mine the text into three categories: positive tweets, neutral tweets, and negative tweets.

Implementation:

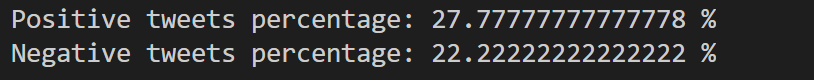
In order to run my code, I first had to run a few packages. The first package I ran was Tweepy, which is the python client for the official Twitter API. This allowed the software to gather all the tweets from the website. Then, I installed the TextBlob, which dives into common natural language processing tasks such as sentiment analysis and classification.

After installing my packages, I was then able to create a class for the twitter client and use that to make different functions. I then used the \_\_init\_\_ function to handle the authentication of the API client. I also used the clean tweet function to clean tweet by removing links and special characters. I also used the TextBlob to pass the tokens to a sentiment classifier. If the polarity of the tweet was greater than zero, it was labeled as ‘positive’, if the polarity of the tweet was equivalent to zero, it was labeled as ‘neutral’, anything else was returned as ‘negative’.

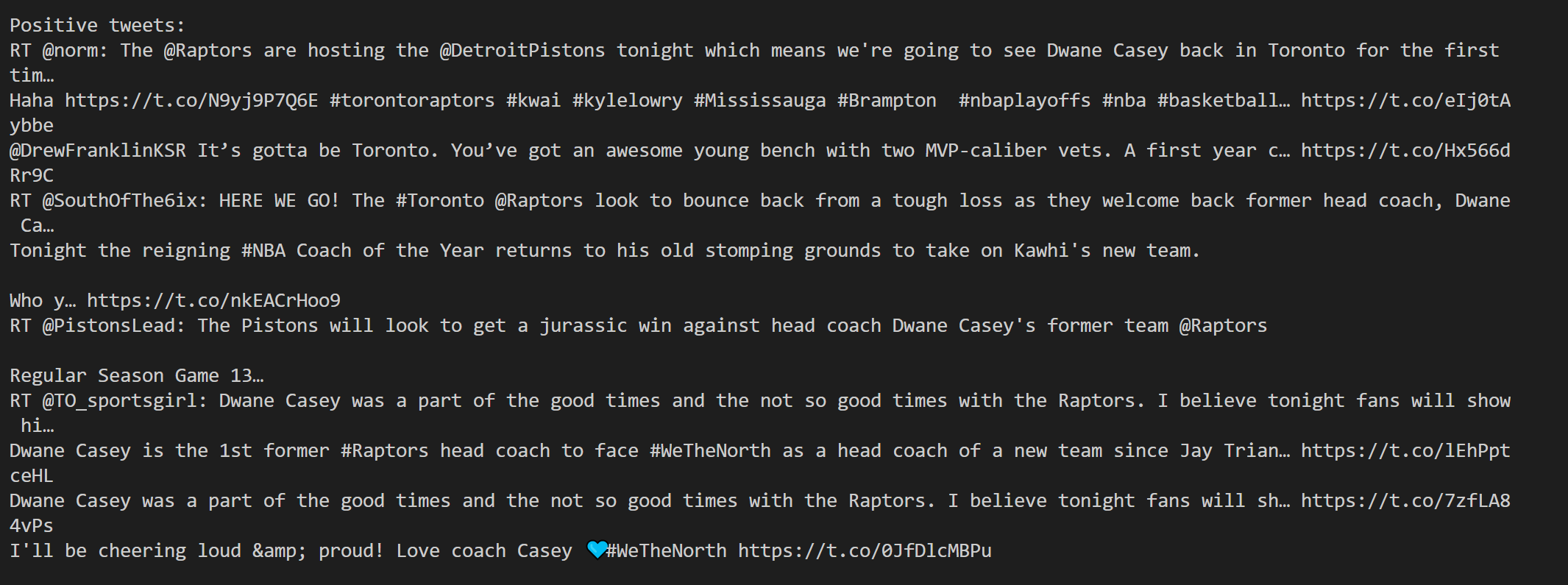
After the tweets were being retrieved put into their categories from the use of sentiment.polarity, I had to present the tweets in an effective. I ended up calculating the percentages of positive and negative tweets and then displayed the first five of each category. Another effective method would have been presenting the most popular, based on engagement or awareness. Popular tweets are a good representation to how most popular view the brand or the account. However, I wanted to stick with using more generic metrics, where we can make a more impactful comparison between negative and positive tweets.

