Project 4
CS6065
Frazier Baker
Jeremiah Greer

Twitter Problem 6:

Twitter users often include hashtags in their tweets as a method of concisely connecting an idea to the rest of their post. Hashtags can set the context of a tweet and give a programmatically searchable indicator as to their topic; therefore, hashtag analysis is very important. In this problem, we desire to determine the most frequent hashtag within our sample set for each day of the week and each hour of the day.

To complete this analysis, we use multiple layers of transformations and actions in Spark. This is a two-part problem, but the parts follow a similar pattern. First, we extract the hashtags and creation time from each tweet, and return pairs of values for each hashtag from each tweet including the hashtag's text and the desired portion of the time (i.e. hour, day of week). This follows the MapReduce custom generation paradigm, and the Spark transformation used is a map. Next, the paired values are grouped by the time element, using the Spark transformation groupByKey. Finally, a map transformation is used to calculate the most frequent hashtag for each timeframe, and the output is printed to standard output using the Spark action collect.

The results of this analysis are shown in the tables below. Table 1 shows the results for the hourly analysis, whereas Table 2 shows the results of the weekday analysis.

Table 1:

Sunday	**Monday**	**Tuesday**	**Wednesday**	**Thursday**	**Friday**	**Saturday**
#dog	#dog	#dog	#dog	#dog	#dog	#dog

Table 2:

12:00AM	*#dog*	*12:00PM*	*#dog*
1:00 AM	*#dog*	*1:00 PM*	*#dog*
2:00 AM	*#winning*	*2:00 PM*	*#dog*
3:00 AM	*#winning*	*3:00 PM*	*#dog*
4:00 AM	*#winning*	*4:00 PM*	*#dog*
5:00 AM	*#dog*	*5:00 PM*	*#dog*

6:00 AM	*#dog*	*6:00 PM*	*#dog*
7:00 AM	*#dog*	*7:00 PM*	*#dog*
8:00 AM	*#dog*	*8:00 PM*	*#dog*
9:00 AM	*#dog*	*9:00 PM*	*#dog*
10:00AM	*#dog*	*10:00PM*	*#dog*
11:00 AM	*#dog*	*11:00 PM*	*#dog*

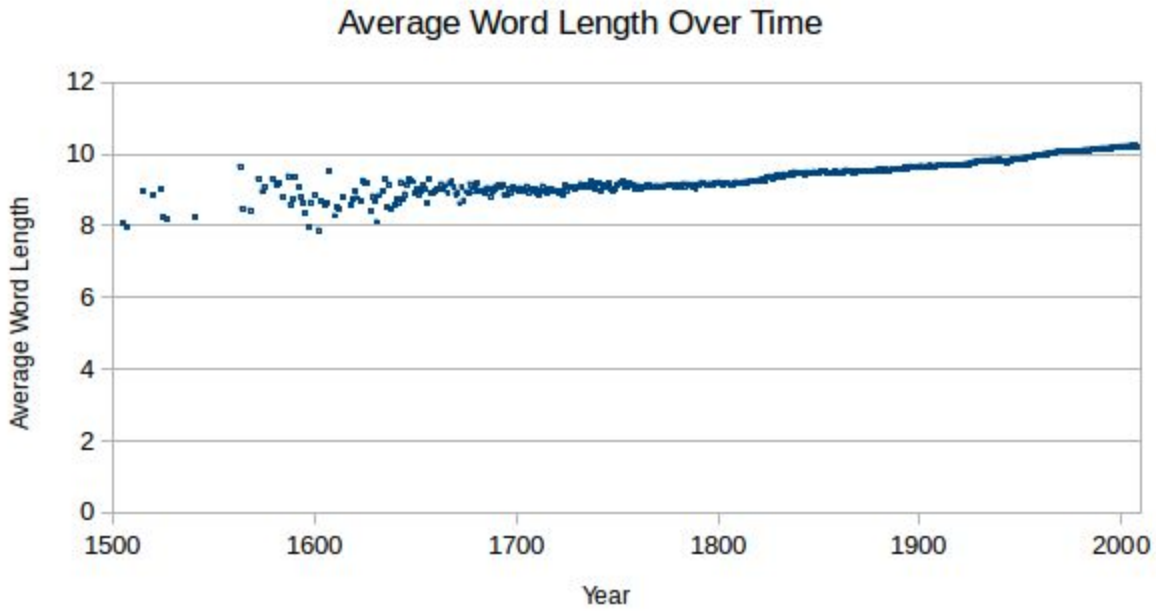
The results of this analysis suggest that the people of the internet enjoy posting about dogs and winning. Though winning may be more popular for a few hours a day, dogs take the prize for every day of the week. While it is understandable and expected that people would share their feelings about both winning and dogs, it should be noted that these results may be influenced by the potentially selective subsetting of the dataset done for the sake of this assignment by the professor for this class, the esteemed Dr. Paul Talaga.

Google 1gram Problem 3:

This question asks “What is the average length of unique words used each year?’ This would potentially show an increase in the complexity of human vocabulary given the average length of words used.

The method of analysis for this question is to sum the lengths of all unique words within the dataset and divide that result by the total number of words, thus giving an average length. To do this in parallel, we once again follow the numerical summarization design pattern to find the average word length. Using Spark, we map the 1gram files into key-value pairs, with the key being the year, and the value being either the length of the word, or the number 1. We then take that result and reduceByKey, summing those values. Finally, we join those sets together, collect and transform them into a python dictionary, and then print the resulting year and division (totalLength/numOfWords).

The results of this analysis are shown in the graph below. The graph plots the average length of unique words for each year. While sporadic at first given the sparsity of the dataset at that time period, this graph indicates that the average length of unique words has increased over time in a slow, linear fashion. This reveals that, over time, the english language has grown more and more complex, though at a slow pace.



Google 1gram Problem 4:

This question asks “What is the average number of syllables of unique words used each year?’ This would potentially show an increase in the complexity of words given the number of syllables used.

The method of analysis for this question is to sum the number of syllables of all unique words within the dataset and divide that result by the total number of words, thus giving an average syllable count. To do this in parallel, we utilize the numerical summarization design pattern to find the average syllable count. Using Spark, we map the 1gram files into key-value pairs, with the key being the year, and the value being either the number of syllables (in this case, vowels) of the word, or the number 1. We then take that result and reduceByKey, summing those values. Finally, we join those sets together, collect and transform them into a python dictionary, and then print the resulting year and division (totalSyllables/numOfWords).

The results of this analysis are shown in the graph below. The graph plots the average number of syllables of unique words for each year. While somewhat inconsistent at first given the sparsity of the data available, the results show that the average number of syllables has stayed about the same, with a small increase between 1800 and 1900, with a decrease after 1900, following a general trend of 3.25 syllables per word.