# Modeling Racially Disproportionate Language on Twitter during NFL Gameplay

A Thesis

Presented to

The Division of

Smith College

In Partial Fulfillment

of the Requirements for the Degree

Bachelor of Arts

Julianna Alvord

May 13, 2019

Approved for the Division

(Statistical & Data Sciences)

Ben Baumer

Randi Garcia

# Acknowledgements

I want to thank a few people.

# Table of Contents

# List of Tables

# List of Figures

# Abstract

The preface pretty much says it all.

Second paragraph of abstract starts here.

# Chapter 1

# Introduction

In August of 2016, San Francisco 49ers quarterback Colin Kaepernick remained sitting by the team's water coolers during the national anthem prior to a preseason game. Immediately, the gesture was widely questioned and criticized. Many fans found his actions to be disrespectful to the sacrifices made by military members. After the game, when asked by the NFL media why he refused to stand, Kaepernick responded by stating, "I am not going to stand up to show pride in a flag for a country that oppresses black people and people of color" (Hauser, 2016). His decision, along with his eventual exit from the NFL, brought attention to racial injustices, peaceful protest, and their connection to sports.

This event, while drawing criticism, also sparked conversations where the main theme ranged from disappointment to hatred. The hateful language that flooded social media did not go unnoticed, and the quarterback's image was tarnished beyond repair. Other black athletes, especially those who support Kaepernick, paid a similar price. This event created an unbreakable connection between football and racially charged language. Though Kaepernick is no longer a part of the NFL, his legacy, and the anger he fueled, is still evident. The analyses by the media focused mainly on the negative language aimed both at Kaepernick and other black players. However, most of these accounts were anecdotal, referencing a few, seemingly isolated events. The extensive aftermath and lack of data-driven research on the topics addressed during this event were the motivation behind this work.

We believe that this negative language towards black players in the NFL is widespread and measurable. Given the relatively new techniques to analyze language and sentiment on a larger scale, research in this area is increasingly accessible to data scientists. Our goal is to quantify and model the accounts of racially disproportionate language on social media using these techniques in order to offer insight into the connection between racial inequality and sport.

This research was conducted through two studies, both of which utilized data from Twitter. The first one focused in on Super Bowl LIII and allowed for what was used as an exploratory data analysis for the second, more balanced study. Here, tweets were gathered for a selected subset of quarterbacks and receivers, half of whom are white and half of whom are black. Our three-level multilevel model was created using the data from this study.

## 1.1 Background

Do I need to include more information here?

### 1.1.1 Research Question

The main research question is as follows: Are sentiments on Twitter more negative towards black NFL players during a failed outcome (a loss) and more positive towards white NFL players during a successful outcome (a win)?

### 1.1.2 Sentiment Analysis

Research dating back as far as 2003 has illustrated the use of sentiment analysis to identify how sentiments are expressed and whether the expressions in a determined body of text indicate positive or negative opinions towards the subject (Nasukawa & Yi, 2003). Sentiments, when done so by humans, are determined by processing the language and context of a text. Sentiment analysis works in a similar way, but also allow for a more automated process of long or multiple texts.

Two types of methods for sentiment analysis are often employed and discussed throughout the literature: machine-learning-based and lexical-based. Machine-learning approaches mainly rely on supervised classification methods, where labeled training data is required. This method can be easily specified to fit one's data. However, there are issues of over-specification to training data which leads to low applicability to testing data. Lexical-based methods use a predetermined list of words, known as a lexicon, where each word is matched to one or more sentiments. In order to achieve a similar level of specification as the machine-learning-based methods, these lexicons must be thorough and contextual to the project at hand (Gonçalves, Araújo, Benevenuto, & Cha, 2013).

Sentiment analysis is not a simple approach. The decision to use this method in a project requires an understanding of the levels of granularity that can be specified. Early research tended to analyze sentiments of entire documents, which can lower the accuracy of the overall polarity assessment (Zhang, Zeng, Li, Wang, & Zuo, 2009). Instead, other researchers employ sentence-level analyses in order to focus on pieces, such as conjunctions, which can significantly change overall sentiment (Meena & Prabhakar, 2007).

For projects, such as this one, whose main purpose is to employ sentiment analysis to research a given topic instead of researching sentiment analysis itself, there will be issues of accuracy. The reason for this is that basic sentiment analyses focus on words individually to determine overall sentiment. An example of inaccuracy would be the fact that conjunctions will not be addressed. An example of this would be the phrase "not good". Using the method introduced below, the phrase would be separated into "not" and "good", where "good" would be categorized as positive and not would be ignored altogether.

A simple process for conducting basic natural language processing and sentiment analyses in R is detailed in the book titled Text Mining with R: A Tidy Approach (Silge & Robinson, 2018). Here, the authors utilize the tidy text format, which is described as one-token-per-row. Therefore, each word is a row which allows for an easy join to a lexicon that is specified to the project.

### 1.1.3 Sentiment Analyses of Twitter and other Online Data

Given the goal of this project is to measure sentiments of football fans by obtaining social media data, it was necessary to determine which site would offer the proper information. We found that Twitter data is commonly used for large-scale sentiment analyses as it is a "massive social networking site" aimed at quick communication. Approximately 400 million tweets are published daily from Twitter's 140 million active users (Kumar, Morstatter, & Liu, 2014). Due to its constant stream of data, many researchers utilize Twitter data for sentiment analysis projects.

Our hypothesis that negative sentiments will be widespread throughout Twitter is backed by much research. A study by Awan (2014) examined 500 tweets to determine how Muslim individuals are being viewed and targeted by Twitter users. The authors found that "the Internet and social media sites such as Twitter have become a popular arena for online hate, partly due to their accessibility and the anonymity they offer for offenders who use it to intimidate, harass, and bully others."

Anonymity plays a huge role in online hate and abuse found on Twitter. Christopherson (2007) discussed the positive and negative results of online anonymity. On the positive side, they offered privacy as an example. Online users can decide how much of themselves to share with others, which can have a positive effect on psychological well-being. On the negative side, the authors connect the theory of group polarization to online anonymity. They define group polarization as the "tendency for like-minded individuals to become more extreme in their thinking following a group discussion." To connect back to the current project, our hypothesis is supported by our belief that many football fans are like-minded and Twitter offers an anonymous platform for anti-black sentiments to become more extreme.

Cleland (2014) found specific evidence of racism online in a sports context. Their study examined 500 posts from an online message board to determine the level and nature of racist language. They found extensive racist and Islamaphobic examples throughout the platform, many of which took place during fan interaction.

Focusing on Twitter, sports fan existence has increased in the past few years with the introduction of game-specific hashtags that guide conversation (Weller, Bruns, Burgess, Mahrt, & Puschmann, 2014). Expanded Twitter activity prompts fan interactions and further data of game reactions, making tweets the ideal text for this project.

### 1.1.4 Multi-Level Modeling

This project uses data that is grouped at different levels, where the processes occurring at a higher level influence the processes of a lower level. The levels in this case include race, team, time, outcome, and individual players. Data that is structured in a heirarchical way is often found within social science contexts, due especially to natural groupings that occur with humans. In order to test the hypothesis and answer the research question for this project, multilevel models are necessary. If we chose to use another modeling technique in this situation, serious theoretical and statistical issues would be present.

Two fallacies that can occur when applying non-multilevel modeling techniques on leveled data are ecological fallacies and atomistic fallacies. The first occurs when patterns observed for groups are assumed to hold true for individuals when this is not necessarily true. The latter occurs when the opposite occurs, namely that patterns observed for individuals are assumed to hold true for groups (Luke, 2004).

## 1.2 Racism in Sport

Though Colin Kaepernick's actions resulted in widespread media attention, many previous studies have addressed issues of racism and racial inequality in football and sports. Most often, these topics are studied through the context of sports media. Early research began in 1977, where Rainville & McCormick (1977) analysed covert forms of prejudice in television broadcasts of nationally televised football games. They found that white players were more likely to receive praise for "good" plays and less likely to receive negative feedback for "bad" plays when compared to their black counterparts. More recently, a 2014 study by Angelini, Billings, MacArthur, Bissell, & Smith (2014) analysed broadcasts from the 2012 London Summer Olympics to determine whether they revealed significant divergences in dialogues for athletes of different racial groups. During this time, white athletes were more likely to be mentioned and there were significant differences in the commentary for white, black, latino, asian, and middle eastern athletes.

The relationship between racism and sport does not simply lie within the media. However, racism in other areas of sports is not easy to determine. Hylton & Lawrence (2016) argues that while it is important to contront frontstage racism, such as that observed in the media and in other, more explicit ways, researchers must be mindful of more subvert forms of racism within sport. Research on subvert forms of racism, such as that from fans, is few and far between. One reason for this may be due to lack of access to data. Many studies surrounding racist dialogue among fans rely on survey data. Research from Cleland & Cashmore (2014) drew from 2,500 responses from soccer fans to an anonymous survey, which aimed to examine the extent of racism within the sport. They found that half of respondents either witness or experience racism in some form.

As seen from Kaepernick's exile and from these previous studies, racism in the NFL, whether apparent or more subvert, is widespread. Though individuals questioning the protest cite military disrespect as their main concern, a study from Intravia, Piquero, & Piquero (2018) found that on average, black respondents were more likely to support all types of anthem protests and believe players who choose to participate should not be punished, compared to white repondents. Those strong race effects remain even after controlling for several key correlates, suggesting that race is a key distinction between support and dissapproval of these NFL anthem protests.

Research on racism, both in sport and across other areas of society, is vital to the wellbeing of people and athletes of color. Numerous studies investigate the relationships between racism and mental and physical health. Williams (1999) found that stress due to stigma, along with individual and institutional discrimination can adversely affect health. Additionally, the study addressed how racism in the United States is, "responsible for the development of an organized system of policies and practices designed to create racial inequality." Due to the powerful structure of this system, people of color are often forced to hide their experiences. This prejudicial system leads to negative outcomes for people of color, and this is no different in the context of sports. In a study by Burdsey (2011), researchers found that all athletes, both racial minorities and racial majorities, tend to downplay the reprecussions of racial microaggressions. Research using large-scale data focusing on racist and racially disproportionate language is necessary to bring attention to issues of racism in sport context and the subsequent extensive negative consequences.

## 1.3 Contribution

### 1.3.1 Reproducible Research

As statistical analyses of data become increasingly common and complex, reproducibility becomes increasingly necessary. In order for research to be reproducible, proper documentation of methodology is crucial. However, this is not practiced by many researchers, especially due to the lack of proper technology. One way to improve statistical reproducibility, a term used by Victoria Stodden Stodden (2014), is by utilizing RMarkdown, an open source markup language. This tool allows for better workflow, by combining a statistical package (R R Core Team (2013)) with a layout package (Baumer, Cetinkaya-Rundel, Bray, Loi, & Horton, 2014).

Given both the importance of reproducibility and the social relevance of this project, we utilize RMarkdown throughout a majority of the cleaning and analyzing processes. However, gathering the data for one aspect this project required the use of Python. The code for this process was written using Jupyter Notebook, an open-source web application.

All of the documents from this project, whether they be RMarkdown documents or

Jupyter Notebooks, have been committed to GitHub, a code hosting platform (GitHub, 2019). Github allows for easy code sharing and collaboration between researchers, making it the ideal platform to aid with reproducibility.

