Load Tweets

Loaded data using Streaming API of Top 10 users based on followers' count. Initialized a table, removed non-ascii characters & sorted the data by followers count followed by screen name.

Expanding URLS

Since many users use URL-shorteners in their tweets, we used the expand URL function to expand the shortened URLs using new HTTP Interface.

Tokenize Tweets

We converted the tweets into tokens. Then, we computed scores based on AFINN. This was done so that we could analyze the tweets as a word cloud. As a part of tokenization, we pre-processed the tweets by splitting them using delimiters, removing numbers, stop-words, etc. and then forming the document matrix by separating all unique words, unique domain names and unique URLs. We form three such separate document matrices for the above three fields. Then we looped over these tokens and updated all the document matrices.

Analyzing the average sentiment of politics tweets

Based on the score file calculated above (using AFINN), we calculate the Net Sentiment Rate (NSR) and generate a histogram. The necessity of this step can be justified by watching the presence of more negativity in the political tweets. Average NSR was found to be –0.48

Frequent Words, Hashtags, Mentions, Cited Websites in Political Tweets

We found the count of each word in the previously created Document Matrix & sorted them by their frequency. Then we generate a scatter plot to show the most frequent words present in political tweets. Similarly, we find the words starting with # from the dictionary of unique words created previously. We then sort these words by their frequency and generate a scatter-plot to show the most frequent hashtags. Similarly, for most frequent mentions (@)

Generating a Social Graph

Now let's think of a way to see the association between users to the entities included in their tweets to reveal their relationships. We have a matrix of words by tweets, and we can convert it into a matrix of users vs. entities, such as hashtags, mentions and links. (A User entity matrix is created)

Handling Mentions
Some of the mentions are for users of those tweets, and others are not. When two users mention another, that forms an user-user edge, rather than user-entity edge. To map such edges correctly, we want to treat mentioned users separately.

Creating the Edge List
Now we can add the user to user matrix to the existing user to entity matrix, but we also need to remove the mentioned users from entities since they are already included in the user to user matrix All we need to do then is to turn that into a sparse matrix and find indices of non zero elements. We can then use those indices as the edge list.

Creating the Graph
Once you have the edge list, it is a piece of cake to make a social graph from that. Since our relationships have directions (user --> entity), we will create a directed graph with `digraph`. The size of the nodes are scaled and colored based on the number of incoming relationships called in-degrees. As you can see, most tweets are disjointed but we see some large clusters of tweets.

Zooming into the Largest Subgraph
Let's zoom into the largest subgraph to see the details. This gives a much clearer idea about what those tweets were about because you see who was mentioned and what sources were cited. You can see a New York Times opinion column and an article from Sweden generated a lot of tweets along with those who were mentioned in those tweets.