**Project 1: R Basics with Twitter Data**

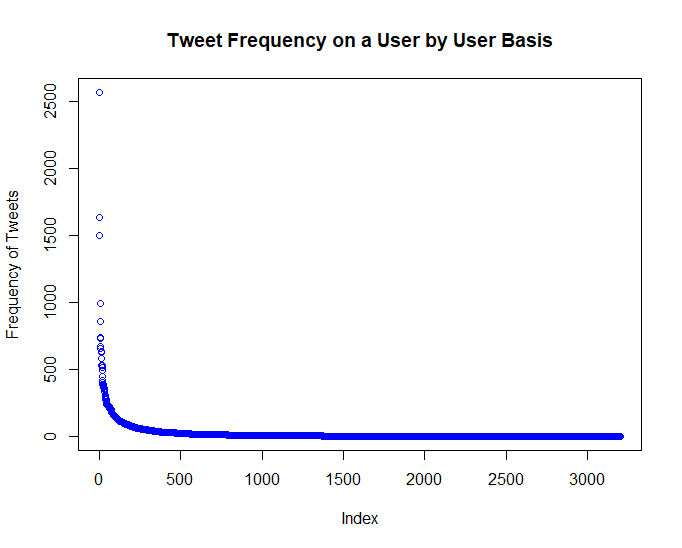
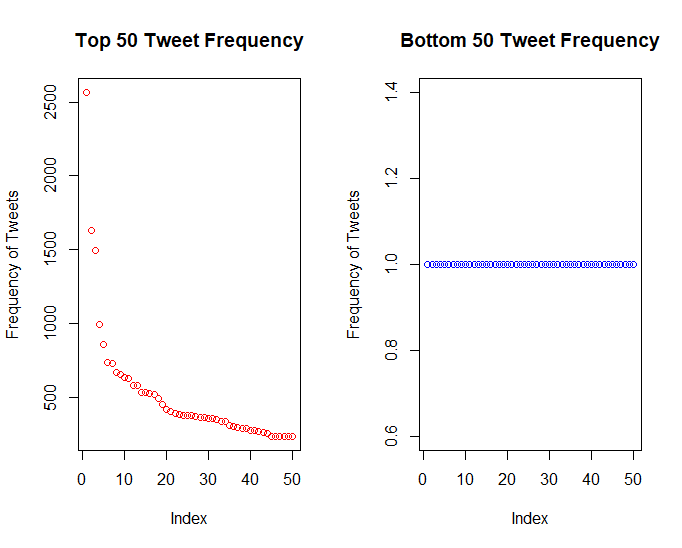
In the time spent working on this project, several things were learned about working with R. R provides an excellent interface for working with datasets and quickly plotting them to get an idea of what the data looks like. This report will cover some of the visualizations developed for this project as well as some of the difficulties incurred by attempting to solve the problems laid out by the instructor.

**Section 1.1: Basic Operations**

This was perhaps one of the easier sections in the project due to my prior experience with R. Reading in a csv was quite easy with R’s built in read.csv functionality. The one problem that was discovered in reading in the data was the inability to provide the path of the csv file starting from the source R file. In Python, if the csv file were in the same directory as the script’s source file, the path to get to the data would simply be ‘./pu2014.csv’. It seems that there is a way to remedy this but it relies on R Studio’s API which seems to be less than ideal if there is any plan to move the R workflow in question to one of the clusters where one would be running the code without R Studio. Conversion of the epoch time (or Unix) time was extremely simple using the as.POSIXct function with an origin of January 1, 1970. The result was

**Section 1.2: Basic Statistics**

This section did present some challenges. Finding the number of tweets by distinct user was fairly simple using R’s built in unique() function. The length of the resulting unique users was the number of users. It was found that there were 3,204 unique users in this dataset. The next challenge took a little more thought and searching of R documentation. Counting the number of tweets for each user was a perplexing problem at first as the first inclination was to simply loop through the data and develop a count saved to a dictionary. Since R discourages this type of programming, I searched through the documentation trying to determine the best way to do this. I had considered an apply function but finally settled on the table function. Table would quickly and easily produce a table of distinct users and the sum of the number of times they occurred in the dataset (i.e. the number of times that user tweeted). To order the data in descending order it was necessary to make use of R’s bracket notation and the order function as seen in the code snippet below.

This snippet of code orders the frequency data in decreasing order and uses that as the row index in tweetCount. A plot was generated from this output and can be seen in **Figure 1a**.

