DATA WRANGLING REPORT

Created by Chisom Promise Nnamani, Udacity Scholar

The purpose of this project is to put in practice what I have learned from the Data Wrangling section in Udacity Data Analyst Nanaodegree program. The dataset that is wrangled is the tweet archive @DogRates, also known as @WeRateDogs. We rate dogs is a Twitter account that rates people's dogs with a humorous comment about the dogs. This ratings almost always have a denominator of 10.

Project Goal:

The goal of this project is to effectively wrangle data related related to dog ratings. The data is sourced from the twitter user @WeRateDogs. Once we have effectively gathered, assessed, and cleaned our data in this project, it can be used for our analysis.

This report briefly escribes my wrangling effort.

Project Details:

The tasks of this project are as follows:

Gathering data Assessing Data Cleanig Data

Gathering Data

The data used for this project consisted of three different datasets that were obtained as following:

`Twitter archive file` : This data was provided in the project guideline. I downloaded it to my workspace by clicking on the `jupyer` icon then upload. I imported the python `pandas` library as pd and used the pandas read_csv() function to read the file into a dataframe named `twitter_archive.`

`Tweet image prediction file` : I imported the Python `requests` and `os` libraries. With the get() function of the requests library, I got the data through its url and saved it in a response variable. Response displayed `200` , meaning that it was successful.

Using the Python `with open` function, I wrote the response's content to a `tsv` file in the same working directory. I then read the downloaded tsv file into a dataframe named `image_prediction` .

`Tweet_Json text` : I created a twitter developer account and created an application for the project. I used the app credentials (consumer_key, consumer_secret, access_toke, and access_secret) for the twitter API authentication. I imported `tweepy` and `json` , authenticated

tweepy.OAuthHandler and set `wait_on_limit` to `True` in the API parameter in order to wait after tweet limit (900) and continue automatically at the end of waiting time. I set the needed tweet id to scrape online from the tweet given in the first dataset, created an empty dictionary to save failed tweets and set up a timer for start and end time.

With the Python `with open` function, I created the `tweet_json.txt` and wrote the output to it, I appended failed ones to the empty dictionary created above. I printed the time taken and the failed dictionary.

With the Python `with open` function again and a `for loop` , I read the `tweet_json.txt` line by line and loaded each line as `json` file. I saved each tweet_id, retweet_count, favorite_count, followers_count and friends_count which I later converted to a dataframe named `tweet_json` .

Assessing Data

Once the three tables were obtained, I assessed the data as following:

Visually: I printed the three different dataframes individually in a jupiter notebook and scrolled through left and righ, up and down. Secondly, I visually assessed the csv files in Excel spreadsheet.

Programmatically: I did various programmatic assessment with various python and pandas methods and functions such as .info(), .describe(), .isnull(), .head(), .tail(), .sample(), .duplicated(), .value_counts() and shape.

Cleaning Data

This part of the data wrangling process was divided into three parts: `Define` , `Code` and `Test` .

These three steps were each on the issues stated in the assess section.

First, I made a copy of the original three datasets.

Twitter_archive = df1_clean Image_predictions = df2_clean Tweet_json = df3_clean

Then, I followed the `Define` , `Code` amd `Test` process and made the following cleaning efforts:

- I removed retweets that won't be used for analysis. I was able to do this using the tweet ids.
- I dropped retweeted_status_id, retweeted_status_user_id, retweeted_status_timestamp, in_reply_to_status_id and in_reply_to_user_id columns because they have over 90% of missing values each.
- I combined the four dog stages spread across four columns into one single column.

- I dropped followers_count and friends_count columns as they don't contain necessary values that would be relevant to the analysis.
- I converted the timestamp column from an int to datetime.
- I converted the tweet_id column from integer to string.
- I dropped all values in the name column that started with small letters because it was confirmed that those names weren't dog names.
- I converted the tweet_id column in image prediction table to a string.
- I changed all p1, p2, and p3 values to lower case.
- I converted tweet_id coulmn in the tweet_json dataframe from integer to string.
- I changed the column label from 'id' to 'tweet_id' in tweet_json(df3) dataset.
- I merged the three dataframes to become one dataframe and merge them on tweet_id column.

Storing the Data

After gathering, assessing and cleaning the data, I saved the merged data in a csv file named `twitter_archive_master.csv`.

Conclusion

This project was so much fun for me! Yes, there were situations I encountered errors and I wpould always have to calm down and trace the source of the errors, which is definitely part of the process.

Data Wrangling is a core skill that anyone who handles data should be familiar with.

I was able to polish my skills more in using Python programming language and its packages to successfully wrangle data and gain insights from these data.

In [ ]: