Analyzing Twitter Data – Wrangling Report

1. Gathering Data

The data for this project was gathered from three sources:

- Manually downloading WeRateDogs' *twitter_archive* (provided by Udacity, includes information on ratings, dog names, tweet text and image etc.)
- Programmatically downloading tweet *image_predictions* using `requests`, (provided by Udacity, includes what breed of dog (or object, animal, etc.) is present in each tweet according to a neural network.)
- Programmatically retrieved tweet data gathered via Twitter's API for Python

2. Assessing Data

The gathered data was assessed for data quality and data tidiness, using visual assessment and programmatic assessment techniques. After thoroughly assessing the three datasets, the following issues were identified:

2.1. Quality issues in the twitter archive

- **Invalid datatypes**: *tweet_id*, *in_reply_to_status_id*, *in_reply_to_user_id*, *retweeted_status_id*, and *retweeted_status_user_id* were `int` or `float` instead of `string`. The *timestamp* and *retweeted_status_timestamp* features were `string` instead of `datetime` objects. *rating_numerator* needed to be changed to `float` instead of `int`, to account for decimal ratings.
- **Dog name issues:** All lowercase dog names were no real dog names, but strings like "such" or "as". These needed to be set to `None`. Null values were represented as a `string` instead of `None`.
- **Dog rating issues:**
    - Decimals ratings were not correctly retrieved
    - Notations and rating-like notations were mistaken for ratings
    - Ratings of other objects or numbers in the tweet text were mistaken for ratings, when there was no dog rating present
    - Some tweets rated multiple dogs, which led to unusually high denominators
- **Dog stages issues:**
    - Null values were represented as a `string` instead of `None`.
    - Some tweets had multiple stages assigned. For some of those, this was due to featuring multiple doggos. A few were wrongly categorized.
- **Inclusion of retweets and tweets without images:** these needed to be removed.

2.2. Quality issues in the image predictions

- **Invalid datatypes**: *tweet_id* was `int` instead of `string`. The *timestamp* feature was `string` instead of `datetime` objects.

2.3. Quality issues in the API data

- **Invalid datatypes**: *tweet_id* was `int` instead of `string`. The *timestamp* feature was `string` instead of `datetime` objects.
- **Inclusion of retweets:** these needed to be removed.

2.4. Tidiness issues

- All 3 dataframes referred to the observational unit "tweet", they should form one table.
- The *dog_stages* variable was spread across several columns. This information should be compiled into one column.
- *Timestamp* and *created_at* showed the same information, but are separate columns. One column with this information is sufficient.

3. Cleaning Data

After thoroughly assessing the data, the original data was copied. Then, the issues which were identified in the assessment stage were systematically addressed. The cleaning followed a Define – Code – Test process.

3.1. Methods applied for fixing data quality issues

- `pandas.DataFrame.copy()`
- `pandas.DataFrame.astype(), pandas.Series.astype()`
- `pandas.to_datetime()`
- `pandas.Series.str.islower()`
- `pandas.DataFrame.drop()`
- `pandas.DataFrame.loc(), pandas.DataFrame.iloc(), pandas.DataFrame.query()` and various slicing and indexing techniques.

3.2. Methods applied for fixing data tidiness issues

- `pandas.melt()`
- `pandas.DataFrame.drop()`
- `pandas.Series.isin()`
- `pandas.Series.isna()`
- `pandas.Series.duplicated ()`
- `pandas.DataFrame.append()`
- `pandas.DataFrame.drop_duplicates()`
- `pandas.DataFrame.merge()`

4. Storing Data

After addressing the data quality and tidiness issues, the data was stored in a clean master dataset: *twitter_archive_master.csv*