Data Wrangling Report: @WeRateDogs

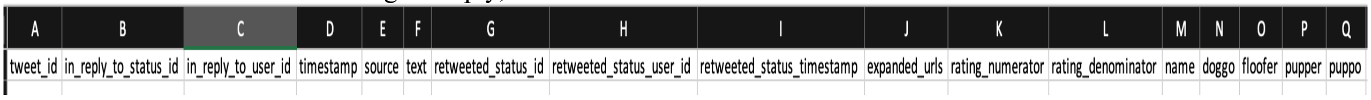
Introduction:

The goal of this project was to wrangle the data regarding a Twitter account (@WeRateDogs) that rates people's dogs with a humorous comment about the dog.

Data Source:

The datasets used were generated from three sources, derived by inculcating different means of gathering data:

1. The first was an internally generated csv file which contained an archive of the account's tweets; this provided a backlog of the dog ratings posted by @WeRate Dogs, with headers such as:
    a. The id assigned to a tweet,
    b. The source of the post
    c. The date and time of the post
    d. The rating (numerator and denominator) given to the dog(s)
    e. The age stage of the dog
    f. Whether a tweet got a reply, retweet.



A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
tweet_id	in_reply_to_status_id	in_reply_to_user_id	timestamp	source	text	retweeted_status_id	retweeted_status_user_id	retweeted_status_timestamp	expanded_urls	rating_numerator	rating_denominator	name	doggo	floofer	pupper	puppo

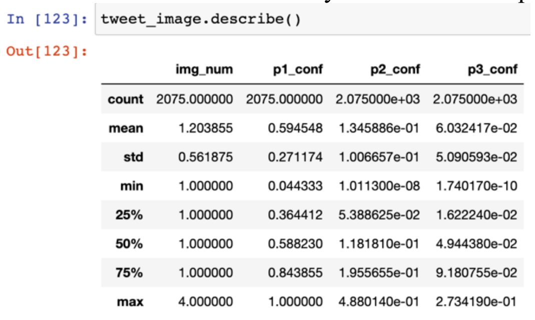
2. The data which contains pictorial information/description of the dogs was a tsv file, which is hosted on Udacity's servers, and needed to be programmatically gathered using the Requests library. It contained headers such as:
    a. Tweet_id
    b. The URL of the image (jpg) posted
    c. Columns for a prediction model
3. The final data source was from the Twitter's API itself, using the Tweepy library. This was needed to get information on the counts of retweets and favorites that each post garnered. Here, an application was made to Twitter, to enable us create a developer profile which would then grant us access to the kind of information we required.

Data Assessment

Assessments were carried out to check for data quality ("is the data dirty?") and data tidiness (is the data messy?") issues; looking for possible pitfalls that could affect the accuracy and usefulness of the data sets/analysis process. This step formed the basis of our cleaning process, as the issues pointed out were noted, alongside possible methods of fixing them.

The datasets were assessed using the two methods:

- **Visual assessment** is done by manually looking through the dataset, either on the pandas' data frame or as a downloaded .csv file.
- **Programmatic assessment** is conducted by calling panda functions and methods to provide insight on the quality and tidiness of the data. For example:
    - df.shape tells us how many rows/columns are within;
    - df.describe returns the summary statistics of the quantitative columns



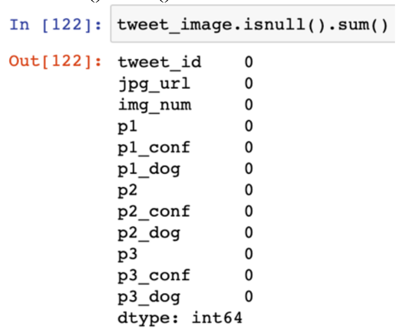
```
In [123]: tweet_image.describe()

Out[123]:
```

	img_num	p1_conf	p2_conf	p3_conf
count	2075.000000	2075.000000	2.075000e+03	2.075000e+03
mean	1.203855	0.594548	1.345886e-01	6.032417e-02
std	0.561875	0.271174	1.006657e-01	5.090593e-02
min	1.000000	0.044333	1.011300e-08	1.740170e-10
25%	1.000000	0.364412	5.388625e-02	1.622240e-02
50%	1.000000	0.588230	1.181810e-01	4.944380e-02
75%	1.000000	0.843855	1.955655e-01	9.180755e-02
max	4.000000	1.000000	4.880140e-01	2.734190e-01

