Using Twitter API to Solve the GOAT Debate: Michael Jordan vs. LeBron James

Jordan Trey Leonard

Follow this and additional works at: https://scholarship.claremont.edu/cmc_theses

Part of the Applied Mathematics Commons, Data Science Commons, and the Statistics and Probability Commons

Recommended Citation
https://scholarship.claremont.edu/cmc_theses/2733

This Open Access Senior Thesis is brought to you by Scholarship@Claremont. It has been accepted for inclusion in this collection by an authorized administrator. For more information, please contact scholarship@cuc.claremont.edu.
Using Twitter API to Solve the GOAT Debate: Michael Jordan vs. LeBron James

submitted to
Professor Mark Huber

by
Jordan Leonard

for
Senior Thesis in Mathematics
May 3, 2021


**Introduction**

**What is Sentiment Analysis?**

Sentiment analysis (Feldman 2013) is a unique data mining (Hand and Adams 2014) tool that refers to the use of natural language processing (Chowdhury 2003), text analysis (Bernard and Ryan 1998), computational linguistics (Grishman 1986), and biometrics (Jain, Flynn, and Ross 2007) to identify, extract, quantify, and study subjective information. It is commonly used to gather information on public opinion by breaking down text to determine whether it contains positive or negative sentiment. Many studies tend to gather their text data from social media platforms due to the large number of users and available content. In this case, I use sentiment analysis (Feldman 2013) to analyze tweet data collected from Twitter in RStudio (Allaire 2012).

**Problem Description**

In the following paper, I gather and analyze Twitter tweets from real users to compare the social sentiment of professional athletes in the National Football League (NFL), National Basketball Association (NBA), Major League Baseball (MLB), as well as athletes who play National Collegiate Athletic Association (NCAA) Division 1 basketball. The reasoning for my analysis of the social sentiment of athletes on Twitter began with my interest in solving the dispute of labeling athletes as the GOAT or the “Greatest of All Time.” Granted that every professional athlete is extremely talented and made it to the professional level for a reason, the label of GOAT is reserved to the best of the best. In the NBA, the discussion tends to come down to comparing LeBron James and Michael Jordan. With that being said, I decided that I would employ the technique of sentiment analysis (Feldman 2013) and a Twitter API (Makice 2009) in an attempt to find some sort of resolution. I also believed this analysis to be vital for the fact that Michael Jordan released a documentary of his NBA career called *The Last Dance* (“Everything You Need to Know about ‘The Last Dance’” 2020) in April of 2020, and LeBron James won an NBA Championship with the Los Angeles Lakers in October of 2020. With these two major events occurring in the same year, I hoped that it would produce enough substance to compare both athletes on a seemingly even scale.

**Data Gathering Process**

In order to publicly and freely access Twitter tweets, it is required to go through an application process in which one is granted confidential keys to access the Twitter API (Makice 2009) for your specific project. Once these keys are granted, you are able to use the keys as search tokens within the rtweet package in RStudio (Allaire 2012) to run the API that enables tweet collection. Using the search_tweets() function, I was able to input a given keyword that I would like to search for across the Twitter database and get an output of tweets that include that keyword. However, the the
Twitter API (Makice 2009) that I was granted is limited in that I am only able to search for 18,000 tweets in a given 15-minute period, and the tweets that are searched over for a given keyword had to have been tweeted in the past 6-9 days. Hence, the API granted me a limited time frame to work with which was unfortunate as I had hoped to access tweets ranging over the past couple of years. A larger time frame would allow me to see how the social sentiment revolving around athletes fluctuated due to their athletic performances and achievements within their respective sporting seasons.

Given that the tweet data collected only ranges over the past 6-9 days from when the API is employed using the `search_tweets()` function, I was still able to gather important results for the athletes in my study as the NBA, MLB, and NCAA basketball seasons were still ongoing. Despite the NFL season not being in progress like the other sports, there was still valuable tweet data to be collected and analyzed. In the data gathering code, it can be seen that the `search_tweets()` function is simple and readable in that it uses a search query argument, `q = [Name] GOAT OR [Name] Goat`, for every athlete. It also includes the arguments `n`, `include_rts = FALSE`, `-filter = "replies"`, and `lang = en`. These arguments specify the desired number of tweets to be returned while filtering out retweets and replies, and only collecting tweets that are in English. These arguments assist in keeping my data concise and focused on the portions that are necessary for further analysis.

Analysis

NBA

Exploring Tweets

Read in Data Using the `search_tweets()` function from the `rtweet` package in the following code chunk, we are able to collect tweet data regarding Michael Jordan. Given that the focus is on tweets that contain the term `GOAT`, the search query `q = "Michael Jordan GOAT OR Michael Jordan Goat"` is adopted to narrow the scope of the search. The values from the search are stored in the variable `mj_goat_tw` and contain 279 observations of 91 variables. This means there is a total of 279 tweets available that are sorted into 91 column variables such as “user_id,” “created_at,” “text,” etc. To ensure that the analysis is done on the same set of data instead of consistently recollecting new tweet data, the `write_as_csv()` function stores the values from the `mj_goat_tw` variable into a CSV file. This not only saves the data in a safe, readable format but grants the ability to read in the data after each session using the `read.csv()` function.

```r
### Michael Jordan
mj_goat_tw <- read.csv("mj_goat_tw.csv", fill = TRUE)
mj_goat_tw2 <- read.csv("mj_goat_tw.csv", fill = TRUE)
```
Now that the tweet data for Michael Jordan is collected, it is time to dive into the data by performing EDA. This enables the ability to fully understand the data that has been collected in order to perform further analyses later on. In the following code, we are using the `pipe` operator from the data variable so that we can see a sample of size 3 for the selected column variables of “created_at,” “screen_name,” “text,” “favorite_count,” “retweet_count.” From the output, it can be seen that the “text” column contains the terms “Michael Jordan” and GOAT. This is important because it ensures that the search_tweets() function from the data gathering process is working properly by producing valuable results.

```r
mj_goat_tw %>%
sample_n(3) %>%
select(created_at, screen_name, favorite_count)
```

<table>
<thead>
<tr>
<th>created_at</th>
<th>screen_name</th>
<th>favorite_count</th>
</tr>
</thead>
<tbody>
<tr>
<td>2021-03-30 02:16:03</td>
<td>not_andrew___</td>
<td>1</td>
</tr>
<tr>
<td>2021-03-31 03:31:48</td>
<td>ulforicks</td>
<td>1</td>
</tr>
<tr>
<td>2021-04-02 22:48:08</td>
<td>Jimmyrealdeal</td>
<td>0</td>
</tr>
</tbody>
</table>

This process can be replicated for the other NBA players within our sample to confirm that the tweets we analyze are in fact referencing each specified player.

Below, we can see the sample of tweets for LeBron James, James Harden, Kevin Durant, and Kobe Bryant which used the same coding process on their respective dataset of tweets.

Since the data outputs contain the “created_at” column variable which labels the date and time that each tweet was published, it would be interesting to take a look at the tweet frequency by users who are invested in the NBA “GOAT” conversation.

**Timeline of Tweets - Frequency Plot**

The `ts_plot()` function allows us to investigate the frequency of tweets as they were tweeted between the dates of “2021-03-26 06:25:39” and “2021-04-03 03:12:00.” In the code below, we are able to specify a desired time interval to model which is where the `hours` and `days` arguments come into effect. Both models display a spike in frequency of tweets on “2021-03-28” which total to 50+ tweets for that day.

```r
### Michael Jordan
ts_plot(mj_goat_tw, "hours") +
  labs(x = NULL, y = NULL,
       title = "Frequency of tweets with Michael Jordan GOAT Keyword",
       caption = "Data collected from Twitter's API via rtweet") +
  theme_minimal()
```
Similarly, we are able to investigate LeBron James’ tweet frequency for tweets between the dates of “2021-03-26 23:49:59” and “2021-04-04 02:17:50.” The following frequency plot has a spike in frequency of tweets on “2021-03-28” and “2021-03-31” which have total of about 70 and 55 tweets for those days, respectively.
Next, we can take a look at James Harden’s tweet frequency between the dates of “2021-03-26 23:49:59” and “2021-04-04 02:17:50.” The following frequency plot has a spike in frequency of tweets on “2021-03-31” which total to about 25 tweets for that day.

Then, we can take a look at Kevin Durant’s tweet frequency between the dates of “2021-03-27 21:57:47” and “2021-04-04 02:26:39.” The following frequency plot has a spike in frequency of tweets on “2021-03-28” and “2021-03-30” which have total of about 20 and 25 tweets for those days, respectively.
Finally, Kobe Bryant’s tweet frequency takes place between the dates of “2021-04-05 06:16:15” and “2021-04-13 03:00:21.” The following frequency plot has a spike in frequency of tweets on “2021-04-06,” “2021-04-09,” and “2021-04-12” which have total of about 6, 10, and 12 tweets for those days, respectively.

When comparing the time intervals of hours versus days, the days interval provides an interesting visual of the daily frequency but the hours argument provides a better insight since we are dealing with a time-frame of only 6-9 days. If we were to be dealing with a dataset of tweets which span the course of a month or more, then the days interval would be an effective model.

The above frequency plots are offer valuable insight into the NBA GOAT conversation as we are able to notice that these athletes are consistently being “talked” about throughout their dataset time-frames. Since each athlete had at least one spike in tweet frequency, it may be beneficial to understand the sentiment/sentiment
polarity during those periods. This would allow us to determine if those spikes were positive or negative, and how it compares to the other days in the dataset.

**Top Tweeting Location**

Another variable that could potentially play a factor in the public sentiment towards an athlete is the location of where a Twitter user lives. When it comes to sports, fans tend to develop competitive attitudes which may lead to a resentment towards opponents. Some fans follow the teams that reside in their hometown or state while others may not, either way, location is present.

In the following code chunk, we can filter the `mj_goat_tw` variable to remove any NA values from consideration in the location column variable, while including the count of tweets from each location in the output. The reason for excluding NA values is that they do not provide any useful information and removing them presents a more efficient model. From the output, we can see that there were 104 tweets from a blank location, 16 tweets from Chicago, IL, and 4 tweets from both the United States and Washington, D.C.. In this case, the 104 tweets from a blank location were not represented as NA values nor were they removed due to the fact that the users did not enable the location feature when they published the tweet. Therefore, the missing location value was replaced with a blank cell when the data was read into the CSV file using the `fill == TRUE` argument. To correct this, we can run the following function to replace those blank cells with NA values so that the filter argument properly selects the non-NA values. When comparing the two outputs it is obvious that the blank cells are removed.

```r
mj_goat_tw2[mj_goat_tw2 == ""] <- NA

mj_goat_tw %>%
  filter(!is.na(location)) %>%
  count(location, sort = TRUE) %>%
  top_n(5) %>%
  kable()
```

<table>
<thead>
<tr>
<th>location</th>
<th>n</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>104</td>
</tr>
<tr>
<td>Chicago, IL</td>
<td>16</td>
</tr>
<tr>
<td>United States</td>
<td>4</td>
</tr>
<tr>
<td>Washington, DC</td>
<td>4</td>
</tr>
<tr>
<td>Boston, MA</td>
<td>3</td>
</tr>
</tbody>
</table>

```r
mj_goat_tw2 %>%
  filter(!is.na(location)) %>%
  kable()
```
```r
count(location, sort = TRUE) %>%
top_n(5) %>%
kable()
```

<table>
<thead>
<tr>
<th>location</th>
<th>n</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chicago, IL</td>
<td>16</td>
</tr>
<tr>
<td>United States</td>
<td>4</td>
</tr>
<tr>
<td>Washington, DC</td>
<td>4</td>
</tr>
<tr>
<td>Boston, MA</td>
<td>3</td>
</tr>
<tr>
<td>Charlotte, NC</td>
<td>2</td>
</tr>
<tr>
<td>Dallas, TX</td>
<td>2</td>
</tr>
<tr>
<td>Detroit, MI</td>
<td>2</td>
</tr>
<tr>
<td>Downtown</td>
<td>2</td>
</tr>
<tr>
<td>Lagos, Nigeria</td>
<td>2</td>
</tr>
<tr>
<td>Los Angeles, CA</td>
<td>2</td>
</tr>
<tr>
<td>Miami, FL</td>
<td>2</td>
</tr>
<tr>
<td>San Francisco, CA</td>
<td>2</td>
</tr>
<tr>
<td>Somewhere</td>
<td>2</td>
</tr>
<tr>
<td>Your head rent free</td>
<td>2</td>
</tr>
</tbody>
</table>

From the following bar chart, we observe that our dataset includes Twitter users all over the United States and even reaches users as far as Lagos, Nigeria. It does make sense that Chicago would hold be the top location since Michael Jordan played for the Chicago Bulls for 14 years which was essentially his entire career.

```r
# Omits NA Locations
mj_goat_tw2 %>%
count(location, sort = TRUE) %>%
mutate(location = reorder(location, n)) %>%
na.omit() %>%
top_n(12) %>%
ggplot(aes(x = location, y = n)) +
geom_col(fill = "red", color = "black") +
coord_flip() +
labs(x = "Count", y = "Location",
     title = "Top Locations of Michael Jordan GOAT Tweets")
```
Similar to the bar chart for Michael Jordan, every other NBA athlete has top locations that spread the U.S. and even extend to other countries/continents. However, some locations listed by users are not real locations but were frequented enough to make the list. Regardless, it is worthwhile to explore all aspects of our data even if that leads to locations such as “Your head rent free.”
Top Locations of LeBron James GOAT Tweets

- United States
- Washington, DC
- Iowa, USA
- Chicago, IL
- San Antonio, TX
- Michigan, USA
- Lagos, Nigeria
- Florida, USA
- Boise, ID
- 2011 and 2007 finals

Top Locations of James Harden GOAT Tweets

- Your head rent free
- NEVADA
- Texas, USA
- nfl
- Houston, TX

Top Locations of Kevin Durant GOAT Tweets

- Los Angeles, CA
- Cleveland, OH
- Boise, ID
- 16 he/him
Similar to other social media applications, Twitter allows users to like/favorite tweets. So, if a tweet has a considerable number of likes it is safe to assume that others share the same opinion and agree with what is being communicated. Using the code below, we can see the top-3 tweets with the most likes/favorites for Michael Jordan. From the output, each tweet relays a positive attitude when it comes to Michael Jordan being considered the GOAT, however, this may not always be the case. Eventually, we will investigate the overall attitude towards Michael Jordan and the other NBA players to see just how positive and/or negative they are.

```
## created_at screen_name favorite_count
## 1 2021-03-28 22:22:59 AllThingsSnyder 387
## 2 2021-03-26 21:50:08 BurnerKhris 356
## 3 2021-04-01 15:17:43 undisputed 252
```

## Word Cloud Analysis

Another text analysis that we are able to observe involves creating a word cloud (Heimerl et al. 2014) which allows us to visualize common words within tweets. What makes this visualization method unique is that the sizing of each word is determined by their frequency which implies their importance/relevance to the overall twitter dataset. So as to gather the dataset containing the top tweeted words, each tweet must be cleaned by removing unnecessary characters and symbols while detecting the strings that are characterized as individual words. In this case we are filtering by regular expressions (Li et al. 2008). Also, *stop words* which are commonly used words that are viewed as unimportant to the text analysis must also be filtered out to shift the focus of the word networks onto the more important word groupings. Once the frequency of each word is accounted for, we can apply the `wordcloud()` function, from the `wordcloud` package.

```
## Top Words
### Michael Jordan
data("stop_words")

words_mj_goat <- mj_goat_tw %>%
  mutate(text = str_remove_all(text, "&amp;|&lt;|&gt;"),
         text = str_remove_all(text,
```
Given that frequency correlates to word size, we can confidently say that “Michael,” “Jordan,” and GOAT are the top words within the dataset. These top words occur 287, 287, and 267 times, respectively, with the next top word being “LeBron” with a frequency of 73. In some sense this outcome was expected, especially with the N.B.A. GOAT debate usually comparing Michael Jordan and LeBron James. Taking a look into LeBron’s dataset of top words, Michael Jordan’s name appears to be the seventh-most frequent word with 57 occurrences.

Note: The following figures may contain inappropriate language, but is included to illustrate the prevalence of such terms within the dataset.
words_harden_goat %>%
  with(wordcloud(word, n, random.order = FALSE,
                  max.words = 100, colors = "black"))

words_kd_goat %>%
  with(wordcloud(word, n, random.order = FALSE,
                 max.words = 100, colors = "gray1"))
words_kobe_goat %>%
  with(wordcloud(word, n, random.order = FALSE,
                 max.words = 100, colors = "yellow2"))

It is interesting to note that most of the word clouds contain words referencing other elite athletes that could be considered the GOAT in their respective sports.

Word Networks

After performing some initial exploratory data analysis on the datasets of our NBA athletes, we can dive deeper into various text analyses such as bigram and trigram (Martin, Liermann, and Ney 1998) analysis. We’ve used the `unnest_tokens()` function to tokenize by a word, but we can also use the function to tokenize by consecutive
sequences of words, called *n-grams* (Cavnar, Trenkle, and others 1994). By determining how often word X is followed by word Y, we can model the relationship between them. This can be done by adding the `token = "ngrams"` argument to `unnest_tokens()` and setting the `n` argument to the number of words we wish to capture in each n-gram.

### Michael Jordan

#### Bigram Analysis

```r
mj_goat_tw_paired_words <- mj_goat_tw %>%
  select(stripped_text) %>%
  unnest_tokens(paired_words, stripped_text, token = "ngrams", n = 2)
```

```r
head(mj_goat_tw_paired_words %>%
  count(paired_words, sort = TRUE), 10) %>%
kable()
```

<table>
<thead>
<tr>
<th>paired_words</th>
<th>n</th>
</tr>
</thead>
<tbody>
<tr>
<td>michael jordan</td>
<td>256</td>
</tr>
<tr>
<td>the goat</td>
<td>155</td>
</tr>
<tr>
<td>is the</td>
<td>92</td>
</tr>
<tr>
<td>jordan is</td>
<td>73</td>
</tr>
<tr>
<td>lebron james</td>
<td>29</td>
</tr>
<tr>
<td>of all</td>
<td>27</td>
</tr>
<tr>
<td>â</td>
<td>24</td>
</tr>
<tr>
<td>u 0001f410</td>
<td>23</td>
</tr>
<tr>
<td>goat michael</td>
<td>22</td>
</tr>
<tr>
<td>goat u</td>
<td>22</td>
</tr>
</tbody>
</table>

```r
mj_goat_tw_sep_words <- mj_goat_tw_paired_words %>%
  separate(paired_words, c("word1", "word2"), sep = " ")
```

```r
mj_goat_tw_filtered_01 <- mj_goat_tw_sep_words %>%
  filter(!word1 %in% stop_words\$word) %>%
  filter(!word2 %in% stop_words\$word)
```

**# new bigram counts:**

```r
mj_goat_tw_bigram_counts <- mj_goat_tw_filtered_01 %>%
  count(word1, word2, sort = TRUE)
```

```r
head(mj_goat_tw_bigram_counts) %>%
kable()
```
To create Michael Jordan’s bigram word network (Zuo, Zhao, and Xu 2016), we must set $n = 2$. In the figure below, we can visualize the relationships between two words whose pairing forms a bigram. Each node represents a word within the filtered dataset and the connection between them is represented by an arrow which begins at word $X$ and points to word $Y$. The frequency of each bigram can be distinguished by the size/boldness of the arrow, like the arrow connecting “Michael” and “Jordan” as compared to the arrow connecting “Steph” and “Curry.” It is fascinating to see that there are multiple bigram chains with the largest located in the bottom left of the figure.

