DATA WRANGLING REPORT

BY AVWUNU ONAROGHENE, **Udacity Scholar**

This project is aimed to be a showcase of the lessons learned from the Data Wrangling course on the Idacity Data Analyst Nanodegree Program. The dataset in used for wrangling (and analyzing and visualizing) is the tweet archive of Twitter user @dog_rates, also known as WeRateDogs. WeRateDogs is a Twitter account that rates people's dogs with a humorous comment about the dog.

Project Objective

The goal here, is to successfully wrangle (gather, assess, and clean) the WeRateDogs Twitter archive Data and other related datasets.

Project Features

Gathering Data

The WeRateDogs Twitter archive

This file was provided as a file on hand. I downloaded his file manually by clicking the following link: `twitter_archive_enhanced.csv`. Once it was downloaded,I uploaded it and read the data into a pandas DataFrame named `archive_df` using the pandas `read_csv()` function.

The tweet image predictions

This file `(image_predictions.tsv)` is present in each tweet according to a neural network. It is hosted on Udacity's servers and I download it programmatically. I imported the Python requestsand of libraries. Using the `get()` function from the requests library, I got the data using the provided url and saved it in a response variable.

On getting a response value of 200, indicating the request was successful, I used the Python with open function to write the response's content to a tsv file in the same directory. I was then able to read the tsv file into a dataframe named `predictions_df` using the pandas `read_csv()` function with the parameter `sep` set to `'\t'`.

Additional data from the Twitter API

Using the tweet IDs in the WeRateDogs Twitter archive, I queried the Twitter API for each tweet's JSON data using Python's Tweepy library and stored each tweet's entire set of JSON data in a file called `tweet_json.txt` file.

I did this by creating a twitter developer account and created an application for the project. I used app credentials (consumer_key, consumer_secret, access_token, and access_secret) for the Twitter API Authentication. I imported tweepy and json then authenticated `tweepy.OAuthHandler` and set `wait_on_limit` and `wait_on_rate_limit_notify` parameters to True in the `tweepy.api` class.I used `tweet_id` in the Twitter archive dataset to access the data online.

I created an empty dictionary to save failed rweets and set up a timer for start and end time. Unfortunately, I had trouble creating this list and because of the time sensitivity of this project, I had to use the option of accessing project data without a twitter developer account where the needed dataset (`tweet_json.txt`) was provided and all I had to do was read this `tweet_json.txt` file line by line into a pandas DataFrame named `json_data_df` with (at minimum) tweet ID, retweet count, and favorite count.

Assessing Data

Visually: I read the three datasets into their respective dataframes setting `max_colunms` and `max_rows` to None, in a jupyter notebook and scrolled through to search for unclean and untidy data.

Programmatically: I used pandas functions such as `.info()`, `.head()`, `.duplicated()` to carry out assessment.

I was able to detect these issues:

Quality Issues(Dirty Data)

1.`archive_df` missing values in `in_reply_to_status_id`, `in_reply_to_user_id`, `retweeted_status_id`, `retweeted_status_user_id`, `retweeted_status_timestamp`

2.`archive_df` timestamp is object datatype instead of datetime

3.`archive_df` there are 181 retweeted tweets

4.`json_data_df` missing values across multiple columns

5.`json_data_df` user column is a duplicate of `id` and `id_str`

6.`json_data_df id` and `id_str` columns have the same values

7.`json_data_df` `created_at` and `archive_df` timestamp have the same values but different titles

8.json_data_df has three columns with no values

Tidiness Issues

1.`archive_df` doggo, floofer, pupper and puppo columns should be one column

2.`predictions_df` `tweet_id` column is arranged serially

Cleaning data

I made a copy of each dataframe first, then I proceeded to use a Define, Code and Test process and cleaned out the issues.

Then I merged the `archive_df` dataframe and `predictions_df` dataframe on the tweet_id column to create a master dataframe with clean and tidy data called `master_df`, then I used pd.concat to add the `favorite_count` and `retweeted_count` columns from the `json_data_df` dataframe to `master_df`.

Storing Data

After gathering, assessing and cleaning the data, I saved the merged data in a csv file named `twitter_archive_master.csv`.