Data Wrangling Report for WeRateDogs Twitter Data by Elaine Lau
Date: May 30, 2020

This report will expand on the steps used to gather, assess, and clean the Twitter Data from WeRateDogs as performed in Jupyter Notebook: **wrangle_act.ipynb**. WeRateDogs is a Twitter Account that rates dogs and generates humorous tweets about the dogs.

Data was gathered from three different sources. For the first one, the pandas *read_csv* method was used to load in the provided'**twitter-archive-enhanced.csv** file. The second source was accessed thru a url that contained a link to tweet image predictions about what breed of dog (or other thing) was present in each tweet. The *requests.get* method was used to get the file and *file.write* was used to write the contents to **image_predictions.tsv** (a tab separated file). The third source, as well as the hardest-to-obtain, was accessed using the Twitter API and Python's tweepy library to get additional information from WeRateDogs Twitter data such as each tweet's retweet count and favorite ('like') count. Code from provided file **twitter-api.py** was modified to query Twitter's API for JSON data for each tweet ID in the Twitter archive and download it to **tweet_json.txt**. Then the information was load into Jupyter notebook using *json.loads* to get data from the file, *append* to load values into a dictionary, and *pandas.Dataframe* to convert the dictionary into a dataframe.

I accessed the Twitter data visually and programmatically. I visually accessed the **twitter-archive-enhanced.csv** file using Sublime Text editor. I programmatically accessed the data using the following methods: *info*, *describe*, *sample*, and *value_counts*. I used *unique* to examine the dog names and image predictions more closely since the description of the project suggested that dog names probably weren't correct and the image predictions may not be of dogs. I also looped thru the text content of tweets with names that may be incorrect such as those that started with lowercase letter or were labelled None to search for the correct names. I also examined the text of tweets with unusual ratings and for dog stages to determine how to correct issues with this data.

I cleaned the following issues with the data:

1) Removed entries from archived data table that are retweets or replies to other tweets because we only want original tweets data. in_reply_to_status_id is not null for replies and retweeted_status_id is not null for retweets.
2) Replaced name 'O' with OMalley. Replaced erroneous dog names with correct name if found in text column by doing a search for name or with None if not. I discovered some names could be found using '*name is*' or '*named*' search pattern.
3) Removed entries that did not have valid ratings. Replaced erroneous rating numerators with rounded numerators for those ratings with decimal numerators. Checked text for '*/10*' pattern to find correct ratings for rating denominators greater than 10. I discovered some correct ratings were missed because '9/11' or '7/11' were also in the text. I created one column for ratings to improve tidiness.
4) Added 'pup' references to pupper stage, which is the youngest. Created a separate dog_stage column in archived dataframe.
5) Searched for more floofer's by adding fluff to search pattern since floofers are fluffy.
6) Converted timestamp to datetime using pd.to_datetime
7) Removed entries in image predictions and archived data that were not dogs.
8) Converted dog stage and floofer columns to category data types.
9) Added retweets and favorite counts to the archived dataframe.
10) Added the best image prediction (#1) to the archived dataframe.

The cleaned and combined archived dataframe was saved to the **twitter_archive_master.csv**.