Results:

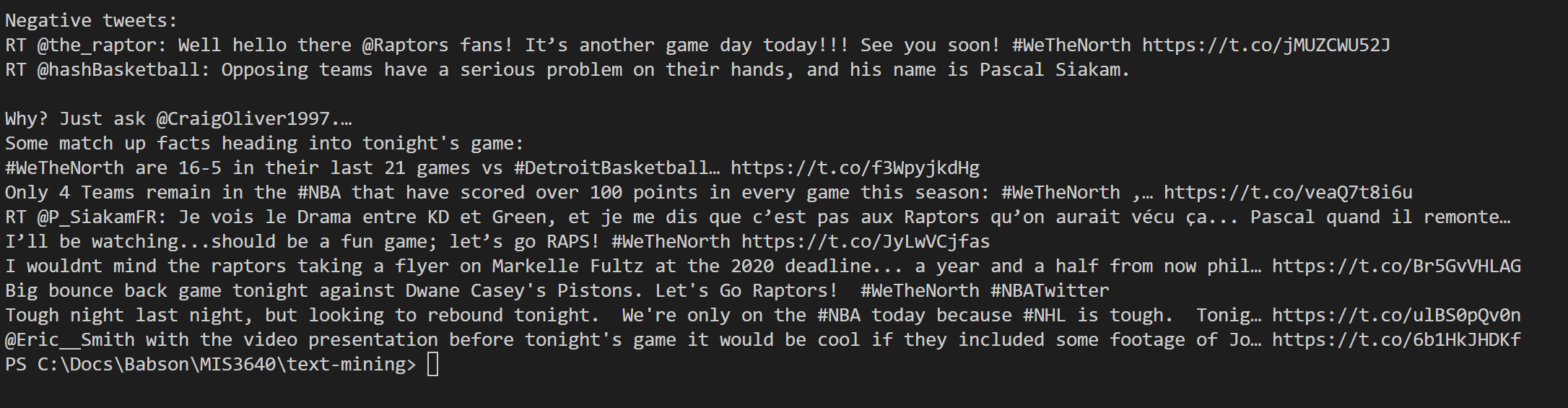
After running my code, I found a lot of interesting results. When I used “#WeTheNorth”, I noticed that the gap between positive and negative tweets weren’t as drastic, but they both had a very low percentage.



27.8% of the tweets analyzed were considered to be positive, while 22.2% were considered to be negative. This would leave a high percentage of neutral tweets being made. Of the 27.7% here are some of the positive tweet from my code:



Based on the result, it seems like the opinion mining was accurate with positive sentiment analysis. Most of the people are talking about Dwayne Casey, who is the head coach of the Raptors. In all of these posts everyone is complimenting the coach or are expressing excitement for the team’s performance and upcoming games.

 Below are the negative tweets:

The results to me are very surprising. I don’t see any negative tweets. I can see how some words like “tough” could be used in a negative context, but all of the tweets listed as negative are positive. This makes me question how accurate this program is when it comes to picking up negative tweets.

Reflection:

I feel like doing a sentiment analysis for negative tweets can be very difficult. I see a trend with some of the negative tweets of how some words can be perceived as negative. It is difficult for the natural language processor to catch onto negative words used in a positive context, since it lacks the ability to detect sarcasm or satire. I have learned that classifying social media posts by a program is not always practical at scale. There needs to be a more accurate sentiment, where a program can find the relationship and sentiment between words.