# Chapter 2

# Ethics

## 2.1 Data Ethics Overview

The recent increases in data access offer a unique opportunity for companies and researchers to gain insights that would otherwise be impossible to determine. These insights could lead to better disease tracking, optimization of business practices, or streamlining something, to name a few examples. The benefits of using data in research across all disciplines are extensive and cannot be understated. However, ensuring that the data usage is ethical and fair is a vital step in the process.

As helpful as data can be, it can be equally as damaging and dangerous if used without consideration of the ethical consequences. As stated by Mittelstadt & Floridi (2016), researchers utilizing big data must have ethical foresight instead of ethical hindsight as the consequences are substantial. While important, many do not focus on ethics or issues of bias. Often, these issues stem from a lack of contextual understanding of data and data science techniques (**???**- need to get). Additionally, there is a lack of a universal code of conduct for data scientists. As data can be used by researchers in any domain, knowledge of ethical issues of statistics and data analysis is not widespread. We will begin by combining multiple sources to create a thorough ethics guide, specifically relating to research involving web scraping.

### 2.1.1 Theory-Driven Web Scraping

Landers, Brusso, Cavanaugh, & Collmus (2016) argue that when scraping data from the web, researchers must follow a hypothetico-deductive modeling approach. This means that data is scraped with a hypothesis in mind and is gathered in order to test the hypothesis or answer a research question. This directs researchers away from hypothesizing after the results are already known. When data is scraped without a specific research question and the study design is unbalanced, it may be more likely for issues of content or construct validity to occur. These may go undetected and demonstrate results that do not accurately represent the phenomenon. The concept of creating a hypothesis after data is collected and analyzed is known as *post-hoc hypothesizing*. It is important to differentiate between presenting post-hoc hypotheses as a priori (PPHA) and presenting post-hoc hypotheses as those which require future empirical verification (Leung, 2011). The latter represents an acceptable process of research, whereas the former is considered unethical in most research domains.

Leung (2011) specifies three types of widespread PPHA. In the first case, researchers create hypotheses directly from their results in order for their study to be theoretically compelling. In the second case, researchers will simply drop hypotheses that are dis-confirmed and fail to introduce them at all. In the third and final case, hypotheses may be added that appear to match the results but are presented a priori. These methods lack the transparency necessary for ethical research practices. This type of post-hoc hypothesizing fails to account for previously studied relevant theoretical concepts, an important aspect of the research process. These issues can be magnified when using data scraped from the web, as larger sample sizes could lead to more significant results. These studies may be more publishable and therefore, ethically questionable research may direct future work on the topic.

Additionally, investigators must recognize that results using scraped data cannot be generalized to the entire population, even if the sample appears widespread. The reason for this, specified by Wallace (2015), is that internet users are inherently different than those who are not on the internet. If the goal is to generalize toward other internet users, that is acceptable in certain cases. However, it is unethical to assume results can be generalized any further.

### 2.1.2 Identifiability

Another important ethical issue of web scraping that must be considered is identifiability. Data from online sources often contain identifying information, even if it does not initially appear so. Data that can be used to distinguish or trace identities either alone or in combination with other information that linkable to an individual is known as personally identifiable information (Krishnamurthy & Wills, 2009). Given the huge increases in social network use, scandals involving data from these sources have been all over the news. Facebook, in particular, has come under scrutiny for their lack of data privacy and transparency. In 2008, Facebook announced that an attack on their server resulted in the exposure of the personal data from over 50 million users (Rosen, 2018). While alarming, PII can be leaked without hacking systems. The ability for researchers to use APIs or other computer science techniques to scrape data from these sites has also resulted in an improper distribution of PII. As an example, researchers Emil Kirkegaard and Julius Daugbjerg Bjerrekær scraped data from OkCupid then released the data set for other researchers to use. However, the data included identifiable information of users, including their sexual preferences, politics, and feelings about homosexuality, among others (**???** to get).

When Kirkegaard was questioned about this ethical breach, he argued that the data were already public and therefore their scraped data was simply making the information accessible for research (**???** to get). This example brings light to the delicate balance of data accessibility and privacy. While reproducibility of research is important, avoiding ethical issues that arise when PII must be given equal attention. Often, determining that line is left up to the discretion of the researcher. Therefore, it is necessary for researchers to be constantly aware of the ethical consequences of gathering and sharing data. In the context of the OkCupid example and the argument from the primary investigator, while the data was technically public, the users did not consent for their data to be used and shared for research outside of the OkCupid website.

Sharing PII can result in consequences that extend beyond simply a lack of consent. (**???**) warn that it can lead to serious problems that range from group discrimination (racism, sexism, ageism, etc.) to group-targeted forms of violence. When evaluating the multiple ethical concerns of big data research, the solution is to follow the two moral duties introduced by (**???**): to foster human rights and improve human welfare. If that is the ultimate goal of the research, ethical concerns will be minimized.

### 2.1.3 Connection to this Research

Throughout the process of this research, we addressed and attempted to avoid any and all ethical concerns. First, we followed the theory-driven web scraping approach when connecting to the Twitter API to gather tweets. Our study was designed purposefully to answer our research question and test our hypothesis. We did not hypothesize post-hoc a priori nor did we generalize our results to populations past football fans who use Twitter.

While gathering tweets through the Twitter streaming and full-archive APIs, we only selected certain meta-data to be included in our final data sets. These did not include usernames, IDs, or other personally identifiable information. Our goal was to analyze sentiments from the words of the tweets and was not at all related to the specific users who produced said tweets. As our code used to gather the tweets is uploaded to GitHub, another researcher could alter the functions we wrote to include identifying variables into the data set. That being said, only a person with data science skills would be able to do so and we expect those using data science techniques to gather data will follow ethical practices.

# Chapter 3

# Study 1

## 3.1 Introduction

The goal of this initial study was to explore tweet data aimed at football players. As will be explained further in the next chapter, there are significant limits on gathering Twitter data that is part of their full archive (tweets that were published prior to seven days earlier). Therefore, we decided it was important to gather additional data that was not limited in order to perform preliminary analyses.

Twitter offers many methods for accessing its data. A few examples include a premium search of the full archive, a standard search, and a filter of real-time tweets. Each of these has specific limits, some more stringent than others. However, the method that allows for the largest amount of tweets is the real-time filtering method. Here, a query and time period are specified and a subset of tweets that match the query are gathered for the length of the time period.

The event that was used for this aspect of the project was Superbowl LIII. We believed that this event would offer an abundance of data, given the amounts of data amassed during previous Super Bowls. Last year, over 100 million people watched the Super

Bowl ((Statista, 2019)) and in 2017, 27.6 million tweets were posted relating to the Super Bowl (**???**).

We did not begin this project with any specific hypotheses or research questions as it was more exploratory in nature. Our main goals were to determine what the most tweeted words were, how many tweets were posted per player, and what the average sentiments were per player. Unlike in the next study, we were not looking to model the data.

## 3.2 Methods

### 3.2.1 Query

The first step of this study was to determine what our query would be. The standard option for filtering real-time tweets allows for up to 400 keywords, 5,000 user ids, and 25 0.1-360 degree location boxes (Twitter, 2019). As our focus was on players, we decided to use the roster of the starting players for each team as the query. The roster was pulled from the CBS Sports website, the network that hosted the game. Their full names were added to a column in a .csv file. Next, in order to increase the data that would be gathered, we decided to add their Twitter handles to the next column. The gathering of player names and Twitter handles was done by hand, and manually entered into the file.

Once in R, the name and Twitter handle columns were each made into lists then joined. This list was saved as an object to be used in the function that gathers live tweets.

### 3.2.2 rtweet Function

A package in R titled rtweet was designed to give access to Twitter's Rest and Streaming APIs (Kearney, 2018). To begin using the functions of this package, users must first become authorized by Twitter Inc., a process that can be done online. Once accepted as a developer, one must create an "app", which in turn, will then create tokens necessary to access the API through functions in the rtweet package.

Once created, a simple function called create_token connects to the app and saves your token to your environment. This means the following code only needs to be run once.

```
create_token(
  app = "my_twitter_research_app",
  consumer_key = "aaaaaaaa",
  consumer_secret = "bbbbbbbb",
  access_token = "cccccccc",
  access_secret = "dddddddd")
```

The next step was to decide the length of time that the function would run to collect tweets. Super Bowl LIII began at 6:30pm E.T. and was expected to last approximately four hours. We decided to run the function for seven hours beginning at 5:30. This would allow us to gather the tweets posted leading up to the game as well as those posted following the game. We believed reactionary tweets would be posted both during game play and in the hours following. The following code was used to load the query data, make a list that included full names and Twitter handles, and gather the streaming tweets.

```
#reading in the data
starters <- read.csv("path/starters.csv", stringsAsFactors = FALSE)
```

```
#pulling out player names
name <- starters$players

#cleaning twitter column, selecting that column, then filtering out those without twit
twitter_clean <- starters %>%
  mutate(twitter_clean = sub("'", "", twitter)) %>%
  select(twitter_clean) %>%
  filter(!twitter_clean == "")

#pulling out twitter handles
twitter <- twitter_clean$twitter_clean

#full name and twitter handle for streaming
full <- c(name, twitter)

# Stream keywords used to filter tweets
q <- paste(full, collapse=',' )

# stream time is in seconds
# ( x * 60 * 60 = x hours)
# Stream for 7 hours
streamtime <- 7 * 60 * 60

## Filename to save json data (backup)
filename <- "path/sb_tweets.json"
```

```
#save as json for later parsing
stream_tweets(q = q, timeout = streamtime, file_name = filename, parse = FALSE, la
```

As you can see from above, the lists of twitter handles and names needed to be formatted as a single string with a comma separating each value that tweets would be matched against. Additionally, the stream time must be in seconds so the numbers of hours chosen, in this case 7, needed to be multiplied by 3,600. Finally, in our stream_tweets function, we included the parameter "parse = FALSE" which saves the tweets as a .json file to my computer instead of loading the file directly to my environment. Later, I parsed this .json file using an additional function within the rtweet package.

```
#parsing entire file
rt <- parse_stream(filename)
```

However, after investigation, we realized that about half of the tweets appeared to be missing when the file was parsed. We knew this to be true because the .json file contained about 1.2 million lines and every other line was a tweet, meaning there should have been around 600,000 tweets in the data frame once the file was parsed. However, only about 300,000 tweets appeared. We realized that the file might have been too large to parse at once so we split our file into halves using the terminal then parsed each of those individually. The first half contained 307056 tweets and the second contained 311572, for a total of 618628 tweets. The two data frames each containing half of the tweets were saved as .rda files for easier loading in the future.

```
#first half
filename2 <- "/Users/juliannaalvord/Documents/nfl sentiment/sb analysis/sb_tweets_
```

```
#parse those tweets from above

rt <- parse_stream(filename2)


#second half

filename3 <- "/Users/juliannaalvord/Documents/nfl sentiment/sb_tweets_half2.json"


#parse tweets from above

rt2 <- parse_stream(filename3)


#saving these parsed files (each half individually)

save(rt2, file = "/Users/juliannaalvord/Documents/nfl sentiment/sb_tweets_half1.rda")


save(rt3, file = "/Users/juliannaalvord/Documents/nfl sentiment/sb_tweets_half2.rda")
```

### 3.2.3 Cleaning

Once these files were parsed, the files were loaded into a new file for cleaning. Once here, the function bind_rows from the dplyr package Wickham, François, Henry, & Müller (2019) was used to stack the two halves and create one tall data frame with 618628 rows and 88 variables. These 88 variables each contain a piece of metadata that is provided by Twitter. Some variables include "user_id", "created_at", and "is_retweet". The next step was to clean the text in order to pull out the names or twitter handles contained in the tweets in order to determine which players the tweet is mentioning.

Two variables contain text that can be analyzed. One is the column "text", which contains the actual UTF-8 of the status update. However, some tweets in the data were quoted, meaning users added comments to an already published tweet. This text was found in the variable "quoted_text". Using the tolower function from the base package R Core Team (2018), the text from these two columns was translated from a mix of upper and lower case characters to only lower case. Then, using the paste function from the base package, another column was created that combined the text from these two lower case character vectors. Once this column was created, we were able to determine if there were duplicate tweets by using the duplicated function again from the base package. In total, 398,231 tweets contained duplicate text. We believed that the majority of those would have been retweeted tweets. Using the grepl function from the base package, we could determine if the text contained the string "rt", indicating that the tweet may be a retweeted text. However, only 86,016 of the tweets contained this string. We decided to leave the duplicate indicator variable in the data frame but did not make a decision about how to handle them at this stage. Instead, we saved this cleaned data frame as another rda to be used for simple natural language processing and sentiment analyses.

```r
#binding rows of the two halves
full <- half1 %>%
  bind_rows(half2)

#flagging duplicates from text+quoted text
full <- full %>%
  mutate(text_low = tolower(text),
         quoted_text_low = tolower(quoted_text),
         full_text_low = paste(text_low, quoted_text_low, sep = ","),
         dup = ifelse(duplicated(full_text_low), 1, 0))

#how many dups?
table(full$dup, useNA = "a")
```

```
#how many of the dups have rt at all
n_dup_rt <- full %>%
  filter(dup == 1) %>%
  mutate(rt = ifelse(grepl("rt", full_text_low), 1, 0)) %>%
  group_by(rt) %>%
  summarise(n = n())

#how many contain "rt"?
table(n_dup_rt$rt, n_dup_rt$n)

#saving file as .rda file
save(full, file = "/Users/juliannaalvord/Documents/nfl sentiment/sb_tweets_full.rda")
```

## 3.2.4 Matching Tweets to Players

Now that we had a data set containing all tweets and a cleaned text column, we sought to determine which tweets mentioned which players in order to properly analyze the data. We started by creating the same list of names and twitter handles that was used as the query within our stream_tweets function. However, this time, the list was made into a single string with the vertical bar separating each element. Then, we used the same tolower function as above to change each character to lower case, in order to properly match the cleaned text variable.

```
#pulling out the names
name <- starters$players

#cleaning twitter column, selecting that column, then filtering out those without twit
```

```
twitter_clean <- starters %>%
  mutate(twitter_clean = sub("'", "", twitter)) %>%
  select(twitter_clean) %>%
  filter(!twitter_clean == "")

#list of twitter handles
twitter <- twitter_clean$twitter_clean

#full name and twitter handle for streaming
full_name <- c(name, twitter)

#making list for str_extract_all function
all_players <- paste(full_name, collapse='|')

#lower case names and twitter handles
all_players_low = tolower(all_players)
```

The function str_extract_all from the stringr package Wickham (2019) was used to test whether the text column contains any of the name or Twitter handles. A list column was created because many of the tweets matched multiple names or twitter handles (when multiple players were mentioned in one tweet). Then, to create a data frame which duplicates the tweet for each player mentioned, the unnest function from the tidyr Wickham & Henry (2018) was used. This function takes a list-column then makes each element of the list its own row.

```
#lower casing text and quoted text to be able to search without missing any play
full_more <- full %>%
 #pulling out the players from either text or quoted text
 mutate(name_text = str_extract_all(full_text_low, pattern = all_players_low))

#unnesting the name_text list column
full_more2 <- full_more %>%
  unnest(name_text)
```

Once complete, it was necessary to filter the data set to include only tweets that mentioned a player by their handle. To do so, I used the grepl function to determine if the new column that pulled out the name or handles from the tweet began with the "@" sign. Then, the other tweets were saved into a different data frame. Each was subsequently joined with the initial starter data set. Finally, these two data sets were row bound and two new columns were created. One filled in the twitter handles for the tweets that mentioned a name and the other filled in the names for the tweets that mentioned a handle.

```
#lowering twitter handles and player names for join
starters2 <- starters %>%
  mutate(twitter_clean = sub("'", "", twitter),
         twitter_clean2 = tolower(twitter_clean),
         name_clean = tolower(players)) %>%
  select(-c(players, twitter, twitter_clean))

#filtering for tweets that mention a player by their @
tweets_names <- full_more2 %>%
  filter(!grepl("@", name_text))

#filtering for tweets that mention a player by their full name
tweets_handles <- full_more2 %>%
  filter(grepl("@", name_text))


#tweets with names join

tweets_names2 <- tweets_names %>%

  left_join(starters2, by = c("name_text" = "name_clean"))


#tweets with handles join

tweets_handles2 <- tweets_handles %>%

  left_join(starters2, by = c("name_text" = "twitter_clean2"))


#row binding those two

tweets_final <- tweets_handles2 %>%

  bind_rows(tweets_names2) %>%

  #next code creates final name and twitter columns by filling in with name_text

        #in tweets with names df, left join gets rid of "name_clean" col

  mutate(name_clean_final = ifelse(is.na(name_clean), name_text, name_clean),

        #in tweets with handles df, left join gets rid of "twitter_clean2" col

        twitter_clean_final = ifelse(is.na(twitter_clean2), name_text, twitter_cl
```

### 3.2.5 Natural Language Processing

The cleaned data set from above was loaded into the environment for both natural language processing and sentiment analysis. For both analyses, it was necessary to rearrange the data by following the steps detailed in the book titled Text Mining in R: A Tidy Approach (Silge & Robinson, 2018). Here, the authors define the tidy text format as being “a table with one-token-per-row”. In this case, the token is an individual word. In other cases, a token could be any meaningful section of text, such as a sentence, phrase, or paragraph. The next analyses were made possible by the package tidytext package (**???**).

THERE WILL BE AN IMAGE HERE but I'm getting an error

To begin, a data frame containing only the words of the tweets was created. This was done by first selecting two columns: the status id, which are unique identifiers for each tweet, and the full text column. From there, the unnest_token function from the tidytext package was used to split the text column into words and create a row for each word of each tweet.

```
words <- tweets_final %>%
    select(status_id, full_text_low) %>%
    unnest_tokens(word,full_text_low)
```

For both of the analyses, only certain words are of interest, especially when adding sentiment or determining which words were the most common. Other words, such as "and", "is", and "the", are known as stop words and were filtered out. A stop words lexicon can be accessed through the tidytext package by using the function get_stopwords. Once the stop words data frame is loaded into the environment, we added additional rows for "words" that are common to tweets but are unnecessary for the environment. After the words and stop words data sets were created, the stop words data set was anti_joined to the words data set to filter those words out. From there, simple natural language processing such as determining most common words were possible.

```
#specifying stop words
my_stop_words <- stop_words %>%
    select(-lexicon) %>%
    bind_rows(data.frame(word = c("https", "t.co", "rt", "amp","4yig9gzh5t","fyy2ceydhi"
#removing stop words
tweet_words <- words %>%
    anti_join(my_stop_words)
```

An additional step was necessary to add sentiments to these words. The tidytext package includes three lexicons containing words and their corresponding sentiments. The three include AFINN from (**???**), bing from (**???**), and nrc from (**???**). The first assigns words with a number from -5 to 5, with 5 being the most positive words and -5 being the most negative words. The second simply assigns words as positive or negative. The third assigns words in a binary fashion into the categories of positive, negative, anger, anticipation, disgust, fear, joy, sadness, surprise, and trust. For our purposes, we used the bing lexicon. For simple counts of sentiment across the whole tweets data set, we simply joined the bing lexicon to the words data set.

```
#getting sentiments- using bing
  bing_lex <- get_sentiments("bing")


#joining words with the sentiments
full_sentiments <- tweet_words %>%
  left_join(bing_lex)
```

### 3.2.6 Sentiment Analysis by Player

Given our interest in the differences in sentiments for players depending on demographic information, we needed to change the format of our data in order to determine the sentiment for each player individually. We ran a loop that ranged from 1-50, as there were 50 starting players in our Superbowl sample. For each player, the larger tweet data set was filtered for the player who's name matched the index of the loop. Then, the same process as above of unnesting the tweets by words, filtering out stop words, then joining to the sentiment lexicon was employed. Lastly, we wanted the data to be formatted in a data frame containing a single row with three columns. One column was the name of the player and the two others were the counts of negative and positive words. This data frame is saved into a list. Outside of the loop, we used the do.call function to execute the rbind function of the 50 data sets. This resulted in a data set of 50 rows and three columns that contained the counts of positive and negative words for each player.

```
#list of names for loop
names <- as.list(starters2$name_clean)

#empty list to add sentiments for each player
datalist = list()

for(i in 1:50) {

  #filter for each person
  tweets <- tweets_final %>%
    filter(name_clean_final == names[i])

  #pick out words
  words <- tweets %>%
    select(status_id, full_text_low) %>%
    unnest_tokens(word,full_text_low)

  #creating df of stop words
```

```
my_stop_words <- stop_words %>%
  select(-lexicon) %>%
  bind_rows(data.frame(word = c("https", "t.co", "rt", "amp","4yig9gzh5t","fyy2c

#anti-join with stop words to filter those words out
tweet_words <- words %>%
  anti_join(my_stop_words)

#getting sentiments
bing_lex <- get_sentiments("bing")

#joining sentiments with non-stop words from tweets
fn_sentiment <- tweet_words %>%
  left_join(bing_lex)

#creating df with n of sentiments
df <- fn_sentiment %>%
  filter(!is.na(sentiment)) %>%
  group_by(sentiment) %>%
  summarise(n=n())

#making df of sentiments for each person
df_2 <- df %>%
mutate(player = names[i]) %>%
spread(key = sentiment, value = n)

datalist[[i]] <- df_2
}


#sentiments n for all players
sentiments_full = do.call(rbind, datalist)
```

This data set titled sentiments_full was then joined to the initial starters data set to match the sentiment counts to other demographic data. We created other variables including the total sentiment words (by adding the negative and positive counts), the percent of negative words (by dividing the negative count column by the total column), and the percent of positive words (by dividing the positive count column by the total column).

```
#joining and creating percentages
starters_sentiment <- starters %>%
  left_join(sentiments_full, by = c("name_clean" = "player")) %>%
  mutate(totalsentiment = positive+negative,
         neg_perc = negative/totalsentiment * 100,
         pos_perc = positive/totalsentiment *100)
```

From here, we were able to create visualizations comparing average negative and positive percentages across different groups, including race, team, and position.

## 3.3 Results

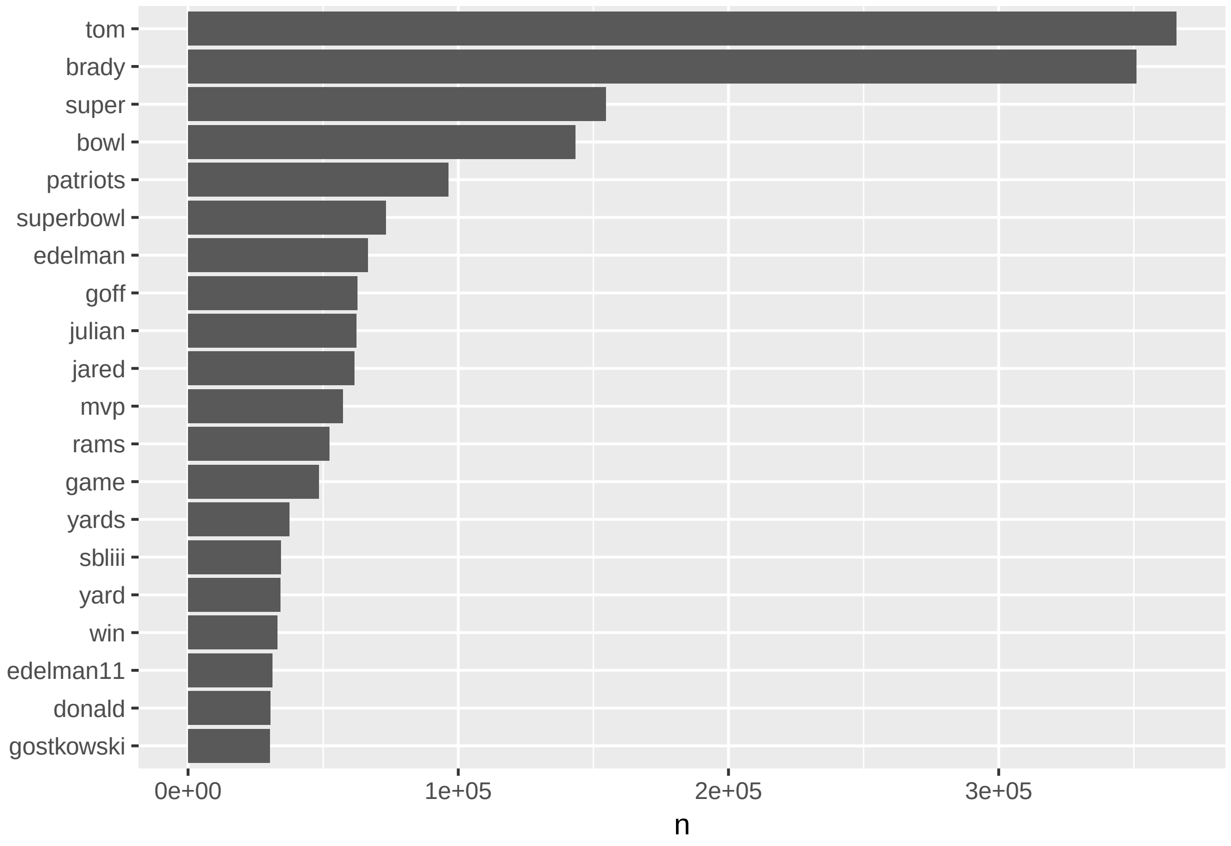
### 3.3.1 What were the most popular words?

In total, 652411 tweets were gathered for the 50 starting players during the 7-hour specified period.

The 20 words that were used the most often (after removing stop words) are shown below.

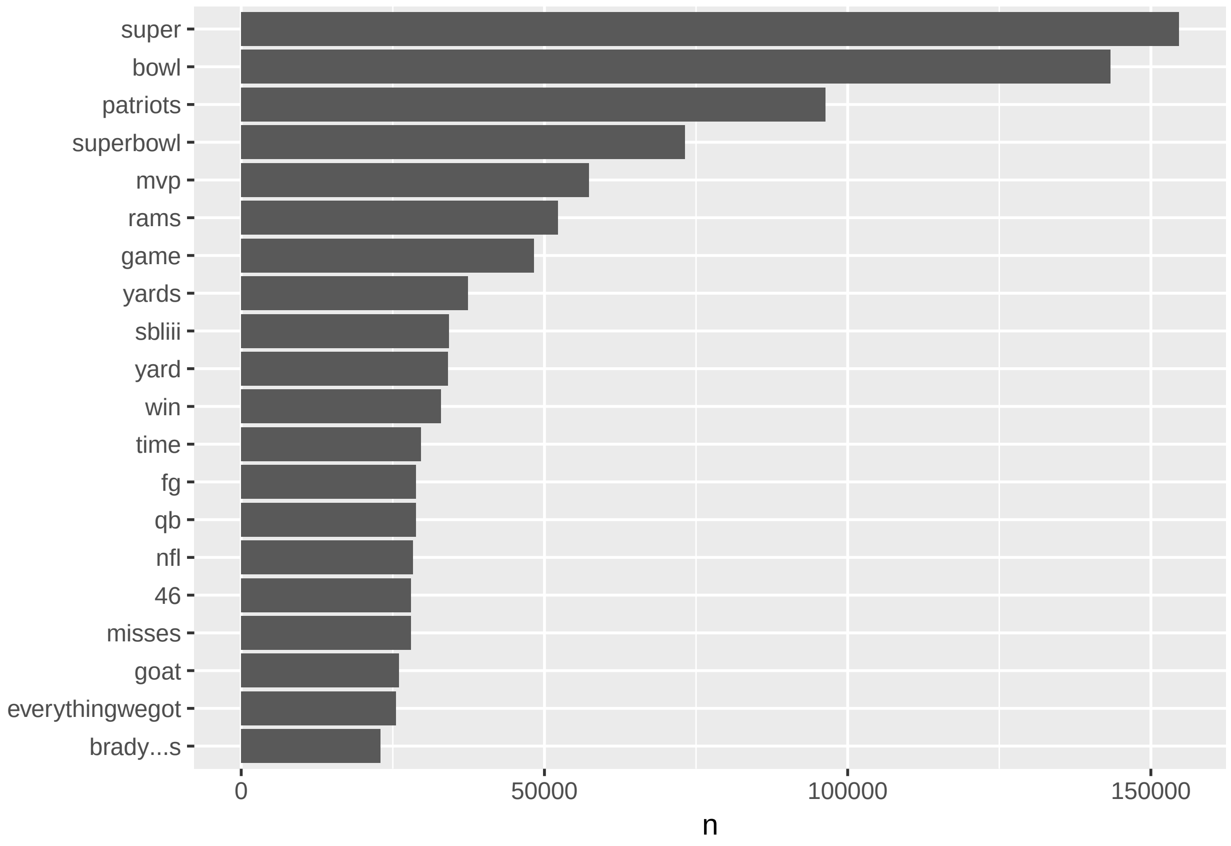
```
#word counts
word_counts <- tweet_words %>%
  count(word, sort = TRUE)

#viz
ggplot(word_counts %>% head(n = 20L) %>% mutate(word = reorder(word, n)), aes(wor
  geom_col() +
  xlab(NULL) +
  coord_flip()
```



As seen from the above visualization, the top four most popular words are “tom”, “brady”, “super”, and “bowl”. Of the 20 top words, 9 were names of players. We decided to determine the 20 top words without names by adding the list of full names and twitter handles to our stop words data set. Those words are below.

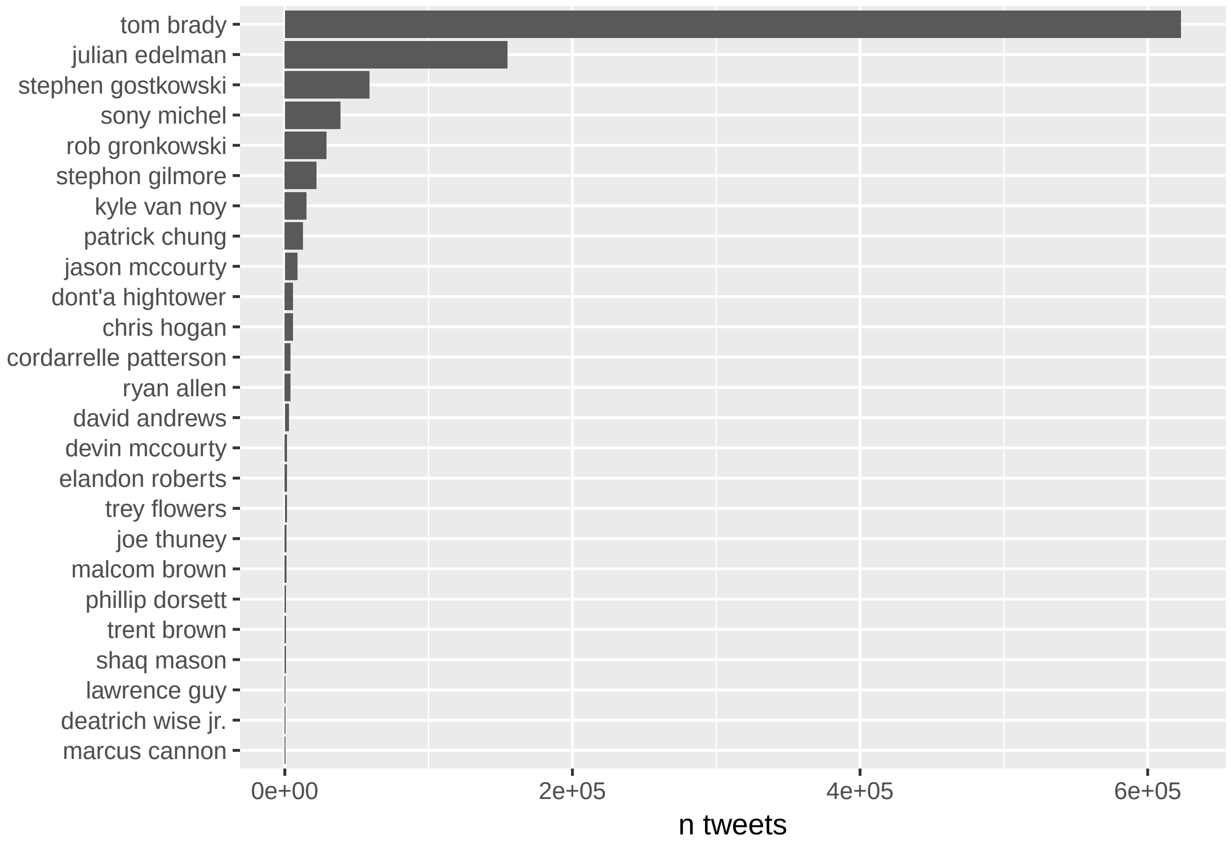
```
ggplot(word_counts %>% head(n = 20L) %>% mutate(word = reorder(word, n)), aes(word, n))
  geom_col() +
  xlab(NULL) +
  coord_flip()
```



Here, the top four most popular words are “super”, “bowl”, “patriots”, and “superbowl”.

