Data Wrangling Report

Project Objectives

- Perform data wrangling on datasets gathered from three different sources which includes: *'twitter-archive-enhanced.csv'*, *'tweet_json.txt'*, *and 'image_predictions.tsv'.*

- Store and analyze *'twitter_archive_master.csv'*, created after wrangling and combining data from our three different data sources.

This report will focus mainly on the Data Wrangling aspect of this Project.

Gathering Data

- The WeRateDogs Twitter archive (file on hand, manual download of 'twitter-archive-enhanced.csv').
- The tweet image predictions ('image_predictions.tsv'). This file was be downloaded programmatically using the Requests library from a provided URL.
- Each tweet's entire set of JSON data (with at minimum tweet ID, retweet count, and favorite count) in a file called 'tweet_json.txt' were stored using Twitter API and Python's Tweepy library. Each tweet's JSON data was written to its own line.

Accessing Data

During Data assessment both visually and programmatically, made the following observation.

Issue	Observation
Data Quality	1. df_api table - missing values in *geo* column. 2. df_api table - *possibly_sensitive_appealable* column does not contain any data. 3. df_archive table - does not seem to contain complete data for dog stages. 4. df_archive table - *timestamp* column is a string object instead of a datetime object. 5. df_archive table - missing *retweeted_status_id* values 6. df_archive table - missing *retweeted_status_user_id* values

	7. df_img_pred table - column *p1* values start with varying cases (upper and lower). 8. df_img_pred table - column *p2* values start with varying cases (upper and lower).
Data Tidiness	1. df_api and df_archive both have ratings and tweet URL in the *full text* column. *2.* df_api table - the same data in two columns *id* and *id_str.*

Cleaning Data

After data assessment, I went on to clean each DataFrame one after the other and the cleaning steps used are as followed

df_api

1. First created a copy of *df_api* called *df_api_clean*.
2. Dropped redundant columns i.e. columns with over 70% NAN values.
3. Next step to cleaning *df_api_clean* was dropping single cardinality variables that do not offer any information to us i.e. columns with only one value.
4. During assessment, found that the "*id*" column is duplicated in *id_str* so we drop *id_str* column and rename *id* to *tweet_id.*
5. Columns *entities, user,* and *extended_entities* contain data in other columns/tables and appear to be very noisy, so dropped them.
6. During Data Tidiness Assessment, found that column *full_text* contains the ratings and the tweet link alongside the tweet, extracted all three into different columns using Regular Expression and created a function *extract_multi_cols* in order to avoid repetition of this task.

df_archive

1. First created a copy of *df_archive* called *df_archive_clean.*
2. Again, the first step of cleaning *df_api_clean* was to drop redundant columns i.e columns with over 70% NAN values.

3. Convert column *timestamp* to a pandas Datetime object as observed during assessment.
4. Rename column *text* to *full_text.*
5. Use *extract_multi_cols* function to solve the same tidiness issues we saw while cleaning df_api.

df_img_pred

1. First, make a copy of *df_img_pred* called *df_pred_clean.*
2. Convert the dog breed names in columns *p1, p2* and *p3* to lowercase.

Storing Data

Combine our three cleaned Dataframes and store it as a csv file ***twitter_archive_master.csv.***