Project: Data Wrangling

During this project, I was given a task of assembling a database of tweets posted by the account WeRateDogs and wrangle the received data.

Gathering

In this project, I used 3 sources of information:

1) **twitter_archive_master.csv** – this file was given to me by the Udacity team. I had to download it manually.
2) **image_predictions.tsv** – this file had to be downloaded programmatically using an URL provided by the Team. I used the Requests library to perform that task.
3) **tweet_json.txt** – this file was supposed to be downloaded using Twitter API (Tweepy). Unfortunately, I was not granted access to it (not yet, I hope – as I have not received any confirmation yet). I had to paste the code provided to the Team to make the report consistent. I also had to manually download the file **tweet_json.txt**

Assessing

After the data were gathered, I had to perform visual and programmatic assessment of the three data frames. During this process, the following issues were discovered:

Quality

• **df_arch** table:
- Some tweets are not original tweets, but replies or retweets;
- Some tweets are not about dogs;
- Multiple dog names in the column **name** are incorrect;
- Rating denominators have multiple values that are different from 10;
- Erroneous data types for: `**timestamp**`, `**in_reply_to_status_id**`, `**in_reply_to_user_id**`, `**retweeted_status_id**`, `**retweeted_status_user_id**`, `**retweeted_status_timestamp**`;

• **df_img** table:
- 2075 tweets covered instead of the full set of 2356 tweets in the archive;
- Different name cases in columns `**p1**`, `**p2**`, and `**p3**`;
- Non-existent dog breed types in columns: bookshop, syringe, flamingo...

• **df_rt** table:
- Data types for `**tweet_id**`, `**retweet_count**`, `**favorite_count**` are not int64;
- Data for 3 tweets are missing

Tidiness

• **df_arch** table:
- Dogtionary dog type is split into 4 different columns (`**doggo**`, `f**loofer**`, `**pupper**`, `**puppo**`);
- The column source is not tidy and can be adjusted into different source types

• **df_img** table:
- AI guesses and probabilities are split into 3 different sections (p1, p2, p3)

• **overall tidiness:**

- After cleaning the three data frames, they can be merged into one - using `df_arch` as a main element.

Cleaning

In order to successfully perform the cleaning process, I used the technique proposed by course trainers: for each issue I used the Define-Code-Test methodology and dealt with all the issues one by one. Before any cleaning was done, copies of the original data frames were created using **.copy()** function.

1) At first, I started with the archive data frame (**df_arch**), which had the most of issues to address. I used such methods as slicing, extracting with Regular Expressions, for loops and many more. Some columns were dropped (for example, those related to replies and retweets, as I only needed the original tweets.

2) Then, I turned to the data frame with images (**df_img**). This dataframe had many columns with AI predictions, confidence intervals and a boolean stating if the AI guessed a dog or something else. I decided to leave only one column with predictions of dogs and one with a confidence interval. If neither of the three guesses resulted in a dog breed, the newly created column dog_breed would state “unknown” with the corresponding value of breed_conf = 0 (since we know that a dog must have a breed).

After cleaning, only 4 columns were retained: `**tweet_id**`, `**jpg_url**`, `**dog_breed**`, `**breed_conf**`.

3) The last data frame to clean was the one with retweets and likes. The most important issue was that the data type shown was « object », but with additional digging it was found that those were integers. Nevertheless, I decided to change the format using astype(int) function applied to all the columns.

4) After this was done, I proceeded to merging these three data frames using the pd.merge functions and using `**tweet_id**` as a linking column.