### 3.3.2 Who had the most tweets from the Patriots?

```
#number of tweets for each player on NE
ggplot(starters_sent_n %>%
        filter(team == "NE") %>%
        mutate(name_clean = reorder(name_clean, n)), aes(name_clean, n)) +
  geom_col() +
  xlab(NULL) +
  ylab("n tweets") +
  coord_flip()
```

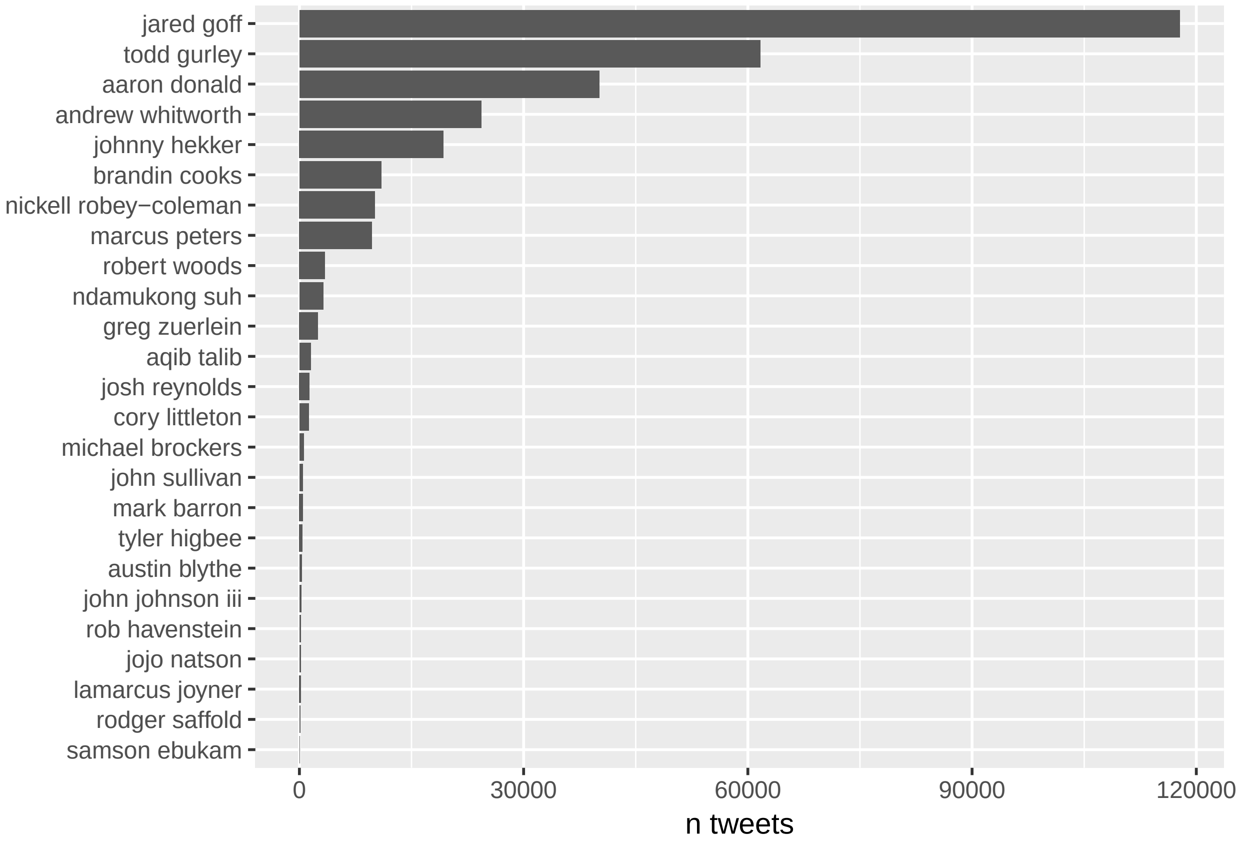


The players with the most tweets are Tom Brady, Julian Edelman, Stephen Gostkowski, Sony Michel, and Rob Gronkowski. Quite a few players had so few tweets compared to the top few players that it cannot be determined from this visualization the exact number of tweets each player had. Below is a table that includes each player, their position, race, and number of tweets.

| name_clean | Position | off_or_def | Race | n_tweets |
|---|---|---|---|---|
| tom brady | QB | O | white | 311496 |
| julian edelman | WR | O | white | 77413 |
| stephen gostkowski | K | S | white | 29453 |
| sony michel | RB | O | black | 19428 |
| rob gronkowski | TE | O | white | 14361 |
| stephon gilmore | CB | D | black | 11023 |
| kyle van noy | OLB | D | black | 7398 |
| patrick chung | SS | D | black | 6349 |
| jason mccourty | CB | D | black | 4495 |
| dont'a hightower | OLB | D | black | 2856 |
| chris hogan | WR | O | white | 2835 |
| cordarrelle patterson | KR | S | black | 2003 |
| ryan allen | P | S | white | 1921 |
| david andrews | C | O | white | 1390 |
| devin mccourty | FS | D | black | 777 |
| elandon roberts | MLB | D | black | 716 |
| trey flowers | LE | D | black | 688 |
| joe thuney | LG | O | white | 553 |
| malcom brown | DT | D | black | 436 |
| phillip dorsett | WR | O | black | 364 |
| shaq mason | RG | O | black | 350 |
| trent brown | LT | O | black | 350 |
| lawrence guy | DT | D | black | 215 |
| deatrich wise jr. | RE | D | black | 200 |
| marcus cannon | RT | O | black | 162 |

### 3.3.3 Who had the most tweets from the Rams?

```r
#number of tweets for each player on LA
ggplot(starters_sent_n %>%
         filter(team == "LA") %>%
         mutate(name_clean = reorder(name_clean, n)), aes(name_clean, n)) +
  geom_col() +
  xlab(NULL) +
  ylab("n tweets") +
  coord_flip()
```



The players with the most tweets are Jared Goff, Todd Gurley, Aaron Donald, Andrew Witworth, and Johnny Hekker. Again, the number of tweets for many players cannot be determined based on this visualization. Again, below is a table of each players demographic information and number of tweets.

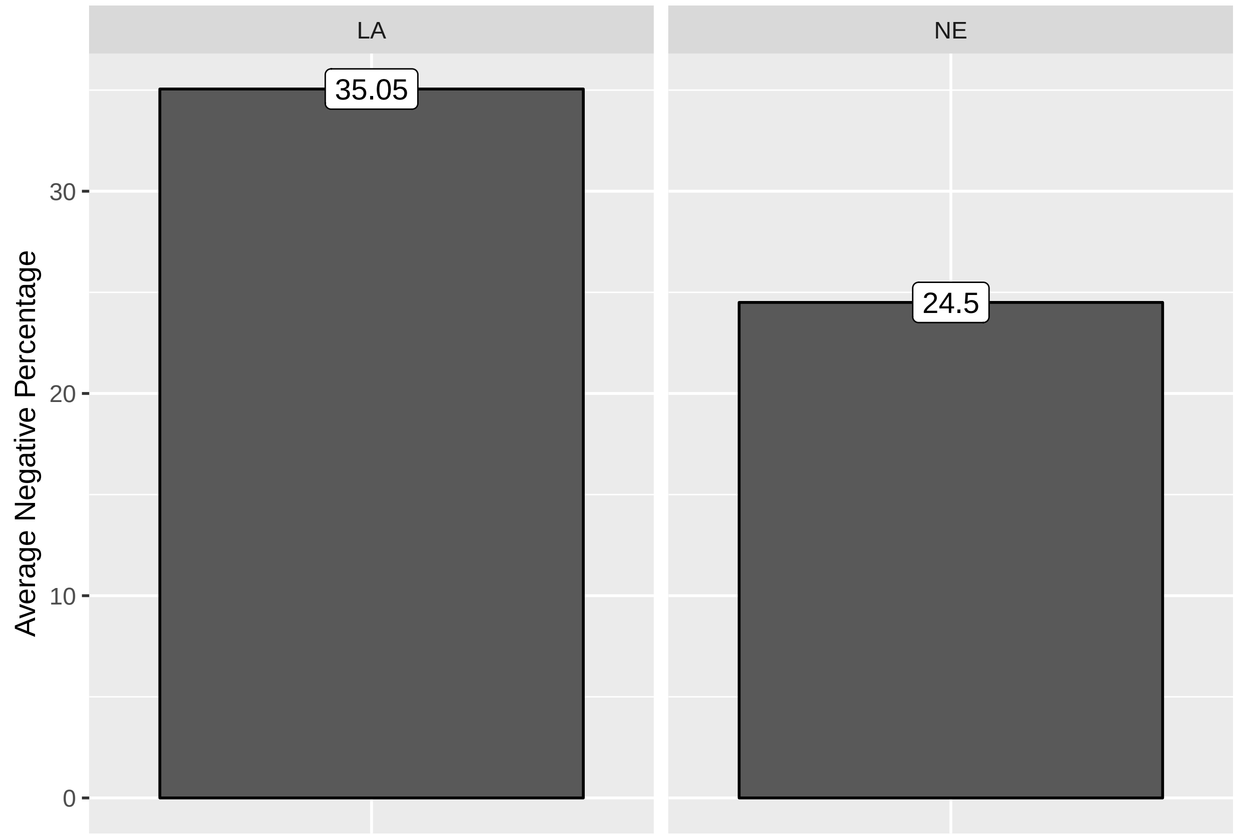
| name_clean | Position | off_or_def | Race | n_tweets |
|---|---|---|---|---|
| jared goff | QB | O | white | 58900 |
| todd gurley | RB | O | black | 30833 |
| aaron donald | RE | D | black | 20068 |
| andrew whitworth | LT | O | white | 12171 |
| johnny hekker | P | S | white | 9638 |
| brandin cooks | WR | O | black | 5481 |
| nickell robey-coleman | CB | D | black | 5039 |
| marcus peters | CB | D | black | 4838 |
| robert woods | WR | O | black | 1691 |
| ndamukong suh | NT | D | black | 1594 |
| greg zuerlein | K | S | white | 1227 |
| aqib talib | CB | D | black | 769 |
| josh reynolds | WR | O | black | 673 |
| cory littleton | ILB | D | black | 632 |
| michael brockers | LE | D | black | 306 |
| john sullivan | C | O | white | 239 |
| mark barron | ILB | D | black | 212 |
| tyler higbee | TE | O | white | 204 |
| austin blythe | RG | O | white | 165 |
| john johnson iii | SS | D | black | 121 |
| rob havenstein | RT | O | white | 106 |
| jojo natson | KR | S | black | 102 |
| lamarcus joyner | FS | D | black | 80 |
| rodger saffold | LG | O | black | 63 |
| samson ebukam | OLB | D | black | 27 |

A few observations from the two tables are that the player with the most amount of tweets on both teams is the quarterback and a majority of the players in the top 5 are offensive players.

### 3.3.4 Which team has a higher negative sentiment percentage?

Though no formal hypotheses were made in this study, we expect that the team that loses will see a higher average negative sentiment percentage. Therefore, we expected the Rams, who lost, to have a higher average negative sentiment compared to the Patriots. This was tested directionally and no models were fit.

```
#sentiments by team
ggplot(starter_sent_2, aes(x = sentiment, y = mean_perc_sent)) +
  geom_bar(stat = "identity", position = "dodge", color = "black") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) + facet_wrap(~team, nc
  ylab("Average Negative Percentage") +
  geom_label(aes(label = round(mean_perc_sent, 2))) +
  theme(axis.text.x = element_blank(), axis.ticks.x = element_blank(), axis.title
```



Luckily, our data matched this expectation. The average negative sentiment percentage for the Rams was 35.05% while the average negative sentiment for the Patriots was 24.5%.

