Overview

The project option I've chosen is to explore a social media API and use it to collect data for a dashboard. The social media site of interest is Twitter. I am using the Twitter API v2 to collect data, and Tableau to visualize the dashboard.

I want to look at what engagement looks like under a certain topic. The sample topic I've chosen is the recent Kellogg's strike. I collected tweet and user information for almost 45000 tweets under this topic, and some of the information I've visualized in my dashboard include:

- What are the top trending hashtags included in tweets on this topic? (frequency of hashtags included in tweets over time)
- engagement with tweets under a certain topic/hashtag (number of retweets and likes)
- what do these tweets include (links, media, geolocations)
- the profile of the tweet (verified? how many followers?)

Plan changes and improvements since the presentation milestone:

- From the feedback, I have figured out how to request more than 100 tweets. For a topic, I can return up to 45000 tweets.
- From the feedback, and also following my original plan, I added sentiment analysis of tweets. For simplicity I am using the SentimentIntensityAnalyzer from the nltk package, so no training is required.
- Since there is a much larger dataset to work with, relying on Tableau to join the data to overcome nesting is not feasible. So I have parsed the JSON file and generated CSV files that are structured for a relational model.
- Instead of presenting multiple topics in a dashboard, I've decided to only focus on one topic at a time, which the sample topic is the Kellogg's strike discussion. Previously I planned to look at multiple trending topics at a time. However, working with the original dataset made me realise that with multiple topics, the trends were confusing and muddy. A better approach would be focus on one topic at a time.

The information collected

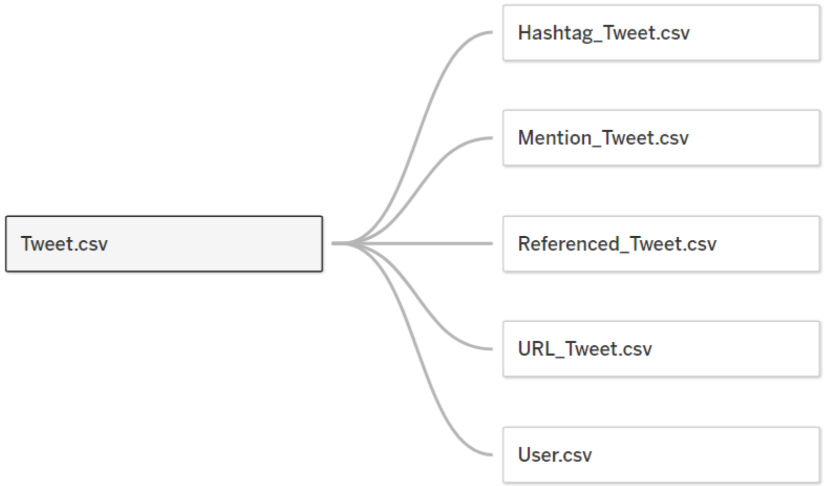
Using the Twitter API v2, the query I ran returned a JSON file containing around 45000 tweets and the information associated with it, like the author id, the time it was created at, if the tweet referenced another tweet and what the type is (a reply or retweet), its public metrics (counts of likes, quotes, retweets, and replies, and if there are any entities included like hashtags, URLs, mentions of other users. The file also contains the information of the users associated with these tweets, like their public metrics (counts of followers, following, listed, tweets), and whether the user is verified or not.

Some definitions for the variables collected to understand the levels of engagement:

- Reply: A reply made by a user is only visible in the reply thread, and not on the user's profile
- Retweet: A user can retweet a tweet to display it on their own profile
- Quote: Like a retweet. However, a caption can also be added
- Listed count: lists are like discussion groups, the listed count displays how many lists a user is in

List of steps and data observation

1. Explore the APIs that Twitter offers and see how I can incorporate the endpoints that they offer into the dashboard
2. Try to request some information with the APIs and get the information onto Tableau (software for visualizing dashboards). Look at the raw data and try to make sense of it. And then play around with it by making some graphs and making notes of any trends I see
    a. After getting the JSON response, my original plan was to directly use the JSON files in Tableau. However, after I figured out how to return more tweets, joining tables from the heavily nested JSON files created tables with too many rows for Tableau to handle.
    I decided to go through the JSON files and convert them to separate CSV files with a relational model so that they could be linked in Tableau:
        Tweet: id, text, user_id, created at, like, quote, reply, type, score
        User: user_id, followers, following, tweet count, listed count, verified
        Hashtag_Tweet: tweet_id, hashtag
        URL_Tweet: tweet_id, url
        Mention_Tweet: tweet_id, (mentions) user_id
        Referenced_Tweet: tweet_id, referenced_tweet_id, type
    Tableau does not support circular relationships, so the linked model is not fully complete. However, the main links are there and they are enough for me to showcase the necessary information.



b. The graphs that I created to visualize the raw were:
   i. Popular hashtags (the frequency of hashtags included in tweets under this topic
   ii. Tweet volume over time and the average sentiment of tweets at that time
   iii. Public metrics overview (the average and max like, quote, reply, and retweet count of tweets under this topic)
   iv. How many tweets contain other entities like hashtags, mentions, and urls
   v. The public metrics of the average profile
   vi. The sentiment distribution

   Some interesting trends I noticed from visualizing this data:
   - Excluding the most popular hashtags mentioning kelloggs, boycotts, and strikes, other popular hashtags are 1u, Solidarity, UnionStrong, Pay, LivingWage, scabs.
   - I was able to collect tweets over the span of 5 days from December 12th to December 15th. The overall volume seems to be declining by each day, meaning I've collected data near the end of the lifetime of this trending topic.
   - The peaks of the tweet volume are around 7 pm-11 pm, and while the overall sentiment is negative, sentiment is more negative around these peaks.
   - The public metrics show that most interactions with the average tweet is through retweeting, as the average tweet gets 6185 retweets, while the average like count is 4, and the average quote and reply count are 0.

3. After I have a good grasp of the data, plan out what filters I will have in the dashboard and see which filters can be used together to best communicate my findings, and put together the dashboard
   a. The filters I have decided to use are:
      i. What hashtags to include

ii. The frequency/popularity of hashtags
iii. Tweet type (original-Null, quoted, replied_to, retweet)
iv. Verified profile - true/false
v. Sentiment score
vi. The hour the tweet was created at

Features and Testing

Features	Testing notes
Connected to the API and specified query parameters to return the information I want	I used a different query for testing, this time using a broader topic: 'Amazon'. I tested it with the maximum number of API calls to see if it would yield exactly 45000 tweets since the query is a lot less specific. However, it only yielded 44885 tweets, which is less than the number of tweets returned by my default query, which is 44903. The third test with an even broader query keyword 'environment' returned 44869 tweets. I quickly realized that the limit of 45000 is not reached due to the fact that this API endpoint only returns tweets within the week.
Utilized pagination: every API request returns up to 100 tweets. For every response from an api request,check if there is a next token, include that in the query to request for the next 100 tweets. This allows collection of up to 45000 tweets every 15 minutes, as only 450 request are allowed in this time frame.	Pagination works well with both the default and the test queries when making the maximum number of API calls.
Parsed the returned nested JSON file to generate CSV files that are structured for a relational model, so that the data is easier to visualize on Tableau.	Since the structure of the JSON file for both the default and test queries are the same, there was no issue parsing all the JSON files to generate CSV files.
Sentiment score calculation for each text	While this is a ready-to-use sentiment analyzer, no training is needed and there are no errors. However, sometimes the score does not reflect the exact sentiment of the tweet. A lot of tweets received positive scores when the sentiment is very negative. Some examples found in Tweet.csv are: ● 'RT @WorkingFamilies: Never believe a company that says it can't afford to pay all of its employees good wages and supply good benefits when…':

	0.8269 ● ‘RT @INAFLCIO: Friends don’t let friends buy Kellogg’s. #BoycottKelloggs https://t.co/U2pmtQzjCI’: 0.7351
Graphs and filters on Tableau to represent all the information collected	Since the structure of the CSV files for both the default and test queries are the same, there was no issue rendering the same graphs and filters for the different datasets.