wrangle_report

June 28, 2022

0.1 Data Wrangling process for #WeRateDogs twitter

0.1.1 Project Goal:

Data wrangling involves various steps in getting the data in it's best possible form for analysis and vizualization for insights since raw data comes through as dirty data. This whole process involves 3 major steps. - Gather - Assess - Cleaning

This required for various libraries to be imported which included: - Numpy - Pandas - Requests - Os - Tweepy - Seaborn - Matplotlib

0.2 Data Gathering:

This step involves acquiring and collecting the data required for the analysis. For this case study, the data source comes off the back of Twitter specifically using an API to get raw data off of the (@WeRateDogs) twitter account which rates various breeds of dogs.

0.2.1 The Data:

Enhanced Twitter Archive The WeRateDogs Twitter archive contains basic tweet data in its current state at 2356 rows. The file used was the [twitter-archive-enhanced.csv] file. This was imported using pandas.

Image Predictions File This file was built off a neural network and is used to classify the dog breeds. The file name [image-predictions.tsv] was imported using pandas but specifically seperated by tabs. This was picked off using the provided url and opened said url through the requests Library. Opening this should provide a 200 response which means it has been written to the [image-predictions.tsv] file.

Additional Data via the Twitter API This file is based off of a twitter archive of the (@WeRateDogs) profile and the tweets made by them within a specific time. Using the Tweepy Library to query the Twitter's API, This can be done by creating a twitter developer's account, set up some code to create an API object, Query various Tweet Id's and write its JSON data to a text file [tweet-json.txt]

0.3 Assessing Data:

This part of the process involves programmaticaly and visually looking through the data to assess and find various quality and Tidiness issues.

Programmaticaly: This involves using various methods from nunique(), info(), value_counts() and other methods of code to break down each file specifically looking for Issues.

Visually: This involves looking at each of the 3 dataframes from a human standpoint, scrolling through the data to find any quality or untidy data.

0.4 Cleaning Data:

Assesing the data, shown below are the various quality issues as well as the untidiness found across the 3 datasets. All three were merged to create a singular data set containing as much cleaned information as possible.

0.4.1 Quality issues

1. Null values represented as None in the dog stage category columns in tweet_dogs. The None values were replaced with (") and eventually replaced with Nan.
2. Missing values from the in_reply_to status, in_reply_to user_id columns. These were removed using the .drop() method.
3. Timestamp column not in datetime format. Converted to datetime using astype() and importing datetime
4. Tweet Id is int format rather than a string in tweet-dogs dataframe. Converted using astype(str)
5. retweeted status_id, retweeted_status_user_id,retweeted_status_timestamp, are retweeted values and won't be used in the analysis. These were removed using the .drop() method.
6. retweeted status_id, retweeted_status_user_id,retweeted_status_timestamp & expanded urls have missing values. These were removed using the .drop() method.
7. Follower_count & Friend_count in Json_tweets have repetitve values. These were removed using the .drop() method.
8. Tweet_Id column in image-predictions is an int64 rather than a string. Converted using astype(str)
9. The Id column in json_tweets is the same asa tweet_Id in other dataframes. Changing the column header using the .replace method
10. Tweet_Id column in json-tweets is an int64 rather than a string. Converted using astype(str)

0.4.2 Tidiness issues

1. Having the dog stage(4 variables in 4 columns) rather than one column in tweet_dogs. By adding the various columns into one column ('stage') and then dropping the 4 columns using the .drop() method.
2. Merge all 3 tables (tweet_dogs, image_predictions, json_tweets). These were merged using the .merge() method

0.5 Storing the Data

After merging the data, it was saved as ##### twitter_archive_master

0.6 Conclusion:

This was a quite a long process and at times stuck for periods of time. It required constant searches and explantions. I do wish to to do this again.

In [ ]: