Noelia Guzman
December 18, 2022

Wrangle Report

I. Data Gathering

In the context of this project, three data sets were gathered: twitter-archive-enhanced, image-predictions, and tweet-json. The steps to gathering each data set are as follows:

1. **Twitter Archive Enhanced CSV File:** This file was gathered programmatically using the Request library, or manually with the resources provided.

2. **Image Predictions TSV File:** This file was downloaded programmatically using the requests library. It was hosted and only made available through Udacity servers. Udacity provided the URL that was used to execute the programmatic download of the file.

3. **Tweet JSON File:** This file was gathered programmatically using the Request library and API access through a Twitter developer account. Another option was to manually download the file through resources provided by Udacity.

II. Data Quality Issues and Steps for Data Cleaning

After gathering the three datasets, I merged them into one dataset and saved them locally. Various quality and tidiness issues with the dataset needed to be addressed. Below are the primary issues that were found and cleaned throughout this project.

➔ **Quality Issue #1: Certain column names were not descriptive enough: img_num, p1, p1_conf, p1_dog, p2, p2_conf, p2_dog, p3, p3_conf, p3_dog**

Some column names (p1, p1_conf, etc.) were not descriptive. The non-descript columns were renamed using .rename(), and I replaced the old column names using 'inplace=True.'

The following columns were renamed to be easily understood:

- img_num = image_number
- p1 = first_prediction
- p1_conf = first_prediction_confidence
- p1_dog = first_prediction_is_dog
- p2 = second_prediction
- p2_conf = second_prediction_confidence
- p2_dog = second_prediction_is_dog

➔ **Quality Issue #2: Some column values for the column “name” were invalid: Each value should either be the dog's name or “None.”**

Upon inspecting the column called “name,”I found the data was inconsistently formatted. There were random words in place of dog names (a, the, etc.). Using .value_counts() uncovered that the invalid values began with a lowercase letter. To fix this, I used a for loop to iterate through the column “name,” and replaced all values with a lowercase letter with “None.” I did not have the data to verify the correct dog names that belonged to those dogs/data entries.

➔ **Quality Issue #3: There are extraneous rows of data that represent retweets.**

Several columns in the dataset contain data that indicates the row represents a retweet rather than an original post. These columns are: in_reply_to_status_id, in_reply_to_user_id, retweeted_status_id, retweeted_status_user_id, retweeted_status_timestamp. It was best to remove these rows of data so as not to skew any analysis performed on the data. I created a for loop to clean this quality issue. I used it to iterate through the dataset, and wherever there were no NaN values in the column retweeted_status_id (since that would mean the row of data represents a retweet), that row was dropped.

After dropping the rows, any columns that held retweet data were dropped as well using .drop().

➔ **Quality Issue #4: Some columns need to be corrected data type.**

After using .info() to inspect the columns in the dataset, it became clear that some columns had inappropriate data types and would need to be adjusted. The .astype() function was used to do so, and a dictionary was passed through it that indicated the column name and the correct data type to use for conversion. The timestamp was separately converted using .to_datetime(). The columns converted to the correct data types were the following:

- ID columns --> string datatype
- Categorical variables --> category data type
- Time-related columns --> datetime datatype
- Count-related columns --> integer datatype
- Numerator and denominator --> float data type

➔ **Quality Issue #5: The "timestamp" column contains data that should be split into different columns for readability and analysis.**

The timestamp column contains the date and time that can be split into two columns "date" and "time." To do so, I created a variable "date_time" which is an index object containing the date values present in each of the entries of the DatetimeIndex object.* The "timestamp" column was then split to create "date" and "time" using the attributes .date and .time. To avoid redundancy, I then dropped "timestamp" from the data.

**Source*

➔ **Quality Issue #6: For the dog breed prediction columns ('first_prediction', 'second_prediction', 'third_prediction') values contained unwanted characters.**

The breed prediction columns have values that contain unwanted characters. These included underscores instead of spaces, and lowercase letters at the start of the dog breed names. This was resolved using .replace() to overwrite "_" with " ".

➔ **Quality Issue #7: The HTML tags are in the values for the column “source.”**

The “source” column contains HTML tags in the values which affect readability. To remove the HTML tags from column “source” we can slice the values to exclude the tags at the beginning and end of the string values using .str.slice(), and use this to make a new column. Then, the remaining unwanted characters were replaced using .replace(). Once the values were displayed with more readability, I dropped the original “source” column.

➔ **Quality #8: The column “text” contains a URL.**

The “text” column contains a URL at the end of each value, affecting readability. I removed the URL using a regular expression:
- r'https://\S*' will match any characters (.) that are not whitespace (\S) after the string 'https://',
- The * indicates the pattern can happen more than once
- The “regex=True” argument tells the str.replace() method to interpret the pattern as a regular expression.

➔ **Tidiness Issue #1: The three separate dataframes can be combined into one.**

I began the project with three separate dataframes. I combined the dataframes into one using .merge() with a left join for easy analysis.

➔ **Tidiness Issue #2: There are individual columns for categories of dogs: doggo, floofer, pupper, puppo**

In this dataset, there are four columns that indicate a "level" or "category" of dog: doggo, floofer, pupper, and puppo. To make the dataset tidier, I combined these four columns into one column called “dog_categories.” I did so by first replacing all “None” values with spaces (to avoid confusing strings when we concatenate), then combined the columns using the “+” operator. I then checked the values of the new column to ensure the values were readable. Any values that needed to be cleaned were cleaned by using .loc to find where the values occurred.