Note: The following figures contain inappropriate language, but is included to illustrate the prevalence of such terms within the dataset.

```
mj_goat_tw_bigram_counts %>%
  filter(n >= 3) %>%
  graph_from_data_frame() %>%
  ggraph(layout = "fr") +
  geom_edge_link(aes(edge_alpha = n, edge_width = n), arrow = a) +
  geom_edge_link(aes(edge_alpha = n, edge_width = n), arrow = a) +
  geom_node_point(color = "red", size = 3) +
  geom_node_text(aes(label = name), vjust = 1.8, size = 3) +
  labs(title = "Bigram Word Network",
       subtitle = "Tweets using Michael Jordan GOAT Keyword",
       x = "", y = "")
```
Like before, Michael Jordan’s trigram word network can be found by adjusting the `unnest_tokens()` function such that \( n = 3 \). Thus, the resulting figure visualizes the relationships between three words whose pairing forms a trigram. Each node represents a word within the filtered dataset and the connection between them is represented by an arrow which begins at word \( X \), points to word \( Y \), and then points to word \( Z \). The frequency of each trigram is also distinguished by the size/boldness of the arrow. Unlike the bigram figure, there are less trigram chains and half of them are extremely bolded. In the table containing the paired words for the bigram and trigram analysis, “â â” and “â â â” occur due to the fact that they are special characters that were included in the “stripped_text” column during the tweet gathering process.

### Trigram Analysis

```r
mj_goat_tw_tri_paired_words <- mj_goat_tw %>%
  select(stripped_text) %>%
  unnest_tokens(paired_words, stripped_text, token = "ngrams", n = 3)

head(mj_goat_tw_tri_paired_words %>%
      count(paired_words, sort = TRUE), 10) %>%
  kable()
```

<table>
<thead>
<tr>
<th>paired_words</th>
<th>n</th>
</tr>
</thead>
<tbody>
<tr>
<td>michael jordan is</td>
<td>70</td>
</tr>
<tr>
<td>is the goat</td>
<td>54</td>
</tr>
</tbody>
</table>
paired_words            n
jordan is the          53
â â â                   20
goat michael jordan    20
the undisputed goat   16
is the undisputed      15
goat of all             14
lebron is the          14
of all sports          14

mj_goat_tw_sep_words_3 <- mj_goat_tw_tri_paired_words %>%
  separate(paired_words, c("word1", "word2", "word3"), sep = " ")

mj_goat_tw_filtered_02 <- mj_goat_tw_sep_words_3 %>%
  filter(!word1 %in% stop_words$word) %>%
  filter(!word2 %in% stop_words$word) %>%
  filter(!word3 %in% stop_words$word)

# new trigram counts:
mj_goat_tw_trigram_counts <- mj_goat_tw_filtered_02 %>%
  count(word1, word2, word3, sort = TRUE)

head(mj_goat_tw_trigram_counts) %>%
kable()

<table>
<thead>
<tr>
<th>word1</th>
<th>word2</th>
<th>word3</th>
<th>n</th>
</tr>
</thead>
<tbody>
<tr>
<td>â</td>
<td>â</td>
<td>â</td>
<td>20</td>
</tr>
<tr>
<td>goat</td>
<td>michael</td>
<td>jordan</td>
<td>20</td>
</tr>
<tr>
<td>baseball</td>
<td>babe</td>
<td>ruth</td>
<td>13</td>
</tr>
<tr>
<td>basketball</td>
<td>michael</td>
<td>jordan</td>
<td>13</td>
</tr>
<tr>
<td>boxing</td>
<td>muhammed</td>
<td>ali</td>
<td>13</td>
</tr>
<tr>
<td>football</td>
<td>tom</td>
<td>brady</td>
<td>13</td>
</tr>
</tbody>
</table>
Replicating the bigram and trigram analysis for LeBron James, James Harden, Kevin Durant, and Kobe Bryant produces the following word network figures.
The varying shapes of the word networks and the bigram/trigram word relationships among the athletes is intriguing to interpret. It is expected that the most bold or one of the most bold n-gram arrow involves the relationship between each player’s first and last name, but the fact that other professional athletes’ names are also present emphasizes the intertwinement of the GOAT debate amongst sports. As long as the search for the GOAT goes on, athletes will continue to be compared and grouped together with those from other sports as Twitter users, fans, and sports media voice their opinion.

**Sentiment Analysis**

To truly understand the connotation behind the GOAT tweets in the datasets involving the NBA athletes, we can perform a text analysis known as sentiment analysis (Feldman 2013). The `get_sentiments()` function offered by the `tidytext` package enables us to retrieve data frames containing words and their corresponding sentiment within a given lexicon (Ding, Liu, and Yu 2008). The available lexicons within the `get_sentiments()` function include “bing,” “afinn,” “loughran,” and “nrc” arguments. What differentiates the lexicons is their word list, the size of each word list, and how the sentiment is evaluated. For example, the `bing` lexicon categorizes sentiment as either “positive” or “negative”; the `afinn` lexicon labels sentiment as numeric values ranging from $[-5, 5]$; the `loughran` lexicon categorizes sentiment as “negative,” “positive,” “litigious,” “uncertainty,” “constraining,” or “superfluous”; the `nrc` lexicon assigns sentiment values consisting of the 8 emotions from *Plutchik’s Wheel of Emotions* (Tromp and Pechenizkiy 2014) to each word.

To see this in action, we can randomly sample 5 rows from each lexicon data frame using the following code. This grants us a glimpse into the lexicons’ word variety and the sentiment values associated with each lexicon.
```r
set.seed(12345)
sample_n(get_sentiments("bing"), 5) %>% kable()

<table>
<thead>
<tr>
<th>word</th>
<th>sentiment</th>
</tr>
</thead>
<tbody>
<tr>
<td>undisputably</td>
<td>positive</td>
</tr>
<tr>
<td>accursed</td>
<td>negative</td>
</tr>
<tr>
<td>bump</td>
<td>negative</td>
</tr>
<tr>
<td>buoyant</td>
<td>positive</td>
</tr>
<tr>
<td>senseless</td>
<td>negative</td>
</tr>
</tbody>
</table>

sample_n(get_sentiments("afinn"), 5) %>% kable()

<table>
<thead>
<tr>
<th>word</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>empathetic</td>
<td>2</td>
</tr>
<tr>
<td>delighting</td>
<td>3</td>
</tr>
<tr>
<td>trauma</td>
<td>-3</td>
</tr>
<tr>
<td>protected</td>
<td>1</td>
</tr>
<tr>
<td>affectionate</td>
<td>3</td>
</tr>
</tbody>
</table>

sample_n(get_sentiments("loughran"), 5) %>% kable()

<table>
<thead>
<tr>
<th>word</th>
<th>sentiment</th>
</tr>
</thead>
<tbody>
<tr>
<td>frivolous</td>
<td>negative</td>
</tr>
<tr>
<td>drag</td>
<td>negative</td>
</tr>
<tr>
<td>quitting</td>
<td>negative</td>
</tr>
<tr>
<td>mediators</td>
<td>litigious</td>
</tr>
<tr>
<td>injures</td>
<td>negative</td>
</tr>
</tbody>
</table>

sample_n(get_sentiments("nrc"), 5) %>% kable()

<table>
<thead>
<tr>
<th>word</th>
<th>sentiment</th>
</tr>
</thead>
<tbody>
<tr>
<td>peaceful</td>
<td>trust</td>
</tr>
<tr>
<td>unhealthy</td>
<td>negative</td>
</tr>
<tr>
<td>cultivate</td>
<td>anticipation</td>
</tr>
<tr>
<td>crowning</td>
<td>positive</td>
</tr>
<tr>
<td>alien</td>
<td>fear</td>
</tr>
</tbody>
</table>

22
When we are ready to perform sentiment analysis (Feldman 2013) on our dataset of tweets, we are only interested in the literal text of the tweet so that the analysis runs smoothly and does not encounter any unnecessary errors. In the following code, it can be seen that a new column variable was created within the original dataset to centralize each tweet’s text, while substituting out the letters that occur in the beginning of a web browser search. Once the text column has been identified, we can create a cleaned dataset that breaks down each tweet in the _stripped_text_ column by word to create a list. Finally, we remove any _stop words_ from the dataset and we arrive at the final product which is listed as _mj_goat_tw_clean_02_.

### Data Cleaning

#### Michael Jordan

```
mj_goat_tw$stripped_text <- gsub("http.*","", mj_goat_tw$text)
mj_goat_tw$stripped_text <- gsub("https.*","", mj_goat_tw$stripped_text)

mj_goat_tw_clean_01 <- mj_goat_tw %>%
  select(stripped_text) %>%
  unnest_tokens(word, stripped_text)

mj_goat_tw_clean_02 <- mj_goat_tw_clean_01 %>%
  anti_join(stop_words)
```

In order to see the sentiment frequencies among the four sentiment lexicons, we can create bar charts that group by each lexicon’s sentiment values and output the most frequent words within those values. Using the cleaned dataset, _mj_goat_tw_clean_02_, we can perform an _inner join_ with the _bing_ lexicon word list which matches sentiment values to the cleaned dataset if a word occurs in both sets. Using that knowledge, we can adjust the dataset to include the number of times each word takes place. In the following figure, we can see the most frequent words being grouped into the “negative” and “positive” sentiment values that the _bing_ lexicon evaluates upon.

Note: _The following figures may contain inappropriate language, but is included to illustrate the prevalence of such terms within the dataset._
If we slightly change the above code such that the inner join is operated on the afinn, loughran, and nrc lexicons, we are able to have more words represented in the figures since the previously noted lexicons offer a greater variety of sentiment values.
Now that we have seen each lexicon’s word frequency variation for Michael Jordan, let us conduct the visualization process for the others as well. Later, we will be determining the sentiment polarity values for each player using the bing lexicon.

By creating a function called `sentiment_bing_score()`, we can input the “text” values from Michael Jordan’s tweet dataset and receive an ordered list of bing sentiment polarities that we can then transform into a readable tibble. One important
facet of the function is that it creates a column score of \(-1\) for words with “negative” sentiment, \(1\) for words with “positive” sentiment, and \(0\) in the case that there are no words in the “text” column for a tweet after being cleaned and filtered. The new tibble can be displayed in a histogram to understand the statistical distribution of the bing sentiment polarities. Repeating this procedure for LeBron James, James Harden, Kevin Durant, and Kobe Bryant assists in comparing the sentiment polarity distributions, visually.

```r
ggplot(mj_goat_tw_sent_score_bing2, aes(x = Score)) + geom_histogram(bins = 15, alpha = 0.9, fill = "red", color = "black") + xlab("Sentiment Polarity: Michael Jordan") + ylab("Count") + theme_minimal()
```

From the histogram, we see that Michael Jordan’s bing sentiment polarity is fairly neutral with a slight advantage on the right which may bring his overall score to being positive. The following code aims to interpret the above histogram by finding the range of values, the overall mean score, and the standard error of that score.

```r
mj_goat_tw_sent_score_bing2$Score %>% summary()
```

```
##    Min. 1st Qu. Median   Mean 3rd Qu.   Max. 
##  -4.0000  0.0000   0.0000  0.07885  1.0000  4.00000
```

```r
tibble(
sent_mean = mean(mj_goat_tw_sent_score_bing2$Score),
sent_err = sd(mj_goat_tw_sent_score_bing2$Score) /
```

26
For Michael Jordan, the bing sentiment polarity scores range from \([-4, 4]\) with a mean and standard error of \(0.079 \pm 0.059\). The purpose of the standard error is to measure the statistical accuracy of the mean, so, the mean is estimated to be between the values of \([0.020, 0.138]\). Therefore, the dataset we gathered and analyzed using the bing lexicon indicates a minor positive sentiment polarity for Michael Jordan.

Moving to LeBron James, we see that he also has a fairly even histogram shape with the majority at 0. Yet, he has a higher frequency of negative scores which may dock his overall polarity.

For LeBron James, the bing sentiment polarity scores range from \([-4, 3]\) with a mean and standard error of \(-0.005 \pm 0.056\). This means that the mean is estimated to be between the values of \([-0.061, 0.051]\). Therefore, the dataset we gathered and analyzed using the bing lexicon indicates a neutral sentiment polarity with a slight lean in the negative direction for LeBron James.
Next, James Harden’s histogram is somewhat even but has been shifted in the negative direction such that it now as a center at about $-1$. This differs from the previous histograms which leads us to believe that his polarity is likely to be negative.

James Harden’s bing sentiment polarity scores range from $[-3, 3]$ with a mean and standard error of $-0.758 \pm 0.093$. As a result, the mean is estimated to be between the values of $[-0.851, -0.665]$. Hence, the dataset we gathered and analyzed using the bing lexicon indicates a negative sentiment polarity for James Harden.
Unlike James Harden, Kevin Durant’s histogram continued the trend of maintaining a distribution that is centered at 0, however, he does have a higher frequency of negative polarity values.

Kevin Durant’s sentiment polarity scores can be seen to range from $[-5, 4]$ with a mean and standard error of $-0.27 \pm 0.12$. Then, the mean can be estimated to be between the values of $[-0.29, -0.15]$. Consequently, the dataset we gathered and analyzed using the bing lexicon indicates a slightly negative sentiment polarity for Kevin Durant.
Kobe Bryant’s sentiment polarity scores range from $[-4, 5]$ with a mean and standard error of $-0.08 \pm 0.16$. The overall mean sentiment polarity can be estimated to be between the values of $[-0.24, 0.08]$. Thus, the dataset we gathered and analyzed using the bing lexicon indicates a slightly negative sentiment polarity with some neutral influence for Kobe Bryant.

```
kobe_goat_tw_sent_score_bing2$Score %>% summary()
##    Min. 1st Qu.  Median      Mean  3rd Qu.     Max.  
## -4.00000  0.00000   0.00000 -0.07843 0.000000  5.00000
```

```
tibble(
  sent_mean = mean(kobe_goat_tw_sent_score_bing2$Score),
  sent_err  =
    sd(kobe_goat_tw_sent_score_bing2$Score) /
    sqrt(length(kobe_goat_tw_sent_score_bing2$Score))
) %>% kable()

<table>
<thead>
<tr>
<th>sent_mean</th>
<th>sent_err</th>
</tr>
</thead>
<tbody>
<tr>
<td>-0.0784314</td>
<td>0.1604971</td>
</tr>
</tbody>
</table>
```

With each NBA athlete’s bing sentiment polarity score having been calculated, they can be ordered from first to last as Michael Jordan, LeBron James, Kobe Bryant,
Kevin Durant, and James Harden, with the first being the most positive and last being the least. There are many factors that can be attributed to a player receiving a positive or negative sentiment polarity score based on tweets but this would require a larger dataset that covers a longer period than $6 - 9$ days.

### Bing Sentiment Polarities of Tweet Frequency Plots

Earlier in the paper, we plotted the tweet frequency by Twitter users for each of the NBA players and their tweet datasets. For each player, there were at least one noticeable spike in tweet frequency which raised interest to understand why it took place and if it was beneficial or detrimental to the sentiment. Using the following code, we can create new datasets which solely contain information applying to the dates of the frequency spikes. Once that is done, we can find the Bing sentiment polarity score like we did in the previous section by taking the mean and standard error.

```r
### Bing Sentiment Score
#### Michael Jordan
mj_goat_tw_freq <-
mj_goat_tw[(mj_goat_tw$created_at >= "2021-03-28 00:00:00" & 
mj_goat_tw$created_at < "2021-03-29 00:00:00"), ]

mj_freq_sent_score_bing <-
lapply(mj_goat_tw_freq$text, 
    function(x){sentiment_bing_score(x)})

mj_freq_sent_score_bing2 <- rbind(
tibble(
    Name = "Michael Jordan", 
    Score = unlist(map(mj_freq_sent_score_bing, "score")), 
    Type = unlist(map(mj_freq_sent_score_bing, "type"))
  )
)

The Bing sentiment polarity score for the day of Michael Jordan’s tweet frequency spike, “2021-03-28,” ranges from $[-3, 2]$ with a mean and standard error of $0.07 \pm 0.11$. This means that the true mean polarity is estimated to be within the values $[-0.04, 0.18]$. This produces a similar mean estimate to that of the overall polarity score, but the spike does have a lower bottom estimate and higher upper estimate. While this frequency spike has a greater chance of producing a negative Bing sentiment polarity, it also has a greater chance for a positive sentiment polarity.

```r
mj_freq_sent_score_bing2$Score %>% summary()
```

```r
## Min. 1st Qu. Median Mean 3rd Qu. Max.  
## -3.00000 0.00000 0.00000 0.07692 0.00000 2.00000  
```
Tweets referring to LeBron James as the GOAT experienced a spike on days of “2021-03-28” and “2021-03-31.” On these dates, the sentiment polarities range between $[-4, 3]$ with a mean and standard error of $-0.14 \pm 0.11$ such that the true mean is estimated to be within the values of $[-0.25, -0.03]$. This frequency spike produced a much more negative bing sentiment polarity when compared to the polarity of his entire dataset. On the above dates the Los Angeles Lakers, who LeBron James plays for, had two games in which they won one and lost the other. LeBron did not play in either game so the negative sentiment does not seem to be the result of his own personal performance, but could have been caused by his own team’s performance and the fact that he did not participate.

Tweets about James Harden had a spike on “2021-03-31.” On this date, the sentiment polarities have a range between $[-3, 0]$ with a mean and standard error of $-1.04 \pm 0.14$ such that the true mean is estimated to be within the values of $[-1.18, -0.90]$. On “2021-03-31” the team that James Harden plays for, the Brooklyn
Nets, had a game that they won against his former team, the Houston Rockets. He participated in the game and had a decent performance in which he scored 17 points, had 6 assists, and 8 rebounds in 27 minutes of play. Despite the victory and performance, the estimated sentiment polarity on this day is about 0.2 more negative than his dataset as a whole.

```
harden_freq_sent_score_bing2$Score %>% summary()
```

```
## Min. 1st Qu. Median Mean 3rd Qu. Max.
## -3.000 -1.000 -1.000 -1.042 -1.000 0.000
```

```
tibble(
  sent_mean = mean(harden_freq_sent_score_bing2$Score),
  sent_err =
    sd(harden_freq_sent_score_bing2$Score) / sqrt(length(harden_freq_sent_score_bing2$Score))
) %>% kable()
```

```
sent_mean sent_err
-1.041667 0.1408973
```

Kevin Durant’s *GOAT* tweets experienced a spike on days of “2021-03-28” and “2021-03-30.” On these dates, the sentiment polarities range between $[-2, 4]$ with a mean and standard error of $-0.02 \pm 0.14$ such that the true mean is estimated to be within the values of $[-0.16, 0.12]$. Kevin Durant also plays on the Brooklyn Nets with James Harden, but there was not a game on the above dates so the increase in frequency was not related to any game performance. However, on “2021-03-30” Kevin Durant and actor, Michael Rapaport, exchanged direct messages which were screenshotted and posted to Twitter by Rapaport. The contents of the messages were not necessarily friendly, yet Kevin Durant’s sentiment polarity is much more neutral and is approximately 0.2 more positive than the original dataset.

```
kd_freq_sent_score_bing2$Score %>% summary()
```

```
## Min. 1st Qu. Median Mean 3rd Qu. Max.
## -2.0000 0.0000 0.0000 -0.02128 0.0000 4.0000
```

```
tibble(
  sent_mean = mean(kd_freq_sent_score_bing2$Score),
  sent_err =
    sd(kd_freq_sent_score_bing2$Score) /
```
Kobe Bryant’s tweet dataset encountered a frequency spike on days of “2021-04-06,” “2021-04-09,” and “2021-04-12.” On these dates, the sentiment polarities range between $[-2, 1]$ with a mean and standard error of $-0.04 \pm 0.15$ such that the true mean is estimated to be within the values of $[-0.19, 0.11]$. The spike in frequency presented a polarity score which is almost identical to that of Kobe’s entire dataset, just a touch more positive. The increase of tweets on “04-12-21” is most likely due to it being the five-year anniversary of his farewell game in which he played his final NBA game and scored 60 points.

After comparing the bing sentiment polarity values for each player in regards to their dataset as a whole and by the spikes in tweet frequency, the frequency spikes were only positive for two of the five NBA athletes.

The following figures represent the histograms of the NBA players from the original bing sentiment polarities along with the sentiment polarities of the frequency spikes as a method of comparison.
From the above results and analyses, the GOAT debate between Michael Jordan and LeBron James can be decided as a victory in the favor of Michael Jordan for having the most positive sentiment polarity of $0.079 \pm 0.059$. LeBron James’ sentiment polarity was not too far behind so it would be interesting to see how much the results vary according to new datasets.

We can also extend our analyses to other sports to determine how their athletes respond to the GOAT debate. With that being said, we may be able to crown a GOAT for each sport using the athletes we sampled from.
NFL

For the NFL, the professional athletes that we gathered Twitter data on include Aaron Rodgers, Jerry Rice, Patrick Mahomes, and Tom Brady. Aaron Rodgers is a quarterback for the Green Bay Packers who won Super Bowl XLV, was named Super Bowl MVP, and is considered to be one of the best quarterbacks in the NFL. Jerry Rice is a former wide receiver who won three Super Bowls (XXIII, XXIV, XXIX), a Super Bowl MVP, and was named to the NFL Hall of Fame in 2010. Patrick Mahomes is a quarterback for the Kansas City Chiefs that won Super Bowl LIV and was named Super Bowl MVP. Finally, Tom Brady is a quarterback for the Tampa Bay Buccaneers who has won seven Super Bowls (XXXVI, XXXVIII, XXXIX, XLIX, LI, LIII, LV), five Super Bowl MVP’s (XXXVI, XXXVIII, XLIX, LI, LV), and is widely considered to be the NFL’s GOAT.

With the above NFL athletes, will focus on the unique and interesting results from the various analyses that the NBA athletes were put through.

Timeline of Tweets - Frequency Plot

**Tom Brady**  To begin, we can take a look at the frequency plots of Tom Brady in both situations of plotting by *hours* and *days*. Within the NFL sample, Tom Brady has the highest frequency of tweets with consistent spikes in activity. The two biggest frequency spikes occurred on “2021-03-27” and “2021-03-28” with approximately 60 and 50 tweets, respectively.
Word Cloud Analysis

Tom Brady  It was also interesting to see how Tom Brady’s word cloud analysis stacked up against the others because his word cloud not only outnumbered the others, but emphasizes his presence and/or dominance within the NFL’s GOAT debate.

Note: The following figures may contain inappropriate language, but is included to illustrate the prevalence of such terms within the dataset

```r
words_tb_goat %>%
  with(wordcloud(word, n, random.order = FALSE,
                   max.words = 100, colors = "red3"))
```
Word Networks

**Tom Brady**  Tom Brady’s bigram and trigram figures also have interesting results as the names of Aaron Rodgers, Patrick Mahomes. It can also be noted that Tom Brady shares a connection with elite athletes in other sports like Michael Jordan, Wayne Gretsky, and Muhammad Ali since they also appear in the networks.

*Note: The following figures may contain inappropriate language, but is included to illustrate the prevalence of such terms within the dataset.*

**Bing Sentiment Polarity**

As we move past the initial analysis phase, we can transition into the sentiment analysis phase to determine each NFL player’s polarity within their dataset. Like before, we can use the results to name a GOAT within the NFL sample and compare them to that of the NBA sentiment polarity values.

**Aaron Rodgers**  Using the same `sentiment_bing_score()` function from the NBA analysis, we can calculate Aaron Rodgers’ sentiment polarity scores to range from $[-2, 5]$ with a mean and standard error of $0.44 \pm 0.23$. Using the standard error, the true polarity is within the values of $[0.21, 0.67]$ which can be interpreted as Aaron Rodgers having a generally positive dataset.

```r
tibble(sent_mean = mean(arodgers_goat_tw_sent_score_bing2$Score),
        sent_err  =
                   sd(arodgers_goat_tw_sent_score_bing2$Score) /
                   sqrt(length(arodgers_goat_tw_sent_score_bing2$Score)))
```
Likewise, we can calculate Jerry Rice’s sentiment polarity scores to range from $[-2, 2]$ with a mean and standard error of $-0.16 \pm 0.14$. Then, the true mean is within the values of $[-0.30, 0.02]$ which means that Jerry Rice’s data has a slightly negative attitude towards him.

Jerry Rice

```
tibble(
    sent_mean = mean(jrice_goat_tw_sent_score_bing2$Score),
    sent_err =
        sd(jrice_goat_tw_sent_score_bing2$Score) /
        sqrt(length(jrice_goat_tw_sent_score_bing2$Score))
) %>% kable()
```

```
<table>
<thead>
<tr>
<th>sent_mean</th>
<th>sent_err</th>
</tr>
</thead>
<tbody>
<tr>
<td>-0.15625</td>
<td>0.1427649</td>
</tr>
</tbody>
</table>
```

Patrick Mahomes

Next, Patrick Mahomes’ sentiment polarity scores seem to range between $[-2, 3]$ with a mean and standard error of $0.24 \pm 0.22$. So, the true mean is within the values of $[0.02, 0.46]$ which is positive but not as much as Aaron Rodgers.

Patrick Mahomes

```
tibble(
    sent_mean = mean(mahomes_goat_tw_sent_score_bing2$Score),
    sent_err =
        sd(mahomes_goat_tw_sent_score_bing2$Score) /
        sqrt(length(mahomes_goat_tw_sent_score_bing2$Score))
) %>% kable()
```

```
<table>
<thead>
<tr>
<th>sent_mean</th>
<th>sent_err</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.2380952</td>
<td>0.2171763</td>
</tr>
</tbody>
</table>
```

Tom Brady

Finally, Tom Brady’s sentiment polarity scores have the greatest range of polarity scores which are between $[-5, 5]$ and have an estimated mean and standard error of $0.371 \pm 0.077$. Therefore, the true polarity is in $[0.294, 0.448]$ which is highly positive. It is even greater than Michael Jordan’s sentiment polarity which was the highest until this point.
Bing Sentiment Polarities of Tweet Frequency Plots

While the sentiment polarities of the spikes in frequency plots for the NBA athletes were not necessarily higher than their overall sentiment polarity, it is sensible to look into how the NFL sample reacts because their response could be entirely different.

**Aaron Rodgers**  Within Aaron Rodgers’ tweet frequency plot which covers the days from “2021-03-27” to “2021-04-03,” there was an increase in tweets on “2021-04-01” and “2021-04-03.” On these days, the sentiment polarity scores have a range of $[-1, 2]$ with a mean and error of $0.38 \pm 0.32$. If the true sentiment polarity for these two frequency spikes is within $[0.06, 0.70]$, then the spikes can be seen as positive influences to the overall sentiment polarity. However, the frequency spikes do not grant a better sentiment polarity since the overall dataset has a greater floor estimate and an almost identical ceiling.

**Jerry Rice**  Jerry Rice’s tweet frequency plot spans from “2021-03-27” to “2021-04-04” with frequency spikes on “2021-04-01” and “2021-04-03.” The sentiment polarity scores for the two days have a range of $[-2, 1]$, as well as a mean and error of $-0.16 \pm 0.18$. Since the true sentiment polarity is within $[-0.34, 0.02]$, then the spikes can be seen as negative influences to the overall sentiment polarity. Also, the tweet
spikes have a worse floor estimate so they are not better than the dataset as a whole.

<table>
<thead>
<tr>
<th>sent_mean</th>
<th>sent_err</th>
</tr>
</thead>
<tbody>
<tr>
<td>-0.1578947</td>
<td>0.1754386</td>
</tr>
</tbody>
</table>

**Patrick Mahomes**  The tweets from Patrick Mahomes’ tweet frequency plot were tweeted between the dates of “2021-03-27” and “2021-04-04” with a surge coming on “2021-04-01.” The sentiment polarity score for this day has a range from \([0, 3]\), along with a mean and error of \(0.60 \pm 0.60\). Given that the true sentiment polarity is within \([0.0, 1.2]\), the increase in frequency was a positive influence on the overall sentiment polarity. On another note, the sentiment polarity for this spike is also greater than the dataset’s making it a successful day.

<table>
<thead>
<tr>
<th>sent_mean</th>
<th>sent_err</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.6</td>
<td>0.6</td>
</tr>
</tbody>
</table>

**Tom Brady**  Lastly, Tom Brady’s tweet frequency plot extends from “2021-03-27” to “2021-04-04” and captured tweet frequency spikes on “2021-03-27” and “2021-03-28.” The sentiment polarity score for these days have a minimum and maximum of \([-3, 5]\), in addition to a mean and error of \(1.41 \pm 0.21\). With the true sentiment polarity of the frequency spikes are in the range of \([1.2, 1.62]\), we are able to consider the spikes as being positive despite being lower than the sentiment polarity calculated in the previous section.
In review, the tweet frequency spikes that took place in the NFL datasets had a better impact than those in the NBA datasets seeing as the sentiment polarities were positive for Patrick Mahomes and Tom Brady. Moreover, we can crown Aaron Rodgers as the GOAT over the other NFL players we researched for having a bing sentiment polarity of 0.44 ± 0.23.