### 3.3.5 Which racial group had a higher average negative sentiment percentage?

We wanted to test to see if there appeared to be directional differences in the mean negative sentiments for black players and white players. This was tested across all players then within teams.

```r
#sentiments by race
ggplot(starter_sent_2, aes(x = sentiment, y = mean_perc_sent, fill = Race)) +
  geom_bar(stat = "identity", position = "dodge", color = "black") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  scale_fill_manual(values=c("black", "white")) +
  ylab("Average Negative Percentage") +
  theme(axis.text.x = element_blank(), axis.ticks.x = element_blank(), axis.title
```

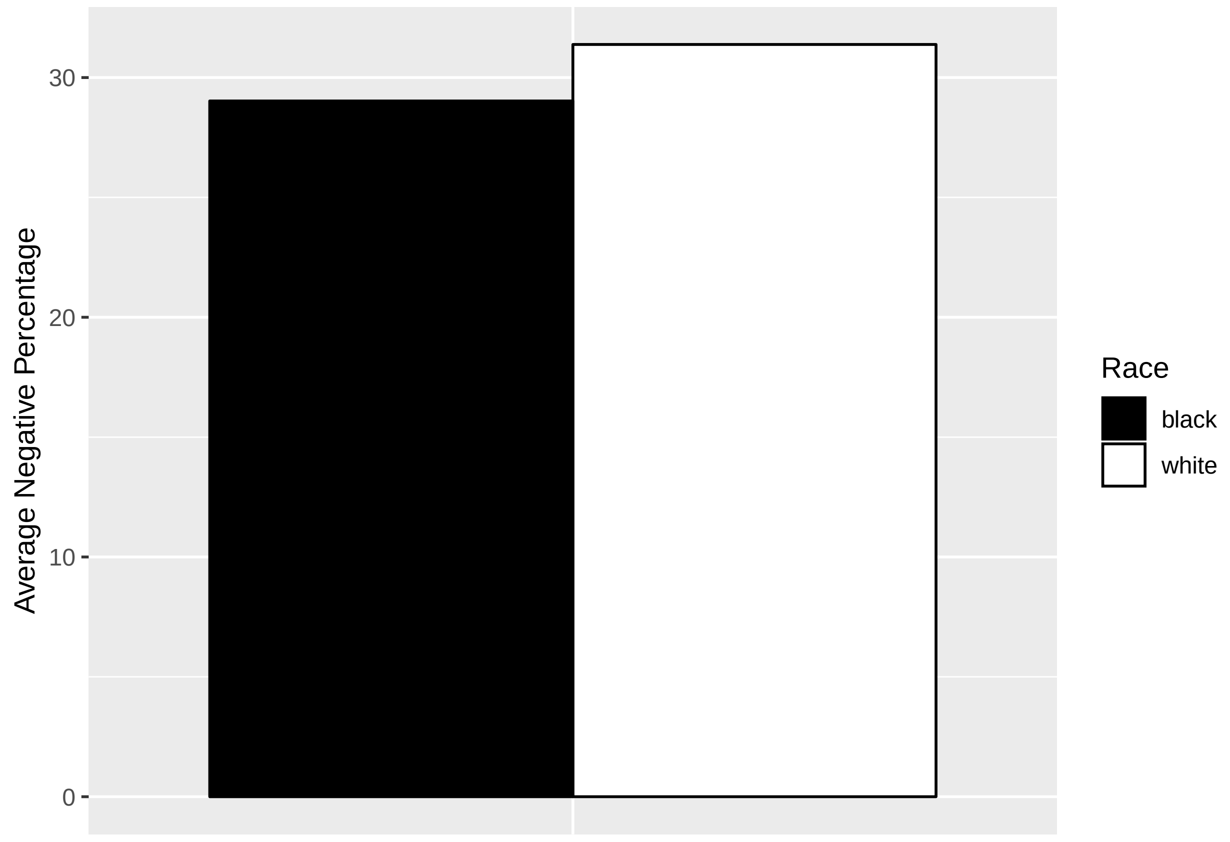
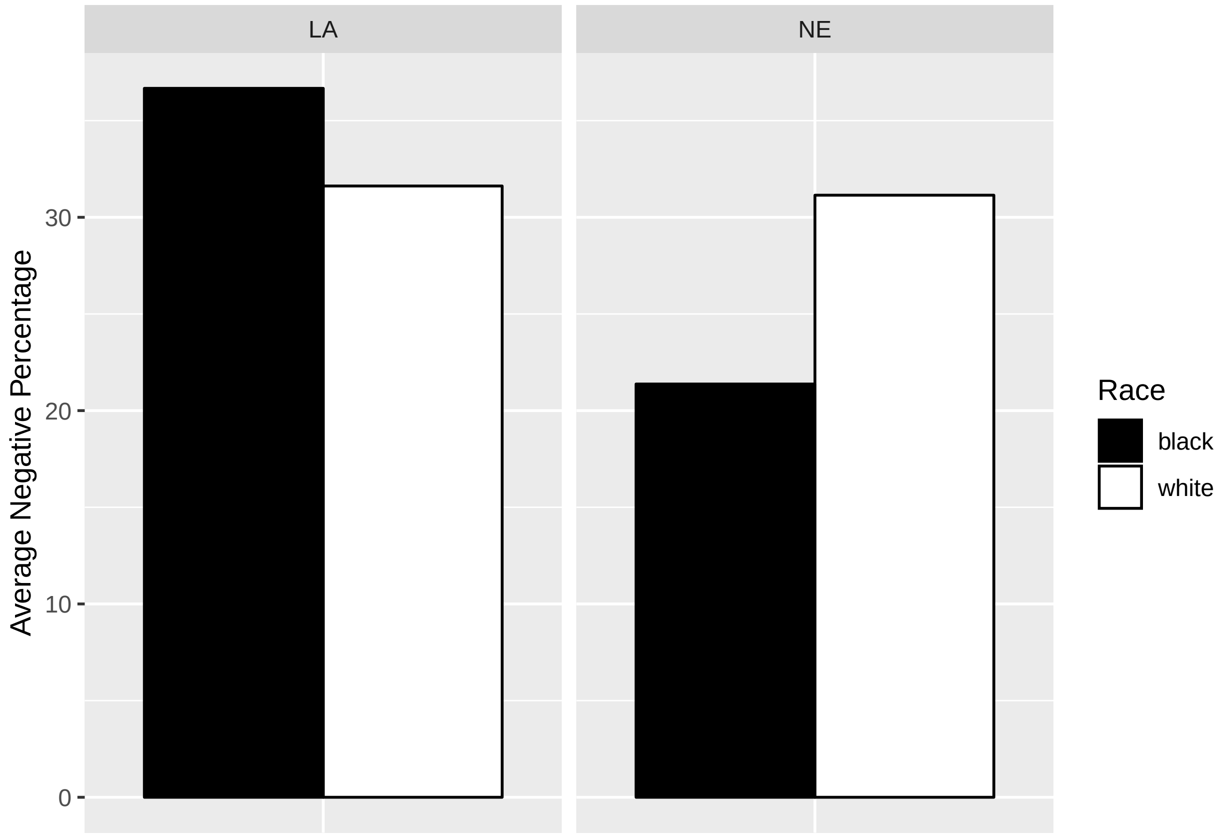


Figure 3.1: Average Negative Sentiment Percentage by Race

From this graph, we can see that white players have higher average negative sentiment percentages across both teams.

```
#sentiments by race and team
ggplot(starter_sent_2, aes(x = sentiment, y = mean_perc_sent, fill = Race)) +
  geom_bar(stat = "identity", position = "dodge", color = "black") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  scale_fill_manual(values=c("black", "white")) + facet_wrap(~team, ncol = 2) +
  ylab("Average Negative Percentage") +
  theme(axis.text.x = element_blank(), axis.ticks.x = element_blank(), axis.title.x = e
```



When split by team, we see a higher average negative sentiment percentage for black players on the Rams and for white players on the Patriots.

## 3.4 Conclusions and Moving Forward

A few basic conclusions and observations were made regarding the results of our first study. First, in regards to the top words, many did not have sentiments attached to them because they are specific to football. For example, one of the words in the top 20 is “goat”. This is a word often used when describing Tom Brady. Technically, it is an acronym and stands for “greatest of all time”. This common word is quite positive however it is not included in the sentiment lexicon. Other words that are specific to football that need to be added to the lexicon as positive words when in a football context are “history”, “ring”, “rings”, “dynasty”, “clutch”, “congrats”, and “g.o.a.t”.

Next, based on the number of tweets for the 50 players, we realized that the quarterbacks and receivers (running backs, wide receivers, and tight ends) were popular on Twitter. Almost all of the players whose positions match those two categories were near the top for their team in terms of number of tweets. This conclusion directed the decisions we made when choosing the subset for our next study.

Finally, from our visualizations from above as well as a contextual knowledge of the game allowed us to confirm that our data was not random. For one example, Stephen Gostkowski, a kicker for the Patriots, had an extraordinarily high negative sentiment percentage at 96.55%. This is mostly unsurprising, as he missed an early field goal that would have given New England their first points of the game. On the opposite side, Julian Edelman, a wide receiver for the Patriots, was named the MVP of the game and his positive sentiment percentage matched this at 90.39%. These examples, along with the differences depending on outcome, gave us confidence that Twitter data was appropriate to use to model sentiment percentage differences between racial groups and depending on outcomes.

# Chapter 4

# Study 2

## 4.1 Introduction

After gathering the analyzing tweets posted during the Superbowl, we wanted to create a balance study that would allow us to build models to determine if there are significant relationships between race, outcome, and percentage of negative tweets. As previously stated, our research question was: Are sentiments on Twitter more negative towards black NFL players during a failed outcome (a loss) and more positive towards white NFL players during a successful outcome (a win)? In order to add outcome to this analysis, we decided to include multiple games as separate time points.

We hypothesized that during a game that was lost, sentiments would be more negative toward black players and during a game that was won, sentiments would be more positive toward white players.

## 4.2 Methods

### 4.2.1 Full-Archive Twitter API

In order to gather tweets through Twitter's APIs, three options are available: the standard, 30-day archive, and the full-archive searches. Within the advanced search methods (30-day and full archive), there is a free "sandbox" option to allow researchers to test applications or other functions. The details of these can be found in chapter 1. The chosen method for this project was the premium full-archive search given our desire to focus on games from 2017. This method has strict limits in terms of the number of tweets that can be scraped per month. Below is an image that specifies these limits.

THERE WILL BE AN IMAGE HERE

In addition to these limits, there are additional monthly limits, depending on how much is spent. We chose a 500/month request limit for this project. Therefore, with 500 requests per month and 500 tweets per request, we were capped at 250,000 tweets total. This restraint determined and influenced many decisions of this study, especially in regard to the choice of players to include in our sample.

### 4.2.2 Chosen Games and Sample

Initially, when beginning this project, we hoped to use individual plays of a game as the measurement level. We hoped to use play-by-play data that was gathered earlier and match the time of plays to the time of tweets. Included in this dataset were plays from the first six games of the 2017 season. However, due to discrepencies in the time variables and time restraints, this was not possible. Regardless, we chosen to continue to focus on those specific games. First, we believed that six games gave us enough time points while still giving us the ability to gather enough tweets to measure sentiments accurately. Moreover, in 2017, US President Donald Trump attacked NFL players who had chosen to kneel in support of Colin Kaepernick on multiple platforms. He went so far as to say “get that son of a bitch off the field”, referring to any black player that decided to kneel (Graham, 2017). We predicted that the increased racial tension caused by the president might be measurable in this study.

As seen from the previous study, some players were barely mentioned on Twitter, even during the most watched football game of the year. However, we did notice that players from two positions in particular were mentioned a lot: quarterbacks and receivers. To ensure that we could gather enough tweets to measure sentiments over a single game, we decided to limit our sample to only quarterbacks and receivers as they seemed to be the most popular positions on Twitter.