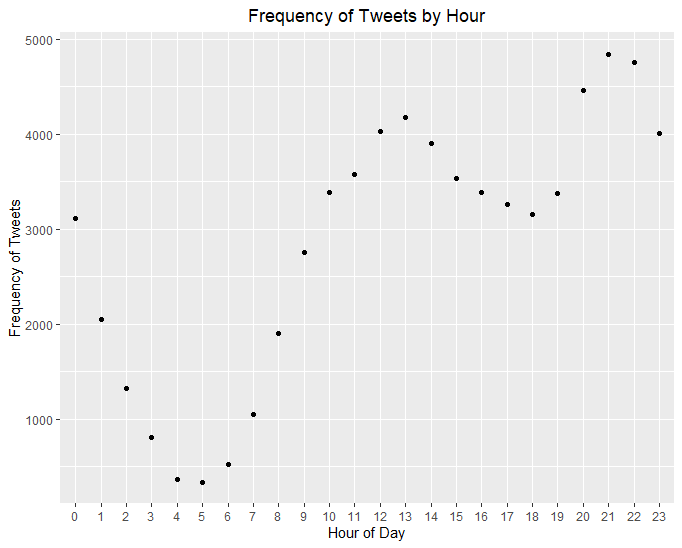
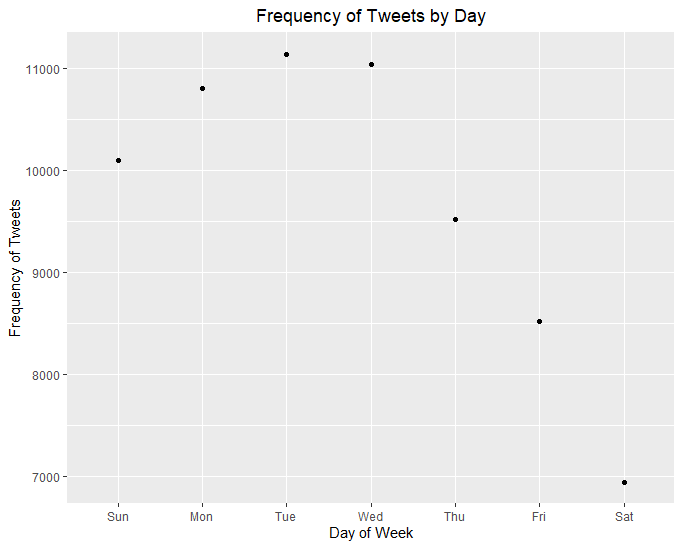
b

a

**Figure 1:** a) Number of Tweets by user. b) Top 50 and bottom 50 tweets by frequency.

The x-axis was intentionally left as an index number rather than user-ID for clarity. This plot does an okay job of showing that a small number of Twitter users generate most of the tweets, it could be better as seen in **Figure 1b**. It becomes very clear that a very small number of users are generating a significant majority of the tweets.

**Section 1.3: Tweeting Behavior Over Time**

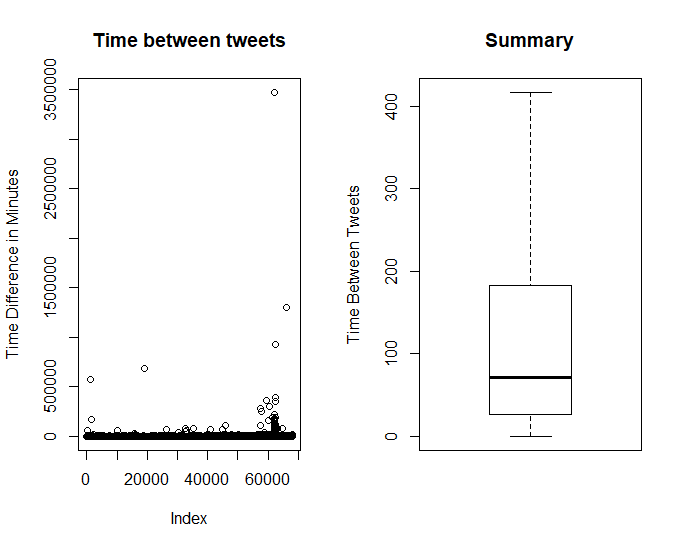
This section took the most time of any of the sections for this project as the visualizations in it took a little more work to get looking right or good. To summarize tweets by day of the week or by hour, I again made use of R’s table function as in previous sections. Table provided an extremely quick count of each tweet by day or hour. The weekdays presented one challenge that took some time to solve. Due to the day of the week being a character rather than numeric value, R didn’t know how they should be ordered when plotted at first. To fix this, it was necessary to invoke the factor function as shown below. Doing so allowed for the plot show in **Figure 2a**.

b

a

**Figure 2:** a) Tweets by weekday. b) Tweets by hour of the day.

**Figure 2a**, demonstrates the distribution of tweets by hour of day. It is interesting to see that Tuesdays are when the most tweets occur and there is a local maxima of tweets around lunch time as shown by the data. Finally, in order to summarize the time between tweets, the diff.date function was used to get the time between each tweet after the tweets were ordered in desceding order. A scatterplot and boxplot are used to summarized the data in **Figure 3**.



**Figure 3:** Time between tweets summarized in a scatterplot and boxplot without outliers.