Wrangle Report

To complete the assignment and realize a study of the WeRateDogs Twitter archive, the data wrangling process consisted in acquiring 3 different datasets:

- The tweets archive
- The results of the image prediction algorithm
- The metadata associated with the tweets

WeRateDogs Twitter archive

The tweets archive has been sent and could be collected on the Udacity website as a CSV file. I could then add this dataset to my project by manually uploading it through the Jupyter notebook interface. The tweets archive dataset can be imported in a notebook using the pandas' **read_csv()** method.

Tweet image predictions

The image prediction dataset has to be fetched from Udacity's server. To retrieve the file, it is necessary to use its location i.e the URL. The python Requests library allows me to send an HTTP request to the server using the URL and get the results as a *response*.

To wrangle this dataset I created a folder named 'images_pred' to later on store the data in it. Then I used the Requests' **get()** and using the URL as argument. Then, I used the **.content** method to write the returning result of the HTTP request into the folder I previously created. The data is being stored as a TSV file in this folder.

To be able to work with this dataset, the pandas **read_csv()** method enables reading the TSV files. I added the arguments sep='\t' to define the tab spaces as separators so the pandas method could build the dataframe consistently.

Tweets' retweet count and favorite ("like") count

The additional metadata associated with the tweets in the archive required some more work to wrangle it.

First step was to get my head around the Tweepy library used to deal with Twitter's API. But most importantly to go through the process of requesting and getting a Twitter developer account.

After obtaining my credential and access token, I could use the template snippet of code to set up the Twitter's API in my notebook thanks to Tweepy's **tweepy.API()**.

Third step was to code a for loop run across the tweets archive using the tweets' id as an argument for the API call get_status(). The call returns a complex JSON format output. To be able to pick up the counts of likes and retweets I had to identify the keys which I did with the help of a fellow data analyst's repository found on internet (https://gist.github.com/dev-techmoe/ef676cdd03ac47ac503e856282077bf2). Although, the API calls has a time limit so as to avoid crashing the servers with too much traffic. As a result the loop is being interrupted after a while before completion. I could fix this issue by adding the *wait_on_rate_limit= True* and *wait_on_rate_limit_notify = True* arguments to **tweepy.API()**. The count of likes and count of retweets are being appended to a list as a JSON format.

The result of the process above is written as a JSON format in a text file to be finally imported as a dataframe in the notebook with the pandas' **from_dict ()**.

To monitor the process I added a timing method to print out how long it took the code to run through the entire dataframe and pull out the tweet's metadata with the time limitations of Tweepy.