In order to get closer to making causal connections between sentiment and race, I needed to ensure that there was an equal number of white and black quarterbacks and receivers. As I had already decided to focus in on 2017, I began my subset by researching which teams had starting quarterbacks who were black. The total for 2017 was 8 out of 32 teams. Then, I found that four of these eight teams also had a top receiver who was white and a top receiver who was black. These four black quarterbacks and the eight corresponding receivers (one white and one black from each team) made up the first half of the subset. To create a balanced design, four white quarterbacks were chosen by determining who was the closest in total yards for the 2017 season for each black quarterback. After ensuring that these quarterbacks’ teams also had a top receiver who was black and a top receiver who was white, those 12 additional players were added to the subset. In total, tweets were gathered for the 24 chosen players. Below is a table of the players, their race, position, corresponding team.

| Name | Team | Position | Race |
|---|---|---|---|
| Joe Flacco | Baltimore Ravens | QB | W |
| Nick Boyle | Baltimore Ravens | R | W |
| Mike Wallace | Baltimore Ravens | R | B |
| Cam Newton | Carolina Panthers | QB | B |
| Devin Funchess | Carolina Panthers | R | B |
| Christian McCaffrey | Carolina Panthers | R | W |
| Andy Dalton | Cincinnati Bengals | QB | W |
| Tyler Kroft | Cincinnati Bengals | R | W |
| A.J. Green | Cincinnati Bengals | R | B |
| Dak Prescott | Dallas Cowboys | QB | B |
| Dez Bryant | Dallas Cowboys | R | B |
| Jason Witten | Dallas Cowboys | R | W |
| Alex Smith | Kansas City Chiefs | QB | W |
| Travis Kelce | Kansas City Chiefs | R | W |
| Tyreek Hill | Kansas City Chiefs | R | B |
| Carson Wentz | Philadelphia Eagles | QB | W |
| Zach Ertz | Philadelphia Eagles | R | W |
| Alshon Jeffrey | Philadelphia Eagles | R | B |
| Russell Wilson | Seattle Seahawks | QB | B |
| Doug Baldwin | Seattle Seahawks | R | B |
| Jimmy Graham | Seattle Seahawks | R | W |
| Jameis Winston | Tampa Bay Buccaneers | QB | B |
| Mike Evans | Tampa Bay Buccaneers | R | B |
| Adam Humphries | Tampa Bay Buccaneers | R | W |

Further investigation revealed that Tampa Bay had a bye during week 1 of the 2017 season and Seattle, Dallas, and Cincinnati had byes during week 6 of the 2017 season. Therefore, if we gathered tweets for all 6 games, we would have missing values. Instead, we decided to choose 5 time points for each team. For the three teams with a bye during week 6, games 1-5 were chosen and for the other 5 teams, games 2-6 were chosen. In total, we would gather tweets for 24 players across 5 games. This totals to 120 different observations. Since we were limited to 250,000 tweets, approximately 2083 tweets could be gathered per player per game.

These 24 players were added to a dataset. Also included were variables corresponding to their twitter handle, race, position, and starting and ending times for each 5 games. It was important to us that the tweets were gathered during game play in order to limit extraneous factors that could affect sentiments toward a player. Given that the average NFL game time in 2016 was just above 3 hours and 8 minutes Schalter (2017), we decided to limit the time period to 3 hours and 15 minutes after the start of the game.

### 4.2.3 Searchtweets and Gathering Tweets using Python

In study 1, we used a function from the rtweet package in R. However, there is no package in R that supports the full-archive search method. Instead, we used a library in Python titled searchtweets. Like in study 1, we were required to setup and load authentication, first online and then using a function called load_credentials.

```
premium_search_args = load_credentials("~/.twitter_keys.yaml",
                                        yaml_key="search_tweets_api",
                                        env_overwrite=False)
```

Then, we loaded our data into the environment using a function called read_csv from the pandas library.

```
data = pandas.read_csv("~PATH/final_subset.csv")
```

Given that we had many variables which needed to change each time we ran the function to gather tweets, we decided to create three functions to simplify the process. The first takes a list of team names and filters the data for players on the designated teams.

```
def get_data(teams):
    data_1 = data[data.Team.isin(teams)]


    return data_1
```

The second and third functions are more complicated. The two are almost exactly the same except one allows for searching twitter for the players' first names and the other searches for their twitter handles. The arguments of these functions include a data set as well as start and end time identifiers. From there, the full names within the player name column is made into a list. Then, to grab the start and end times, the columns that match the start and end arguments are selected and saved as objects. Once these lists have been created, the function gathers tweets using two functions from the searchtweets library: gen_rule_payload and ResultStream. The first takes a query, start time, end time, and maximum results per request specification. Our function loops through each name or twitter handle and enters it as the query. Then the start and end objects are used as the start and end times. The maximum tweets per request, 500, was kept constant. Added to the query is a string "-is:retweet", which limits the tweets to exclude those which are retweeted. The rule that was specified using the previous function is then added for the rule_payload argument of the ResultStream function. This second function also requires the maximum number of results and pages to search through to be specified, which we kept constant at 2000 and 4, respectively. The final argument is our authentication object.

Below is the tweet-gathering loop from one of our functions. Both entire functions can be found in the data appendix.

```python
#running loop for tweets


    all_tweets = []


    for handle in newtwitter:


        rule = gen_rule_payload(handle + " -is:retweet",
                                from_date = start,
                                to_date = end,
                                results_per_call = 500)


        rs = ResultStream(rule_payload=rule,
                          max_results=2000,
                          max_pages=4,
                          **premium_search_args)


        tweets2 = list(rs.stream())


        [print(tweet.all_text) for tweet in tweets2[0:10]];


        all_tweets.extend(tweets2)


        time.sleep(10)
```

The next part of the function selects certain aspects of the twitter metadata and adds it to columns in a data frame. This data frame that includes the tweet, length, tweet id, date, number of likes, numbers of retweets… as columns is returned at the end of our function.

For each set of teams with the same starting and ending times each week, these two functions are run. Below is an example.

```
teams = ["Kansas City Chiefs", "Tampa Bay Buccaneers", "Baltimore Ravens", "Philadelphia
data_t1_1 = get_data(teams)


data_t1_1_tweets = get_tweets(data_t1_1, start = 'T1_start', end = 'T1_end')
```

There are additional steps including binding all of these data sets together, which I'll talk about in more depth.

### 4.2.4 Updating cleaning and sentiment analysis from Study 1

Here I'll go into the small changes from the first study that were made. They include:

- Adding a time identifying column to the tweets data frame
- Writing a function to get sentiments for each player at each time point

### 4.2.5 Modeling

Here I'll discuss multi-level modeling and why in particular we chose multilevel modeling as our approach of choice. I will discuss time, team, and player as clusters. Also math here.

Here is some math that I will adjust for our model

I used the lme4 package in R. More about that too.

## 4.3 Results

### 4.3.1 Summary Statistics

Will add a few, others will be in data appendix.

### 4.3.2 Multi-level Model

```
model3 <- lme(neg_perc ~ outcome * Race + position,
              random = ~ 1 | Team/Name, data = data2)


#summary(model3)
```

The resulting formula: $neg_perc = 66.28\hat{\beta_0} - 19.10Outc\hat{ome}W - 6.28Ra\hat{c}eW - 1.5Posi\hat{ti}onR + 6.55Outcome\hat{W} : RaceW + e$

The intercept for this model suggests that for this data, the percentage of negative words for black players during a game that was lost, holding position constant, is 66.28%. This percentage drops by 19% when the game is won, a significant difference, p $<$ 0.05. When the game is lost but the player is white, this percentage drops by 6.28%, holding position constant. This is not significantly different than for black players, p $>$ 0.05. For a black receiver during a game that was lost, the percentage of negative words will drop by 1.5%. This is not significantly different than for quarterbacks, p $>$ 0.05.

Below is a baseline logistic regression model which can't pick up the random effects.

```
model_l <- glm(pos_perc/100 ~ outcome + Race + position + outcome*Race, data = model_dat
```

```
Warning in eval(family$initialize): non-integer #successes in a binomial
glm!
```

```
#summary(model_l)
exp(coef(model_l))
```

```
 (Intercept)       outcomeW          RaceW      positionr outcomeW:RaceW
   0.5156094      2.1737081      1.3018900      1.0569748      0.7504664
```

As you can see, there are BIG important differences.

Need to discuss model fit.

I'll also tell you what the MLM is doing to account for the random effects.

I plan on adding more variables to the model relating to how the players did in the game (# of yards, etc.)

## 4.4 Conclusion

Want to discuss the significant relationship between outcome and percent of negative words. Also, though race was not significant, it's still directionally what we expected for a loss. Not for a win though.

### 4.4.1 Limitations

Plenty:

- Missing data for some players
- Lack of data because of Twitter limits
- Perceived race
- Wish for more demographic data
- Duplicated tweets

### 4.4.2 Future Directions

Larger sample, more time points. Hindered by cost.

# Chapter 5

# Final Remarks

I would like to bring it all together here without repeating myself from the end of the previous chapter.

Summary? More broad conclusions about race + sport maybe.

Also how Twitter could make data more accessible to researchers.

# Appendix A

# The First Appendix

### A.0.1 Python function to gather tweets mentioning names

```python
def get_tweets_name(data, start, end):

    #getting the twitter handles
    twitter = data.Name.tolist()

    newtwitter = []

    for i in range(len(twitter)):
        a = twitter[i].replace("'", "")

        newtwitter.append(a)

    print(newtwitter)
```

```
#getting start date

start = data[start]

start = start.tolist()

start = start[0].replace("'", "")

print(start)


#getting end date

end = data[end]

end = end.tolist()

end = end[0].replace("'", "")

print(end)



#running loop for tweets
```

```
all_tweets = []

#some_tweets = []

#for i in range(4):

for handle in newtwitter:

    rule = gen_rule_payload(handle + " -is:retweet",
                            from_date = start,
                            to_date = end,
                            results_per_call = 500)

    rs = ResultStream(rule_payload=rule,
                      max_results=2000,
                      max_pages=4,
                      **premium_search_args)

    tweets2 = list(rs.stream())

    [print(tweet.all_text) for tweet in tweets2[0:10]];

    all_tweets.extend(tweets2)

    time.sleep(10)
```

```
#all_tweets.extend(some_tweets)




#creating df




# We create a pandas dataframe as follows:
data_tweets = pd.DataFrame(data=[tweet.text for tweet in all_tweets], columns=['Twee

#adding more columns
data_tweets['len']  = np.array([len(tweet.text) for tweet in all_tweets])
data_tweets['ID']   = np.array([tweet.id for tweet in all_tweets])
data_tweets['Date'] = np.array([tweet.created_at_datetime for tweet in all_tweets])
data_tweets['Likes']  = np.array([tweet.favorite_count for tweet in all_tweets])
data_tweets['RTs']    = np.array([tweet.retweet_count for tweet in all_tweets])
data_tweets['Quoted'] = np.array([tweet.quoted_tweet for tweet in all_tweets])
data_tweets['Q_or_RT'] = np.array([tweet.quote_or_rt_text for tweet in all_tweets])
data_tweets['User_ent_text'] = np.array([tweet.user_entered_text for tweet in all_tw
data_tweets['retweeted_tweet'] = np.array([tweet.retweeted_tweet for tweet in all_tw
data_tweets['user_mentions'] = np.array([tweet.user_mentions for tweet in all_tweets
data_tweets['profile_location'] = np.array([tweet.profile_location for tweet in all_
data_tweets['in_reply_to_screen_name'] = np.array([tweet.in_reply_to_screen_name for
data_tweets['created_at_string'] = np.array([tweet.created_at_string for tweet in al
data_tweets['tweet_type'] = np.array([tweet.tweet_type for tweet in all_tweets])
data_tweets['retweeted_tweet'] = np.array([tweet.retweeted_tweet for tweet in all_tw
data_tweets['all_text'] = np.array([tweet.all_text for tweet in all_tweets])



return data_tweets
```