**Comparing Sentiment Histograms**

In the figures below, we can compare the histograms of the sentiment polarity distributions that we computed in the past two sections. While the shapes of the histograms vary among the situations, the biggest change was the decrease in the count of the values which dropped from 150 to 25.

```r
ggplot(nfl_goat_tw_sent_score_bing, aes(x = Score, fill = Name)) + geom_histogram(bins = 15, alpha = 0.9) + facet_grid(~Name) + xlab("Original Sentiment Polarity") + ylab("Count") + theme_minimal()
```
MLB

Another sport that we will be pursuing analytically is baseball and our sampled athletes consist of Clayton Kershaw, Miguel Cabrera, Mike Trout, and Shohei Ohtani of the MLB. Clayton Kershaw is currently a pitcher for the Los Angeles Dodgers who has three Cy Young Awards (2011, 2013, 2014) and won the World Series in 2020. Miguel Cabrera plays for the Detroit Tigers as a first baseman, is a two-time American League MVP (2012, 2013), and won the World Series in 2003. Mike Trout plays center field for the Los Angeles Angels, was selected to the All-MLB First Team in 2019 and 2020, and is a three-time American League MVP (2014, 2016, 2019). Shohei Ohtani also plays for the Los Angeles Angels as a pitcher who is a Japan Series champion (2016), and a Pacific League MVP (2016).

Timeline of Tweets - Frequency Plot

Mike Trout Among the MLB players in our sample, Mike Trout has the most unique and active frequency plot with multiple spikes, but the most significant came on “2021-03-30” and “2021-04-02.” On those days, the second frequency plot allows us to decipher that the number of tweets increased from 1 to 6 and 9 to 18, respectively.
Word Cloud Analysis

**Mike Trout**  
Mike Trout’s word cloud was also the most unique among the others as it consisted of more than just his name and actually referenced baseball terms such as the American and National League, as well as baseball legend Barry Bonds. Michael Jordan’s last name was also referenced in the cloud analysis.

*Note: The following figures may contain inappropriate language, but is included to illustrate the prevalence of such terms within the dataset*

```r
words_mtrout_goat %>%
  with(wordcloud(word, n, random.order = FALSE,
                 max.words = 100, colors = "red"))
```
Word Networks

Shohei Ohtani  For the bigram and trigram word network analysis, Shohei Ohtani’s data performed the best while also having substance. For example, Mike Trout has many connections with his n-gram networks but they are so crowded that it is not readable. In Shohei’s trigram network, it is interesting that all of the word pairings are bold and connect to form interesting shapes. Another interesting fact is that his network only contains baseball terms or other MLB players, not athletes from other sports like in the NBA and NFL word networks.
Bing Sentiment Polarity

Clayton Kershaw  After using the `sentiment_bing_score()` function on Clayton Kershaw’s dataset, we can use the following code to see that the Bing sentiment polarities have a range of $[-2, 1]$. Plus, the sentiment polarities have a mean and standard error of $-0.25 \pm 0.75$ which puts the true polarity value in $[-1.0, 0.5]$. This implies that the tweets in his dataset have a slightly negative connotation.

```r
tibble(
  sent_mean = mean(kershaw_goat_tw_sent_score_bing2$Score),
  sent_err =
    sd(kershaw_goat_tw_sent_score_bing2$Score) /
    sqrt(length(kershaw_goat_tw_sent_score_bing2$Score))
) %>% kable()
```

<table>
<thead>
<tr>
<th>sent_mean</th>
<th>sent_err</th>
</tr>
</thead>
<tbody>
<tr>
<td>-0.25</td>
<td>0.75</td>
</tr>
</tbody>
</table>

Miguel Cabrera  Taking a look into Miguel Cabrera’s sentiment scores, we see that they range from $[0, 4]$ with a mean and error of $1.00 \pm 0.77$. This would put the true polarity value somewhere within $[0.23, 1.77]$. Even with the bounds being so large, the attitudes are still positive.

```r
tibble(
  sent_mean = mean(mcabrera_goat_tw_sent_score_bing2$Score),
  sent_err =
    sd(mcabrera_goat_tw_sent_score_bing2$Score) /
    sqrt(length(mcabrera_goat_tw_sent_score_bing2$Score))
) %>% kable()
```

<table>
<thead>
<tr>
<th>sent_mean</th>
<th>sent_err</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.7745967</td>
</tr>
</tbody>
</table>

Mike Trout  In the case of Mike Trout, he has minimum and maximum polarity bounds of $[-1, 2]$ which obtain a mean and error of $0.000 \pm 0.084$. Due to the true sentiment polarity being in $[-0.084, 0.084]$ such that it is equally as negative as it is positive, the dataset’s tweets portray a neutral opinion of Mike Trout.

```r
tibble(
  sent_mean = mean(mtrout_goat_tw_sent_score_bing2$Score),
```
Finally, after observing Shohei Ohtani’s sentiments we notice that the polarities are captured in a range from $[-2, 1]$. Upon further research, these polarity values possess a mean and error of $-0.14 \pm 0.34$ which places the true sentiment polarity among the values of $[-0.48, 0.20]$. With that said, the tweets in the dataset have a more negative tone towards Shohei than positive.

Bing Sentiment Polarities of Tweet Frequency Plots

Once again, we will gather the sentiment polarity of the spikes in the tweet frequency plots for each MLB athlete before determining whether the spikes had a positive or negative impact on the sentiment polarity of the entire dataset. Meanwhile, it is helpful to understand if the increase in tweet frequencies were positive or negative because that offers a reason for further research into what the cause was.

Clayton Kershaw

Clayton Kershaw’s frequency plot reports on the dates of “2021-03-30” to “2021-04-05,” including the tweet surge on “2021-03-31” and “2021-04-05.” The sentiment polarity values of these days have the minimum and maximum bounds $[-2, 1]$ coupled with a mean and standard error of $-0.67 \pm 0.88$. This leaves us with a sentiment polarity that is mostly negative and between $[-1.55, 0.21]$. Subsequently, the increase in tweet frequencies offer a negative influence that is not a better than the polarity of the entire dataset.
Miguel Cabrera  Secondly, Miguel Cabrera’s frequency plot extends from “2021-04-01” to “2021-04-02.” Within this time frame, there was a single tweet on “2021-04-02” that caused a spike and obtained a value of $4 \pm NA$. Since there is only one tweet, it is not really possible to interpret the results without being biased, especially with the value being 4.

Mike Trout  Next, Mike Trout’s dataset contains tweets from “2021-03-29” to “2021-04-05” which undergo an increase in tweet frequency on the dates of “2021-03-30” and “2021-04-02.” The listed days have sentiment values that range from $[-1, 1]$ which lead to a mean and error of $-0.04 \pm 0.12$. The true sentiment polarity can ultimately be defined within $[-0.16, 0.08]$ which is marginally negative. This also follows the trend of being worse than the dataset’s overall polarity since the original sentiment is neutral.
Shohei Ohtani  Finally, Shohei Ohtani’s dataset captures tweets beginning on “2021-03-30” and through “2021-04-05.” The day that underwent a frequency spike is listed as “2021-04-05” and is understood to have a mean and standard error of 0.25 ± 0.25, including minimum and maximum values of [0, 1]. This presents us with a true sentiment polarity that can be defined within 0.0 ± 0.5 which is positive and greater than Shohei’s sentiment that includes the entire dataset.

tibble(
    sent_mean = mean(sohtani_freq_sent_score_bing2$Score),
    sent_err =
        sd(sohtani_freq_sent_score_bing2$Score) /
        sqrt(length(sohtani_freq_sent_score_bing2$Score))
  ) %>% kable()

<table>
<thead>
<tr>
<th>sent_mean</th>
<th>sent_err</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.25</td>
<td>0.25</td>
</tr>
</tbody>
</table>

To sum up the sentiment analysis (Feldman 2013) of the MLB players, Shohei Ohtani was the only player whose spike in tweet frequency had positive sentiment polarity and was positively influential to the polarity of their dataset as a whole. Furthermore, Miguel Cabrera’s sentiment polarity of 1.00 ± 0.77 was the highest among the others in the sample, thus, we can crown him the GOAT of our MLB sample.

Comparing Sentiment Histograms

In the figures below, we are able to compare the histograms of the sentiment polarity distributions that we computed in the prior two sections. The shapes of the histograms marginally shift from one figure to the next, but the main differences are the decrease in the count of the sentiment values which dropped from 40 to 15, and Miguel Cabrera’s lack of negative values in the sentiment frequency histogram.

```r
ggplot(mlb_goat_tw_sent_score_bing, aes(x = Score, fill = Name)) +
  geom_histogram(bins = 15, alpha = 0.9) +
  facet_grid(~Name) +
```
NCAA Basketball

After focusing on the NBA, NFL, and MLB, I thought that it would be interesting to shift the focus to analyzing the NCAA. The NCAA Division I Men’s and Women’s Basketball Tournaments both take place in March, and given their vast popularity and media coverage, I took the opportunity to include two players from each tournaments into my NCAA Basketball sample. On the men’s side, I selected Drew Timme and
Jalen Suggs who both played for the Gonzaga Bulldogs, the runner-ups in the 2021 championship game. For the women, I selected Aari McDonald who played for the Arizona Wildcats, the runner-ups in the 2021 championship game. I also chose Paige Bueckers who plays for the UConn Huskies as she lead her team to the Final Four and became the first freshman to win AP Player of the Year, Naismith Trophy, Wooden Award POY, and the Nancy Lieberman Award.

