Kelly Murphy

Applied Data Analytics Project – API and text mining

Github repository: https://github.com/Kelly47/ADA-Tweet-Stream-Sentiment.git

# Executive Summary

For this project I was interested in using sentiment analysis to plot people’s emotions real-time using tweets. Specifically, I wanted to see if I could plot the constantly changing emotions of sports fans during a game. I was able to accomplish this using Twitter’s API to stream tweets about two competing teams. I then categorized each tweet based on which team it was about, and doing sentiment analysis on the tweet. Finally, I was able to use ggplot in R to plot the results of the sentiment analysis, and visually represent how each teams’ fans were feeling throughout the course of the competition.

# Introduction

The idea for this project started after learning how to do sentiment analysis. We ran sentiment analysis on books and plotted it to visually depict the changing emotions of the book. After learning this technique I asked myself if this could be done real time with tweets. Specifically, I was curious if I would be able to pull tweets about two sports teams who are playing each other, and be able to plot the emotions of each teams’ fans live during competition. I wanted to start with sports because, in terms of emotions, it is a pretty controlled environment where you know exactly when you should be seeing certain emotions out of certain fans based on when points are scored.

# Data

In order to run a sentiment analysis on tweets, there are several different layers of data being used. First, there is the raw tweet data that is being streamed in. This data has all the information about the tweet such as the user, time, number of favorites and retweets, location, the user’s profile image url, and much more. All of this information is stored in an object called ‘status’. However, we are only interested in the tweet, so we are able to only look at the tweets by pulling ‘status.text’. Once the necessary analysis has been done to get the sentiment, the results of the sentiment analysis for each tweet get written to a tab-delimited file that will be used to plot the sentiment in R. This file has three columns: Name, Time, and Sentiment. Name describes which group the tweet was written about. In my example I did the analysis on an NFL game between the Saints and the Falcons. Therefore the names in the first column of the file are either ‘Saints’ or ‘Falcons’. Next, is the time column. This is simply the system time when the line in the file was written. Finally, there is the sentiment column. This column gives a running total of the sentiment for each group. So for every positive word the number in this column increases by one. For every negative word the number decreases by one. If the word is neutral, then the number stays the same. The following shows a sample of what the output .txt file looks like:

Name Time Sentiment

saints 18:31 0

saints 18:31 0

saints 18:31 1

saints 18:31 1

saints 18:31 1

falcons 18:31 0

falcons 18:31 -1

falcons 18:31 -1

falcons 18:31 -1

falcons 18:31 -2

Over the course of this NFL game the text file accumulated a total of 1,030,817 rows. This includes 369,834 total rows for the Falcons, and 660,983 rows for the Saints

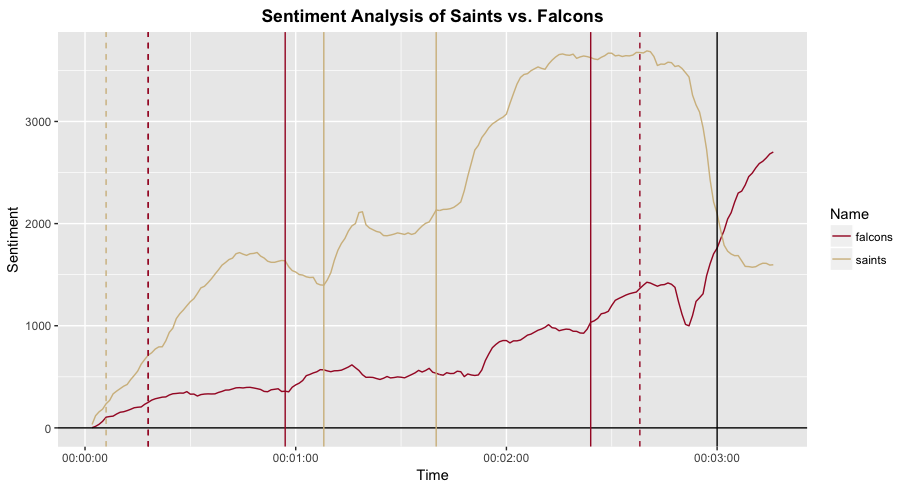
# Methods

In order to run this analysis we start by using Twitter’s API to stream tweets. Tweets are streamed based on the keywords that are entered into the variables ‘group1’ and ‘group2’. As the tweets are streamed, there are if-statements to check which group the tweet is about. If the tweet contains both keywords specified, then we skip that tweet because it will apply the same sentiment to both groups and those sentiment scores would cancel each other out. Within each if-statement the tweet is broken up into individual words. Each word in the tweet is then looked up in a dictionary containing all of the words from the tidytext\_sentiment.txt file. This file contains a comprehensive list of words classified as either positive or negative. If the word in the tweet matches any of these tidytext words, then the total sentiment for the group is incremented accordingly. For every positive word the sentiment is increased by one, and for every negative word the sentiment is decreased by one. The information is written to an output file as explained earlier.

Next, the data needs to be plotted. In order to do this, we move to R. In order to watch the sentiment be plotted in real time, we first create a repeat loop. This is an infinite loop that will keep running until you manually interrupt the console. Within the loop we first read in the output data that has all of our sentiment scores. Once the data is in a data-frame, dplyr is used to group this data. We want to see a minute-by-minute update of the sentiment, so we group all of the data by Name and Time, then show the last sentiment within this group. This will give is the last sentiment score for each group in each minute, therefore showing us the change in each groups sentiment for the entire minute. Then, we will use dplyr to show the top 15 rows based on the time column. This will restrict the data to only the last 15 minutes, or however long you choose. Once the data has been manipulated in this fashion, we then use ggplot to plot each groups sentiment with time on the x-axis and the sentiment score on the y-axis.

# Results

The results from the entire game can be seen below in **Figure 1**. I added vertical lines to the graph to represent scoring plays in the game. The color of the line indicates which team scored and the type of line indicates the type of scoring play. Solid vertical lines indicate a touchdown, and dashed vertical lines indicate a field goal. The black vertical line indicates the end of the game. Based on the plot I would conclude that you are able to run sentiment analysis on streaming tweets to determine how people are reacting real time. We can see that after scoring plays there is generally a noticeable change in emotion, and that the overall sentiment of each teams’ fans does a pretty good job visualizing the emotions of the game.



**Figure 1: Sentiment Analysis of Saints vs. Falcons**

# Future Direction

There are several ways that I would like to improve upon my initial analysis. First, I have never worked with Shiny before, and think it would be worthwhile to create a Shiny app for my plot. I also would be interested in exploring if there was an effective way to have each group be on the same scale. For example, in my plot, Saints fans tweeted almost twice as much, so there reactions were much more noticeable. I tried using dplyr to get the mean sentiment score of each minute, but this did not have a noticeable effect on the plot.