# Data Wrangling Report

For this project, we had to perform Data Wrangling to gather, assess and clean data provided by @WeRateDogs twitter page. @WeRateDogs is a platform that rates dogs on a scale of 1/10 (ideally) but most often the ratings are more than 10.

***Gathering the data:***

For this project we had to gather data from three sources.

1. WeRateDogs had provided its data of tweets to us in the form of a csv file (twitter_archive_enhanced.csv) which had to be uploaded into the jupyter notebook. This data was then loaded into a pandas dataframe called csv_file_data using the pandas read_csv function
2. The instructor had hosted a file onto the Udacity Servers which included the output of a neural network. This tsv file contained 3 predictions about whether or not the image in the tweet was a dog and the confidence score for each prediction. From this file, we could gather the information about the breed of the dog as well as confirm if the rating was for a dog or not. This file was programmatically downloaded using the requests library and then loaded into a dataframe named tsv_image_data using the pandas read_csv function.
3. Next source of data was the twitter api exposing a lot of meta data about each of our tweets. For this we had to query the twitter api using tweepy library. We had to setup a Twitter developer account to get secure tokens to access the twitter api. Once this was set, we could query the twitter api to gather json data for all the tweet_ids present in our given csv file (twitter_archive_enhanced). This json data was then written to a file called tweet_json_data.txt and was loaded into the jupyter notebook by using the pandas read_json function.

Thus, our three data sources were gathered as three dataframes:
csv_file_data, tsv_image_data, tweet_json_data

***Assessing the data:***

For assessing the first step was to do visual assessment. During visual assessments, multiple issues were found in both the quality and tidiness of the data.

- There were many incorrect values in the ratings columns (rating_numerator, rating_denominator).
- There were many incorrect values in the name columns.
- Following the principles of tidy data, doggo, flooper, pupper and puppo needed to form a single column as they depicted single piece of information.

Programmatic assessment using pandas commands like info, value_counts and drilling down into the rating and name columns gave some more insights:

- Ratings were not uniform as there were many entries with rating_denominator > 10. These were mostly ratings for groups of dogs.
- Also, since data had retweets and non-dog ratings, these had to be removed.
- All the tables had to be joined to form a single entity as they all gave information about a single entity which were the tweets.

***Cleaning the data:***

Cleaning the data involved following a systematic approach of defining the cleaning task, implementing the code and testing the same.

- First a copy of all the three dataframes was made to keep the original datasets handy.
- Many incorrect records that were identified were fixed by finding the index of the incorrect entry and replacing the cell with correct value using the pandas loc function.
- There were a lot of incorrect names in the data but since there was no way to clean them programmatically all at once and they did not add much value to the analysis, they were left as it is.
- Some ratings, which were not ratings but were incorrectly identified as ratings, like 24/7 were dropped using the pandas drop function.
- All the ratings for groups of dogs were dropped using pandas drop function to make the rating_denominator a uniform value of 10.
- Duplicated data and data that had no images were identified using the retweet_status_id and expanded_urls column. These were filtered out to give only original ratings with images.
- tsv_image_data_clean was modified to become a dataframe with only 3 columns - tweet_id, jpg_url, dog_breed
- tweet_json_data was modified to drop all irrelevant columns except tweet_id, favourite_count and retweet_count
- All the three tables were then joined together using tweet_id as the unique identifier. This became the master dataset which was then stored in the jupyter notebook as twitter_archive_master.csv