**Timeline of Tweets - Frequency Plot**

*Paige Bueckers* From Paige Buecker’s tweet frequency plot below, she has a high count of tweets referring to her as the *GOAT* as well as surges that appear throughout the entire dataset. The most notable and drastic peaks could be viewed as taking place on “2021-03-30” and “2021-04-01.” The second frequency plot looks at the dataset from another angle by adjusting the time interval to group tweets by *days* instead of *hours.*
Word Cloud Analysis

**Jalen Suggs**  
A few things to notice about Jalen Suggs’ word cloud is that Paige Bueckers’ name appears, the word “final” is also mentioned which may be referring to the championship game, and that “buzzer beater” is listed because of his game-winning shot in the semi-finals against the University of California, Los Angeles (UCLA) that advanced Gonzaga to the finals.

*Note: The following figures may contain inappropriate language, but is included to illustrate the prevalence of such terms within the dataset*
Paige Bueckers  Similarly, Jalen Suggs’ name appears in Paige’s word cloud. Paige’s word cloud contains the most words among her fellow NCAA basketball players with some referencing her achievements and extraordinary skills.

Word Networks

Jalen Suggs  Using the bigram and trigram analyses, we are able to see the connections between words that are in pairs and in trios. After seeing the top words in his word cloud, it is interesting to see how those words are all connected.

Paige Bueckers  Using the bigram and trigram analyses, we are also able to see the connections between w pairs and trios of words in Paige’s dataset. One interesting point is that Paige has a stronger connection to the term GOAT in both the bigram and trigram plots when compared to Jalen’s figures.
Bing Sentiment Polarity

As we did for the other sports and athletes, we will use the `sentiment_bing_score()` function to calculate the Bing sentiment polarity for the NCAA athletes’ datasets.

**Drew Timme**  
Drew Timme’s sentiment polarities can be found within the range $[-1, 2]$, meanwhile, the sentiment polarity has a mean and standard error of $0.06 \pm 0.14$. Adjusting for the error produces a true sentiment polarity between $[-0.08, 0.20]$ which is slightly positive, overall.

```r
tibble(
  sent_mean = mean(timme_goat_tw_sent_score_bing2$Score),
  sent_err  =
)```
Jalen Suggs  

Jalen Suggs’ sentiment polarities can be found ranging between $[-1, 3]$, with a sentiment polarity that has a mean and standard error of $0.030 \pm 0.067$. Adjusting for the error produces a true sentiment polarity between $[-0.037, 0.097]$ which is slightly positive, like Drew Timme.

```
tibble(
  sent_mean = mean(jsuggs_goat_tw_sent_score_bing2$Score),
  sent_err =
    sd(jsuggs_goat_tw_sent_score_bing2$Score) /
    sqrt(length(jsuggs_goat_tw_sent_score_bing2$Score))
) %>% kable()
```

<table>
<thead>
<tr>
<th>sent_mean</th>
<th>sent_err</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.03</td>
<td>0.0673525</td>
</tr>
</tbody>
</table>

Aari McDonald  

Aari McDonald’s sentiment polarity can be found within the values $[-1, 0]$, with a sentiment polarity that has a mean and standard error of $-0.091 \pm 0.091$. Adjusting for the error produces a true sentiment polarity between $[-0.182, 0.000]$ which is slightly slightly negative.

```
tibble(
  sent_mean = mean(amcdonald_goat_tw_sent_score_bing2$Score),
  sent_err =
    sd(amcdonald_goat_tw_sent_score_bing2$Score) /
    sqrt(length(amcdonald_goat_tw_sent_score_bing2$Score))
) %>% kable()
```

<table>
<thead>
<tr>
<th>sent_mean</th>
<th>sent_err</th>
</tr>
</thead>
<tbody>
<tr>
<td>-0.0909091</td>
<td>0.0909091</td>
</tr>
</tbody>
</table>
**Paige Bueckers**  
Paige Buecker’s sentiment polarity can be found in the range of values with a minimum and maximum of $[-1, 3]$, such that the polarity that has a mean and error of $0.37 \pm 0.10$. Adjusting for the error produces a true sentiment polarity between $[0.27, 0.47]$ which is positive, and the highest among the other NCAA athletes.

```r
tibble(
  sent_mean = mean(pbueckers_goat_tw_sent_score_bing2$Score),
  sent_err =
    sd(pbueckers_goat_tw_sent_score_bing2$Score) / sqrt(length(pbueckers_goat_tw_sent_score_bing2$Score))
) %>% kable()
```

<table>
<thead>
<tr>
<th>sent_mean</th>
<th>sent_err</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.369863</td>
<td>0.1004307</td>
</tr>
</tbody>
</table>

**Bing Sentiment Polarities of Tweet Frequency Plots**

### Frequency Bing Sentiment Score

Drew Timme’s tweet dataset ranges from “2021-03-30” to “2021-04-06” in which there were two spikes on
“2021-04-05” and “2021-04-06.” On those days, the bing sentiment polarity scores range from $[0, 2]$ with a mean and standard error of $0.25 \pm 0.25$ meaning that the true mean is likely between the values of $[0.0, 0.5]$. These values are much more positive than Timme’s overall sentiment polarity score, so the spike acted in his favor.

```
tibble(
  sent_mean = mean(timme_freq_sent_score_bing2$Score),
  sent_err =
    sd(timme_freq_sent_score_bing2$Score) /
    sqrt(length(timme_freq_sent_score_bing2$Score))
) %>% kable()
```

<table>
<thead>
<tr>
<th>sent_mean</th>
<th>sent_err</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.25</td>
<td>0.25</td>
</tr>
</tbody>
</table>

**Jalen Suggs** Jalen Suggs’ tweet dataset ranges from “2021-03-31” to “2021-04-06” where he experienced a spike in frequency on “2021-04-04.” The sentiment polarity ranges from $[-1, 3]$ with a mean and standard error of $0.024 \pm 0.076$. This would place the true mean between $[-0.052, 0.1]$ which also turns out to be more positive than his original sentiment polarity.

```
tibble(
  sent_mean = mean(jsuggs_freq_sent_score_bing2$Score),
  sent_err =
    sd(jsuggs_freq_sent_score_bing2$Score) /
    sqrt(length(jsuggs_freq_sent_score_bing2$Score))
) %>% kable()
```

<table>
<thead>
<tr>
<th>sent_mean</th>
<th>sent_err</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0238095</td>
<td>0.0756994</td>
</tr>
</tbody>
</table>

**Aari McDonald** Aari McDonald’s dataset that covers the dates from “2021-04-03” to “2021-04-05” underwent a couple spikes on “2021-04-03” and “2021-04-05” and offer polarities that range from $[-1, 0]$ with a mean and standard error of $-0.17 \pm 0.17$. This gives a true mean within the values of $[-0.34, 0]$, proving that the spike was detrimental since the floor estimate almost doubled in value.

```
tibble(
  sent_mean = mean(amcdonald_freq_sent_score_bing2$Score),
  sent_err =
```
### Paige Bueckers

Finally, the tweet dataset for Paige Bueckers captures the dates “2021-03-29” to “2021-04-05,” with spikes on “2021-03-30” and “2021-04-01.” On these dates, the sentiment polarity has a minimum and maximum of $[-1, 3]$ with a mean and standard error of $0.22 \pm 0.10$. With the true estimate being between $[0.12, 0.32]$, these values are still smaller than the overall dataset but has more of a neutral effect since they are positive.

```r
tibble(
  sent_mean = mean(pbueckers_freq_sent_score_bing2$Score),
  sent_err =
    sd(pbueckers_freq_sent_score_bing2$Score) /
    sqrt(length(pbueckers_freq_sent_score_bing2$Score))
) %>% kable()
```

<table>
<thead>
<tr>
<th>sent_mean</th>
<th>sent_err</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.2195122</td>
<td>0.1018858</td>
</tr>
</tbody>
</table>

After comparing each NCAA basketball player’s Bing sentiment polarity of their entire dataset to their spikes in tweet frequency, the frequency spikes’ polarity was more positive, or better, for only one of the four players. On a positive note, Paige Buecker has earned the title of GOAT among the NCAA Basketball athletes within our sample after earning a Bing sentiment polarity score of $0.37 \pm 0.10$.

### Comparing Sentiment Histograms

The following figures represent the histograms of the NCAA basketball players from the original Bing sentiment polarities along with the sentiment polarities of the frequency spikes as a method of comparison.
Comparisons Across Sports

The two athletes with the highest bing sentiment polarity are Miguel Cabrera of the MLB and Aaron Rodgers of the NFL with scores of $1.00 \pm 0.77$ and $0.44 \pm 0.23$, respectively. On the other hand, the two athletes with the lowest bing sentiment polarity are James Harden of the NBA and Aari McDonald of NCAA Women’s Basketball with scores of $-0.758 \pm 0.093$ and $-0.17 \pm 0.17$, respectively. This shows
that athletes receive criticism and praise even as professionals who are at the top of their game. To summarize, the GOATs within our four-sport sample are Michael Jordan for the NBA, Aaron Rodgers for the NFL, Miguel Cabrera for the MLB, and Paige Bueckers for NCAA Women’s Basketball.

**Conclusion**

Regardless of the results that were presented by the various analyses performed on the athletes’ datasets, there are always improvements that can be made. To begin, it would be more insightful to increase the sample size of the athletes within each sport along with the number of sports being analyzed. This would allow for the ability to compare and contrast on a greater scale, whether it is internally or externally. Another improvement would involve gathering more tweets per athlete, over period of time that is greater than $6 - 9$ days. This would improve the accuracy of the bing sentiment scores by decreasing the standard error. Increasing the time period would also make it easier to locate frequency trends which could assist in gaining a better understanding of how to predict an athlete’s sentiment by using results based on prior events. One final improvement is simply recreating the analysis multiple times because social media can be more subjective than factual, meaning that opinions are easily changed. This could be represented as an athlete’s sentiment polarity being positive one day but the next it changes to being negative due to a bad performance or some other “negative” event. Although the Twitter API (Makice 2009) did present some limitations, it was a valuable experience learning to use access real Twitter data and exploring the capabilities that it had to offer.

In conclusion, the answer to the main question that motivated this entire project and analysis is that Michael Jordan has been confirmed to be the GOAT over LeBron James based on the bing sentiment polarities that were calculated using the tweet datasets gathered by the Twitter API (Makice 2009).
Bibliography


Appendix

Code that was applied to Michael Jordan’s tweet data but not included in the paper can be found here. As far as the other athletes involved, their results are replicated using the same code but utilize their dataset’s name in place of the mj seen below.

Libraries

