Data Wrangling Project

Introduction

In this project we are required to perform the data wrangling process on data collected from a Twitter page "WeRateDogs", then we are required to perform some analysis on the wrangled data. The data wrangling process consists of three main steps:

1. Gathering
2. Assessing
3. Cleaning

We can iterate on these steps; meaning that we can revisit each one of these steps if needed even after we finish cleaning.

Gathering

The first step that we need to do is to gather the data. We are going to collect the data from three different sources:

- CSV file called “twitter-archive-enhanced.csv” (this file contains data extracted from tweets in the WeRateDogs page like the rating, dog name, etc)
- TSV file called” image-predictions.tsv” (this file contains data generated from an image prediction algorithm that is applied on images in the tweet page)
- JSON file called “twee_json.txt” (this file is generated using the Tweepy API that will bring additional tweet data), using the json library we are able to extract the necessary column form the file.

Each of these files need to be loaded into a Pandas data frame.

Assessing

The second step is to assess the data; in this step we want to look for issues that are making our data dirty and messy. We can assess these issues with two methods, visually and programmatically.

Below are the issues that I found categorized as either a quality issue or tidiness issue (eight quality and 3 tidiness).

Quality issues

- Some of the rating_numerator are either less than 10 or greater than 14, after visually checking these entries I found that some of these values are due to error in reading the text(this could occur for text containing dates (e.g. 4/10(4th July) or places (e.g. 7/11).
- column names in tweet-archive-enhanced are ambiguous
- timestamp and retweeted_status_timestamp are not in the correct data type
- 26% of the predictions in p1 are false, which is supposed to be the strongest prediction.
- column names in image-prediction must have meaningful names
- There are entries without tweet_id in the json file
- There are duplicated tweet_id in the json file
- tweet_id is in integer data type not object.

Tidiness issues

- Found that there are 23 entries a rating_denominator that is not 10, however this column will be removed since it's useless.
- Last 4 columns (doggo, floofer,pupper and puppo) should be in one column instead of four.
- Visually looking at the image-prediction data, I see multiple columns that I don't need in my final cleaned data (e.g. p2 data and p3 data).

Cleaning

The final step in data wrangling is to clean the data from the issues that we found in the assessment. Cleaning consist of three steps:

1. Defining
2. Coding
3. Testing

Below is how we define the solution to all of these problems:

Quality cleaning

- Fix entries with numerator not between 10 and 14 by replacing them with the correct values from its text and remove the ones that don't have a correct value.
- Change column names: 'timestamp' to 'tweet_time', 'text' to 'tweet_text','rating_numerator' to 'dog_rating/10', 'name' to 'dog_name' in tweet-archive-enhanced.
- Change data type of tweet_time from object to datetime.
- Remove entries where the first prediction is false.
- Change column names: 'p1' to 'predicted_dog_breed', 'p1_conf' to 'prediction_confidence' in image-prediction.
- Remove rows without a tweet_id.
- Remove rows with a duplicated tweet_id; leaving the ones with both retweet count and favorite count.
- Change data type of tweet_id from integer to object

Tidiness cleaning

- Remove the denominator column since its value is constant and should not be different.
- Replace all the “None” values in the 4 stages column with a null, then create a column called ‘stage’ and add the values from the 4 stages column (one or multiple), after that add a comma between values with multiple stage
- Remove second and third predictions with their coefficient and breed, also remove p1_dog since it has served its purpose.

With these detentions we can translate them into code and test that our data is clean, but before we apply them; we must make a copy for all of the data frames and work on these copies (the code can be found in the Ijupyter notebook).

When we’re done with these steps, we can store the cleaned data and do our analysis on it.