    - df.duplicated() find duplicates

- df.isnull().sum() to know the sum of null values within the columns



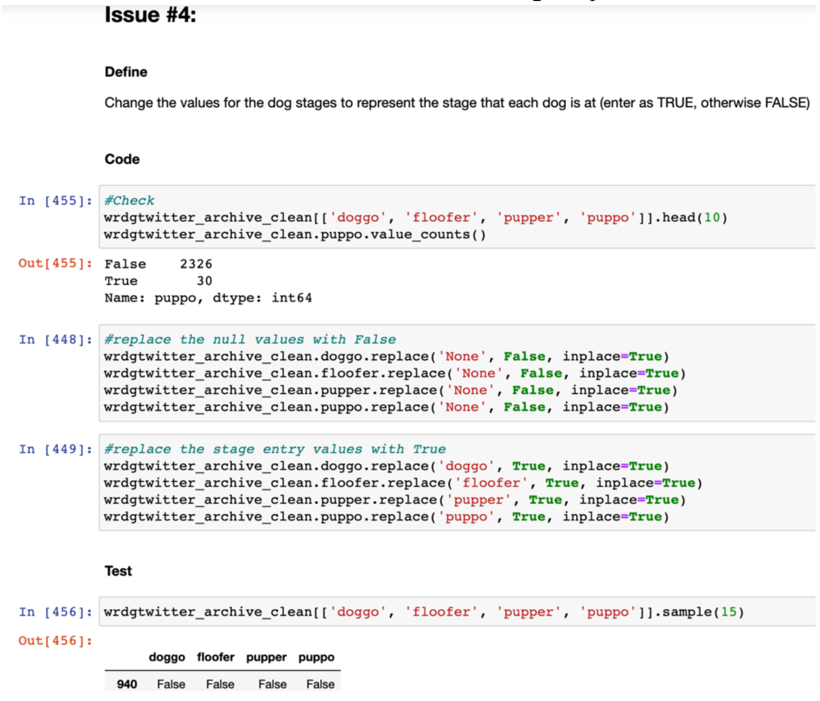
```
In [122]: tweet_image.isnull().sum()

Out[122]: tweet_id     0
          jpg_url      0
          img_num      0
          p1           0
          p1_conf      0
          p1_dog       0
          p2           0
          p2_conf      0
          p2_dog       0
          p3           0
          p3_conf      0
          p3_dog       0
          dtype: int64
```

Data Cleaning

The quality and tidiness issues noted in the "assess" step are fixed here with the use of relevant and common methods, functions carried out on the data frame.
Some things to note:

- Make duplicates of the datasets before cleaning; that way you can always return back to the original sets
- Data Cleaning could be done:
    - Visually; mostly on spreadsheet applications
    - Programmatically to minimize (code) repetition ad save time
- The three steps to programmatic data cleaning were utilized:
    - ***Define***, in writing, the issue to be addressed. This is done to promote scalability and reproducibility by other
    - Fix the issue with ***Code***
    - ***Text*** to confirm that the cleaning script worked



Issue #4:

Define

Change the values for the dog stages to represent the stage that each dog is at (enter as TRUE, otherwise FALSE)

Code

```
In [455]: #Check
          wrdgtwitter_archive_clean[['doggo', 'floofer', 'pupper', 'puppo']].head(10)
          wrdgtwitter_archive_clean.puppo.value_counts()

Out[455]: False    2326
          True       30
          Name: puppo, dtype: int64

In [448]: #replace the null values with False
          wrdgtwitter_archive_clean.doggo.replace('None', False, inplace=True)
          wrdgtwitter_archive_clean.floofer.replace('None', False, inplace=True)
          wrdgtwitter_archive_clean.pupper.replace('None', False, inplace=True)
          wrdgtwitter_archive_clean.puppo.replace('None', False, inplace=True)

In [449]: #replace the stage entry values with True
          wrdgtwitter_archive_clean.doggo.replace('doggo', True, inplace=True)
          wrdgtwitter_archive_clean.floofer.replace('floofer', True, inplace=True)
          wrdgtwitter_archive_clean.pupper.replace('pupper', True, inplace=True)
          wrdgtwitter_archive_clean.puppo.replace('puppo', True, inplace=True)
```

Test

```
In [456]: wrdgtwitter_archive_clean[['doggo', 'floofer', 'pupper', 'puppo']].sample(15)

Out[456]:
```

	doggo	floofer	pupper	puppo
940	False	False	False	False

It is paramount that whatever issues have been noted during the assessment are cleaned.

Analysis and Visualization

To complete the task, a few analysis and visualization steps were carried out, to derive and share some insights from the data.