```r
# Set-up
library(rtweet)
library(ggmap)
library(igraph)
library(ggraph)
library(tidytext)
library(ggplot2)
library(dplyr)
library(readr)
library(magrittr)
library(wordcloud)
library(widyr)
library(tidyr)
library(utils)
library(wordcloud)
library(purrr)
library(stringr)
library(knitr)
library(grid)
library(ggpubr)
library(tinytex)
```

Code

```r
## Michael Jordan
### Code Collection

mj_goat_tw <-
search_tweets(
  q = "Michael Jordan GOAT OR Michael Jordan Goat",
  n = 300,
  include_rts = FALSE,
  `~filter` = "replies",
  lang = "en")
```
### Exporting tweets to CSV

write_as_csv(mj_goat_tw, "mj_goat_tw.csv")

### Most Liked Tweets

mj_goat_tw %>%
  arrange(-favorite_count) %>%
  top_n(3, favorite_count) %>%
  select(created_at, screen_name, favorite_count)

### Arrow within bigram/trigram networks

a <- arrow(length = unit(.075, "inches"), type = "closed")

### Trigram plot

mj_goat_tw_trigram_counts %>%
  filter(n >= 3) %>%
  graph_from_data_frame() %>%
  ggraph(layout = "fr") +
  geom_edge_link(aes(edge_alpha = n, edge_width = n), arrow = a) +
  geom_edge_link(aes(edge_alpha = n, edge_width = n), arrow = a) +
  geom_edge_link(aes(edge_alpha = n, edge_width = n), arrow = a) +
  geom_node_point(color = "red", size = 3) +
  geom_node_text(aes(label = name), vjust = 1.8, size = 3) +
  labs(title = "Trigram Word Network",
       subtitle = "Tweets using Michael Jordan GOAT Keyword",
       x = "", y = "")

### Arranging bigram/trigram plots

ggarrange(lebron_bigram_plot, lebron_trigram_plot, ncol = 2)
ggarrange(harden_bigram_plot, harden_trigram_plot, ncol = 2)
ggarrange(kd_bigram_plot, kd_trigram_plot, ncol = 2)
ggarrange(kobe_bigram_plot, kobe_trigram_plot, ncol = 2)
## Michael Jordan
### Bing Lexicon Counts/Histogram

```r
mj_goat_tw_bing_word_counts <- mj_goat_tw_clean_02 %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort = TRUE) %>%
  ungroup()

mj_goat_tw_bing_word_counts %>%
  group_by(sentiment) %>%
  top_n(10) %>%
  ungroup() %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(word, n, fill = sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free_y") +
  labs(title = "Sentiment of Michael Jordan Goat Tweets using Bing Lexicon",
       y = "Word Frequency",
       x = NULL) +
  coord_flip()
```

## Michael Jordan
### AFINN Lexicon Counts/Histogram

```r
mj_goat_tw_afinn_word_counts <- mj_goat_tw_clean_02 %>%
  inner_join(get_sentiments("afinn")) %>%
  count(word, value, sort = TRUE) %>%
  ungroup()

mj_goat_tw_afinn_word_counts %>%
  group_by(value) %>%
  top_n(4) %>%
  ungroup() %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(word, n, fill = value)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~value, scales = "free_y") +
  labs(title = "Sentiment of Michael Jordan Goat Tweets using AFINN Lexicon",
       y = "Word Frequency",
       x = NULL) +
  coord_flip()
```
## Michael Jordan
### Loughran Lexicon Counts/Histogram

```r
mj_goat_tw_loughran_word_counts <- mj_goat_tw_clean_02 %>%
  inner_join(get_sentiments("loughran")) %>%
  count(word, sentiment, sort = TRUE) %>%
  ungroup()

mj_goat_tw_loughran_word_counts %>%
  group_by(sentiment) %>%
  top_n(5) %>%
  ungroup() %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(word, n, fill = sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free_y") +
  labs(title = "Sentiment of Michael Jordan Goat Tweets using Loughran Lexicon",
       y = "Word Frequency",
       x = NULL) +
  coord_flip()
```

## Michael Jordan
### NRC Lexicon Counts/Histogram

```r
mj_goat_tw_nrc_word_counts <- mj_goat_tw_clean_02 %>%
  inner_join(get_sentiments("nrc")) %>%
  count(word, sentiment, sort = TRUE) %>%
  ungroup()

mj_goat_tw_nrc_word_counts %>%
  group_by(sentiment) %>%
  top_n(5) %>%
  ungroup() %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(word, n, fill = sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free_y") +
  labs(title = "Sentiment of Michael Jordan Goat Tweets using NRC Lexicon",
       y = "Word Frequency",
       x = NULL) +
  coord_flip()
```
## Bing Sentiment Score Function

```r
sentiment_bing_score <- function(twt) {
  # Step 1: Perform text cleaning (on tweet)
  twt_tbl = tibble(text = twt) %>%
    mutate(
      # Remove http elements manually
      stripped_text = gsub("http\S+", "", text)
    ) %>%
    unnest_tokens(word, stripped_text) %>%
    anti_join(stop_words) %>%
    inner_join(get_sentiments("bing")) %>%
    count(word, sentiment, sort = TRUE) %>%
    ungroup()
  # Create a column "score" that assigns a -1 to all negative words
  # and 1 to all positive words
  mutate(
    score = case_when(
      sentiment == 'negative' ~ n*(-1),
      sentiment == 'positive' ~ n*(1)
    )
  )

  # Calculate the total score
  sentiment.score = case_when(
    nrow(twt_tbl) == 0 ~ 0, # if there are no words, score is 0
    nrow(twt_tbl) > 0 ~ sum(twt_tbl$score) # sum the positive & negatives
  )

  # This is to keep track of which tweets
  # contained no words at all from the bing list
  zero.type = case_when(
    nrow(twt_tbl) == 0 ~ "Type 1", # Type 1: no words at all, zero = no
    nrow(twt_tbl) == 0 ~ "Type 2" # Type 2: zero means sum of words = 0
  )

  list(score = sentiment.score, type = zero.type, twt_tbl = twt_tbl)
}
```

### Michael Jordan

#### Applying Bing Sentiment Score Function

```r
mj_goat_tw_sent_score_bing <- lapply(mj_goat_tw$text, function(x){sentiment_bing_score(x)})
```
mj_goat_tw_sent_score_bing2 <- rbind(
  tibble(
    Name = "Michael Jordan",
    Score = unlist(map(mj_goat_tw_sent_score_bing, "score")),
    Type = unlist(map(mj_goat_tw_sent_score_bing, "type"))
  ),
  tibble(
    Name = "LeBron James",
    Score = unlist(map(lebron_goat_tw_sent_score_bing, "score")),
    Type = unlist(map(lebron_goat_tw_sent_score_bing, "type"))
  ),
  tibble(
    Name = "James Harden",
    Score = unlist(map(harden_goat_tw_sent_score_bing, "score")),
    Type = unlist(map(harden_goat_tw_sent_score_bing, "type"))
  ),
  tibble(
    Name = "Kevin Durant",
    Score = unlist(map(kd_goat_tw_sent_score_bing, "score")),
    Type = unlist(map(kd_goat_tw_sent_score_bing, "type"))
  ),
  tibble(
    Name = "Kobe Bryant",
    Score = unlist(map(kobe_goat_tw_sent_score_bing, "score")),
    Type = unlist(map(kobe_goat_tw_sent_score_bing, "type"))
  )
)

## NBA
### Bing Sentiment

nba_goat_tw_sent_score_bing <- rbind(
  tibble(
    Name = "Michael Jordan",
    Score = unlist(map(mj_goat_tw_sent_score_bing, "score")),
    Type = unlist(map(mj_goat_tw_sent_score_bing, "type"))
  ),
  tibble(
    Name = "LeBron James",
    Score = unlist(map(lebron_goat_tw_sent_score_bing, "score")),
    Type = unlist(map(lebron_goat_tw_sent_score_bing, "type"))
  ),
  tibble(
    Name = "James Harden",
    Score = unlist(map(harden_goat_tw_sent_score_bing, "score")),
    Type = unlist(map(harden_goat_tw_sent_score_bing, "type"))
  ),
  tibble(
    Name = "Kevin Durant",
    Score = unlist(map(kd_goat_tw_sent_score_bing, "score")),
    Type = unlist(map(kd_goat_tw_sent_score_bing, "type"))
  ),
  tibble(
    Name = "Kobe Bryant",
    Score = unlist(map(kobe_goat_tw_sent_score_bing, "score")),
    Type = unlist(map(kobe_goat_tw_sent_score_bing, "type"))
  )
)

## NBA
### Bing Sentiment of Frequency Plots

nba_freq_sent_score_bing <- rbind("
<table>
<thead>
<tr>
<th>Name</th>
<th>Score</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Michael Jordan</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LeBron James</td>
<td></td>
<td></td>
</tr>
<tr>
<td>James Harden</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Kevin Durant</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Kobe Bryant</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>