Wrangle Report

Gather

In this project, I downloaded the WeRateDogs Twitter archive 'twitter_archive_enhance.csv' from the class. Then, read this csv file using pandas's read_csv api and stored it in a dataframe 'df_twitter'.

Since to download 'image_predictions.tsv', I used the 'requests' library using a provided url to download this file and write it to the Project Workspace. Then, I read this tsv file using pandas's read_table api and stored it in a dataframe 'df_predict'

Next, I collected additional information of each tweet id in the 'df_twitter' table using the 'Tweepy' library. In this process, I passed each twitter id to the get_status method and write the tweet data in the _json property to a new line in a newly created file 'tweet_json.txt'. After that, I read the file 'tweet_json.txt' line by line and extract the tweet ID, retweet_count, and favorite_count of each row and add it to a new dataframe 'df_count_data'

Assess

Next, I assessed the collected data by visually inspecting the csv files, and also programmatically assessed the data using pandas to look for quality and tidiness issues.

Below are pandas's api that I used extensively to assess data:

- info() – Inspect a data type of each column
- value_counts() – Inspect frequencies of values in a particular column
- sort_values() – sort values of a particular column
- sample() – randomly select rows in a table

Clean

I used copies of the original dataframes in this process.

Quality Issues

`df_twitter` table

Issues	Process
1. source column should only contain text inside the "a" element	I extracted the innerText of the a element in the source column using a regular expression and pandas' str.extract method.
2. source column should be of type 'category'	The df_twitter_clean.source_name.astype('category') is used to coverted the source_name column to a categorical column
3. timestamp and retweeted_status_timestamp should be of type datetime	pd.to_datetime is used to covert columns to a datetime type
4. Columns for the various stages of dog: doggo, floofer, pupper, and puppo should hold a boolean data type	Convert the string 'None' to a Boolean value False. Otherwise, convert the string to a Boolean value True
5. expanded_urls should contain a list of urls and not a single string for multiple urls and they should not be duplicated	Use "str.split(',').apply(np.unique)" to split text to a list of urls and only keep the unique value
6. Some tweets with no dog names has unexpected dog name such as 'a', 'actually', 'all', 'an', 'by', 'getting', 'his', 'incredibly', 'infuriating', 'just', 'life', 'light', 'mad', 'my', 'not', 'officially', 'old', 'one', 'quite', 'space', 'such', 'the', 'this', 'unacceptable', 'very', and 'None'	List of unexpected was collected during the assess process. Replace any name that matches the list with an empty string

`df_predicts` table

Issues	Process
1. Name of the prediction should be all lowercase to make them consistent	Convert strings in the 'p1', 'p2', and 'p3' columns to lowercase via the 'str.lower()' method
2. p1, p2, p3 should be of type 'category'	Apply the 'astype("category")' to the 'p1', 'p2', and 'p3' columns

Tidiness Issues

Below are **Tidiness** issue that I identified:

Issues	Process
1. The ‘doggo’, ‘floofer’, ‘pupper’, and ‘puppo’ columns for the various stages of dog should be reshaped into a single column 'stage_of_dog'	If all columns contain a Boolean value False, add a string ‘none’ to a newly created column ‘stage_of_dog’. Otherwise, if any column have its column name as its value, use this column name to set a value in the ‘stage_of_dog’. Then, convert the ‘stage_of_dog’ column to a categorical column.
2. df_predict and df_count_data should be part of the df_twitter table	Use ‘pd.merge’ to merge df_predict and df_count_data tables to the df_twitter table