# wrangle_report

June 27, 2022

## 1 DATA WRANGLING REPORT

#### 1.0.1 By Rachael Olakunmi Ogunye, Udacity Scholar

The purpose of the project is to put into practice all that I learnt from the Data Wrangling lesson in the Data Analysis Nanodegree offered by Udacity. The dataset I wrangled is the tweet archive @DogRates, also known as @WeRateDogs. WeRateDogs is a Twitter account that rates individual's dogs with a humorous comment about the dogs. This ratings almost always have a denominator of 10.

#### 1.0.2 Project Goal:

The project's goal is to effectively wrangle WeRateDogs Twitter data to create interesting analyses and visualization. The Twitter archive is great but additional gathering, assessing and cleaning is required before analysis.

#### 1.0.3 Project Details:

**This report highlights the various steps I took to obtain a clean dataset ready for analysis.**

Gathering Data

Assessing Data

Cleaning Data

#### 1.0.4 Gathering Data

The three different datasets used in this project were obtained as follows:

1) **Enhanced Twitter Archive file:** This was provided by Udacity. I manually downloaded this file by following the link provided in the classroom, twitter_archive_enhanced.csv, I then upload it into my Jupyter notebook workspace and read the data into a pandas DataFrame. I imported the pandas library as it's 'pd' and used the pandas '.read_csv()' function to read the file into the dataframe named 'df_enhanced'.

2) **Tweet Image Prediction file:** This file (image_predictions.tsv) is present in each tweet according to a neural network. It is hosted on Udacity's servers and was downloaded programmatically using the Requests library and the following URL: https://d17h27t6h515a5.cloudfront.net/topher/2017/August/599fd2ad_image-predictions/image-predictions.tsv I import the Python requests and os libraries. With the

’.get()’ function of the requests library, I obtained the data through its url and saved it in a response variable. Response displayed 200, an HTML code for successful. Then using the ’with open’ function, the content was saved to a ’tsv’ file in the same working directory. I then read the downloaded tsv file into the dataframe named ’df_image_predictions’.

3) **Tweet_Json text:** I accessed this file without a twitter account by reading the tweet_json.txt file, provided by Udacity, line by line with the Python ’with open’ function and a ’for’ loop into a pandas DataFrame, called ’df_tweets’ with (at minimum) tweet ID, retweet count, and favorite count.

### 1.0.5 Assessing Data

**Visual Assessment**: Each of the gathered DataFrames above was displayed in the Jupyter Notebook for visual assessment purposes. The DataFrames were also assessed in an external application (Excel).

**Programmatic Assessment**: I assessed programmatically using the various pandas’ functions and methods such as; ’.head()’, ’.tail()’, ’.info()’, ’.describe()’, ’.isnull()’, ’.sample()’, ’.duplicated()’, and ’.value_counts()’.

At the end of the assessment process, eight(8) quality issues and two(2) tidiness issues were observed.

**Quality Issues**
**df_enhanced table**

1. Missing data values in the ’in_reply-to_status_id’, ’in_reply_to_user_id’, ’retweeted_status_id’, ’retweeted_status_user_id’, retweeted_status_timestamp columns and expanded_urls. This is a completeness issue.

2. Wrong representation of the datatype of ’timestamp’ column as ’object’ instead of ’datetime’, a validity issue.

3. Wrong representation of ’tweet_id’ datatype, as ’int’ instead of ’string’. This is a validity issue.

4. Misrepresentation of null values as ’none’ in the different dog stage columns (doggo, floofer, pupper, puppo), a consistency issue.

5. Data inconsistency (lowercase names and erroneous dog names like ’none’, ’a’, ’an’ ’the’, etc.) in the ’name’ column.

**df_image_predictions table**

6. Data inconsistency (lowercase names) in ’p1’, ’p2’ and ’p3’ columns.

7. Wrong representation of ’tweet_id’ datatype, as ’int’ instead of ’string’. This is a validity issue.

**df_tweets table**

8. Wrong representation of ’tweet_id’ datatype, as ’int’ instead of ’string’. This is a validity issue.

**Tidiness Issues**

1. The dog stages are in four different columns (doggo, floofer, pupper, and puppo) instead of one in the df_enhanced table.
2. Three dataframes instead of one

### 1.0.6 Cleaning Data

**This was done using the Define, Code and Test framework but first, I made copies of the original datasets. Th**

df_enhanced_clean

df_image_predictions_clean

df_tweets_clean

The following cleaning efforts were carried out:
**df_enhanced_clean DataFrame**

1) I dropped rows that have retweeted values (′retweeted_status_id′, ′retweeted_status_user_id′ and ′retweeted_status_timestamp′ as retweets will not be used for the analysis. I did this by reassigning only the entries with null values to the dataframe.
2) I dropped columns with over 90% of missing data values and columns that will not be used for further analysis using the pandas′ ′.drop()′ function.
3) I replaced ′None′ in ′doggo′, ′floofer′, ′pupper′ and ′puppo′ columns with empty strings using pandas′ ′.replace()′ method.
4) I created a new column called ′dog_stage′, a categorical variable by joining the ′doggo′, ′floofer′, ′pupper′ and ′puppo′ columns and replaced the empty spaces with null values using pandas′ ′np.nan′ and ′.replace()′ method.
5) I dropped the ′doggo′, ′floofer′, ′pupper′ and ′puppo′ columns using the ′.drop()′ method.
6) I onverted the ′timestamp′ datatype - ′object′ to ′datetime′ using pandas′ ′.to_datetime()′ function.
7) I converted the ′twitter_id′ datatype - ′int′ to ′str′ using the ′astype()′ method.
8) I replaced lowercase and erroneous names with ′None′ in the ′name′ column using a ′for′ loop and pandas′ ′.replace()′ method.

**df_image_predictions_clean DataFrame**

9) I converted all names in ′p1′, ′p2′, and ′p3′ columns to capital with the ′.str.title()′ method.
10) I converted the ′twitter_id′ datatype - ′int′ to ′str′ using the ′astype()′ method.

**df_tweets_clean DataFrame**

11) I converted the ′twitter_id′ datatype - ′int′ to ′str′ using the ′astype()′ method.
12) I merged all three dataframes into ′master_dataset′ on the ′twitter_id′ column using the ′.merge()′ function.

### 1.0.7 Storing Data

The cleaned master DataFrame was stored in a CSV file named ’twitter_archive_master.csv’.

### 1.0.8 Conclusion

This was an exciting project. Although I had a number of challenges, I took time to trace the source of errors, recheck codes and read documentations. I am now familiar with using Python programming language and its packages to successfully wrangle data and gain insights from the data. This is all part of the process of becoming a better Data wrangler.