### A.0.2 Python function to gather tweets mentioning handles

```
def get_tweets_handle(data, start, end):

    #getting the twitter handles
    twitter = data.Twitter_handle.tolist()

    newtwitter = []

    for i in range(len(twitter)):
        a = twitter[i].replace("'", "")

        newtwitter.append(a)

    print(newtwitter)




    #getting start date
```

```
start = data[start]

start = start.tolist()

start = start[0].replace("'", "")

print(start)


#getting end date

end = data[end]

end = end.tolist()

end = end[0].replace("'", "")

print(end)



#running loop for tweets

all_tweets = []

#some_tweets = []
```

```
#for i in range(4):

for handle in newtwitter:

    rule = gen_rule_payload(handle + " -is:retweet",
                            from_date = start,
                            to_date = end,
                            results_per_call = 500)

    rs = ResultStream(rule_payload=rule,
                      max_results=2000,
                      max_pages=4,
                      **premium_search_args)

    tweets2 = list(rs.stream())

    [print(tweet.all_text) for tweet in tweets2[0:10]];

    all_tweets.extend(tweets2)

    time.sleep(10)

#all_tweets.extend(some_tweets)



#creating df
```

```
    # We create a pandas dataframe as follows:
    data_tweets = pd.DataFrame(data=[tweet.text for tweet in all_tweets], columns=['Twee

    #adding more columns
    data_tweets['len']  = np.array([len(tweet.text) for tweet in all_tweets])
    data_tweets['ID']   = np.array([tweet.id for tweet in all_tweets])
    data_tweets['Date'] = np.array([tweet.created_at_datetime for tweet in all_tweets])
    data_tweets['Likes']  = np.array([tweet.favorite_count for tweet in all_tweets])
    data_tweets['RTs']    = np.array([tweet.retweet_count for tweet in all_tweets])
    data_tweets['Quoted'] = np.array([tweet.quoted_tweet for tweet in all_tweets])
    data_tweets['Q_or_RT'] = np.array([tweet.quote_or_rt_text for tweet in all_tweets])
    data_tweets['User_ent_text'] = np.array([tweet.user_entered_text for tweet in all_tw
    data_tweets['retweeted_tweet'] = np.array([tweet.retweeted_tweet for tweet in all_tw
    data_tweets['user_mentions'] = np.array([tweet.user_mentions for tweet in all_tweets
    data_tweets['profile_location'] = np.array([tweet.profile_location for tweet in all_
    data_tweets['in_reply_to_screen_name'] = np.array([tweet.in_reply_to_screen_name for
    data_tweets['created_at_string'] = np.array([tweet.created_at_string for tweet in al
    data_tweets['tweet_type'] = np.array([tweet.tweet_type for tweet in all_tweets])
    data_tweets['retweeted_tweet'] = np.array([tweet.retweeted_tweet for tweet in all_tw
    data_tweets['all_text'] = np.array([tweet.all_text for tweet in all_tweets])


    return data_tweets
```

### A.0.3 Function in R to determine sentiment for each time point

```r
sent_tall <- function(t){

names <- as.list(subset2$name_clean)

datalist = list()

for(i in 1:24) {



  #filter for each person and the correct time
  tweets <- tweets_final %>%
    filter(name_clean_final == names[i],
           time == paste0("t_", t))

  #pick out words
  words <- tweets %>%
    select(ID, full_text_low) %>%
    unnest_tokens(word,full_text_low)

  #creating df of stop words
  my_stop_words <- stop_words %>%
    select(-lexicon) %>%
    bind_rows(data.frame(word = c("https", "t.co", "rt", "amp","4yig9gzh5t","fyy2c
  #anti-join with stop words to filter those words out
  tweet_words <- words %>%
    anti_join(my_stop_words)

  #joining sentiments with non-stop words from tweets
  fn_sentiment <- tweet_words %>%
    left_join(sent_full)

  #creating df with n of sentiments
  df <- fn_sentiment %>%
    filter(!is.na(sentiment)) %>%
    group_by(sentiment) %>%
    summarise(n=n())

  #making df of sentiments for each person
  df_2 <- df %>%
  mutate(player = names[i]) %>%
  spread(key = sentiment, value = n)

  datalist[[i]] <- df_2

}

sentiment_full <- do.call(bind_rows, datalist)

sentiment_full <- sentiment_full %>%
  mutate(totalsentiment = negative + positive,
         neg_perc = negative/totalsentiment * 100,

         pos_perc = positive/totalsentiment *100,

         time = paste0("t_", t),

         player = as.character(player))



return(sentiment_full)


}
```

# References

Angelini, J. R., Billings, A. C., MacArthur, P. J., Bissell, K., & Smith, L. R. (2014). Competing separately, medaling equally: Racial depictions of athletes in nbc's primetime broadcast of the 2012 london olympic games. *Howard Journal of Communications*, *25*(2), 115–133.

Awan, I. (2014). Islamophobia and twitter: A typology of online hate against muslims on social media. *Policy & Internet*, *6*(2), 133–150.

Baumer, B., Cetinkaya-Rundel, M., Bray, A., Loi, L., & Horton, N. J. (2014). R markdown: Integrating a reproducible analysis tool into introductory statistics. *arXiv Preprint arXiv:1402.1894*.

Burdsey, D. (2011). That joke isn?t funny anymore: Racial microaggressions, color-blind ideology and the mitigation of racism in english men?s first-class cricket. *Sociology of Sport Journal*, *28*(3), 261–283.

Christopherson, K. M. (2007). The positive and negative implications of anonymity in internet social interactions:"On the internet, nobody knows you're a dog". *Computers in Human Behavior*, *23*(6), 3038–3056.

Cleland, J. (2014). Racism, football fans, and online message boards: How social media has added a new dimension to racist discourse in english football. *Journal* *of Sport and Social Issues*, *38*(5), 415–431.

Cleland, J., & Cashmore, E. (2014). Fans, racism and british football in the twenty-first century: The existence of a ?colour-blind?ideology. *Journal of Ethnic and Migration Studies*, *40*(4), 638–654.

GitHub, I. (2019). *Hello world. GitHub repository.* `https://guides.github.com/activities/hello-world/`; GitHub.

Gonçalves, P., Araújo, M., Benevenuto, F., & Cha, M. (2013). Comparing and combining sentiment analysis methods. In *Proceedings of the first acm conference on online social networks* (pp. 27–38). ACM.

Graham, B. A. (2017). Donald trump blasts nfl anthem protesters:'Get that son of a bitch off the field'. *The Guardian.*

Hauser, C. (2016). Why colin kaepernick didn?t stand for the national anthem. *Retrieved on November*, *14*, 2016.

Hylton, K., & Lawrence, S. (2016). 'For your ears only!'Donald sterling and backstage racism in sport. *Ethnic and Racial Studies*, *39*(15), 2740–2757.

Intravia, J., Piquero, A. R., & Piquero, N. L. (2018). The racial divide surrounding united states of america national anthem protests in the national football league. *Deviant Behavior*, *39*(8), 1058–1068.

Kearney, M. W. (2018). *Rtweet: Collecting twitter data.* Retrieved from `https://cran.r-project.org/package=rtweet`

Krishnamurthy, B., & Wills, C. E. (2009). On the leakage of personally identifiable information via online social networks. *Proceedings of the 2nd ACM Workshop on*

*Online Social Networks*, 7–12.

Kumar, S., Morstatter, F., & Liu, H. (2014). *Twitter data analytics*. Springer.

Landers, R. N., Brusso, R. C., Cavanaugh, K. J., & Collmus, A. B. (2016). A primer on theory-driven web scraping: Automatic extraction of big data from the internet for use in psychological research. *Psychological Methods*, *21*(4), 475.

Leung, K. (2011). Presenting post hoc hypotheses as a priori: Ethical and theoretical issues. *Management and Organization Review*, *7*(3), 471–479.

Luke, D. A. (2004). *Multilevel modeling* (Vol. 143). Sage.

Meena, A., & Prabhakar, T. (2007). Sentence level sentiment analysis in the presence of conjuncts using linguistic analysis. In *European conference on information retrieval* (pp. 573–580). Springer.

Mittelstadt, B. D., & Floridi, L. (2016). The ethics of big data: Current and foreseeable issues in biomedical contexts. *Science and Engineering Ethics*, *22*(2), 303–341.

Nasukawa, T., & Yi, J. (2003). Sentiment analysis: Capturing favorability using natural language processing. In *Proceedings of the 2nd international conference on knowledge capture* (pp. 70–77). ACM.

R Core Team. (2013). *R: A language and environment for statistical computing.* Vienna, Austria: R Foundation for Statistical Computing. Retrieved from `http://www.R-project.org/`

R Core Team. (2018). *R: A language and environment for statistical computing.* Vienna, Austria: R Foundation for Statistical Computing. Retrieved from `https://www.R-project.org/`

Rainville, R. E., & McCormick, E. (1977). Extent of covert racial prejudice in pro football announcers' speech. *Journalism Quarterly*, *54*(1), 20–26.

Rosen, G. (2018). *Security update.* Facebook. Retrieved from `https://newsroom.fb.com/news/2018/09/security-update/`

Schalter, T. (2017). *What makes nfl games take so long?* FiveThirtyEight. Retrieved from `https://fivethirtyeight.com/features/what-makes-nfl-games-take-so-long/`

Silge, J., & Robinson, D. (2018). *Text mining with r: A tidy approach.*

Statista. (2019). *TV viewership of the super bowl in the united states from 1990-2019 (in millions).* `https://www.statista.com/statistics/216526/super-bowl-us-tv-viewership/`; Statista.

Stodden, V. (2014). What scientific idea is ready for retirement. *Edge.*

Twitter, I. (2019). *Filtering realtime tweets.* `https://developer.twitter.com/en/docs/tweets/filter-realtime/api-reference/post-statuses-filter.html`; Twitter Developer.

Wallace, P. (2015). *The psychology of the internet.* Cambridge University Press.

Weller, K., Bruns, A., Burgess, J., Mahrt, M., & Puschmann, C. (2014). *Twitter and society* (Vol. 89). Peter Lang.

Wickham, H. (2019). *Stringr: Simple, consistent wrappers for common string operations.* Retrieved from `https://CRAN.R-project.org/package=stringr`

Wickham, H., & Henry, L. (2018). *Tidyr: Easily tidy data with 'spread()' and 'gather()' functions.* Retrieved from `https://CRAN.R-project.org/package=tidyr`

Wickham, H., François, R., Henry, L., & Müller, K. (2019). *Dplyr: A grammar of data manipulation.* Retrieved from `https://CRAN.R-project.org/package=`

`dplyr`

Williams, D. R. (1999). Race, socioeconomic status, and health the added effects of racism and discrimination. *Annals of the New York Academy of Sciences*, *896*(1), 173–188.

Zhang, C., Zeng, D., Li, J., Wang, F.-Y., & Zuo, W. (2009). Sentiment analysis of chinese documents: From sentence to document level. *Journal of the American Society for Information Science and Technology*, *60*(12), 2474–2487.