Wrangle and Analyze Data – Udacity Project

Wrangle Report

Introduction:

This document is a report on wrangling efforts in a Jupyter Notebook. The dataset is the tweet archive of Twitter user @dog_rates. WeRateDogs is a Twitter account that rates people's dogs with a humorous comment about the dog. These ratings almost always have a denominator of 10. The numerators, though? Almost always greater than 10. 11/10, 12/10, 13/10, etc. Why? Because "they're good dogs Brent." The goal is to wrangle WeRateDogs Twitter data to create interesting and trustworthy analyses and visualizations.

Data wrangling is a core skill that ***everyone*** who works with data should be familiar with since so much of the world's data isn't clean. The 3 phases of data wrangling is

- Gather
- Assess
- Clean

Gathering Data:
The data for this project was gathered from

1. WeRateDogs Twitter archive by Udacity:
   The Twitter Archive was downloaded from Udacity as a CSV file and was imported into the Jupyter Notebook using pandas' 'read_csv' method. Further, it was stored in a DataFrame named *'df_twitter_archive'.*

2. A URL to download Image Predictions file by Udacity:
   Using the requests library & 'get' method, data was downloaded programmatically in *'image_predictions.tsv'* file. Then loading it in a DataFrame named *'df_image_predictions'* using pandas' 'read_csv' method.

3. Twitter's API:
   We used Tweepy to query Twitter's API for additional data beyond the data included in the WeRateDogs Twitter archive. This additional data will include retweet count and favorite count. Retrieving the tweet_id from the *'df_twitter_archive'* in a list, I made a loop through each tweet & the Twitter API to get each tweet's JSON data. Then, retrived the ('favorite_data', 'retweet_data', 'favourites_count' , 'created_at') and stored it in a list named *'df_list'* . The try-except blocks where used for the tweets corresponding to a few tweet IDs in the archive may have been deleted and the errors where stored in the list named *'error_list'.* Lastly, using the *'df_*list' a panda DataFrame was created named *'tweet_data'*. We got 2331 tweets retrieved with respect to the tweet_id 's and 25 errors.

Assessing Data:

After gathering and loading the data in the DataFrames we assess it to identify data quality & tidiness issues. Issues with content are quality issues. Low quality data is also known as dirty data. Tidiness issues are structural issues that prevent easy analysis. Untidy data is also known as messy data. Hence, I performed Visual & Programmatic Assessment for the data and identified following issues

Quality:

- In 'df_twitter_archive' table, 'source' column format is bad and cannot be read easily.
- We may want to change the column type (in_reply_to_status_id, in_reply_to_user_id, retweeted_status_id, retweeted_user_id) to string because we don't want any operations on them.
- The rating_numerator & rating_denominator columns have invalid values.
- There are invalid names (a, an and less than 3 characters) in 'name' column.
- There are retweeted tweets, and we do not want it.
- Change datatypes for 'tweet_id', 'timestamp', 'source', 'favorites', 'retweets', 'rating_numerator', 'rating_denominator'
- Some tweets have 2 different tweet IDs, that is retweets

Tidiness:

- Dog stages is in 4 columns (doggo, floofer, pupper,puppo). Move into one column
- Merge 'df_image_predictions' * 'tweet_data' into 'df_twitter_archive' table.

Cleaning Data:

We clean all the issues documented while assessing data. First, I created copies of the DataFrames and performed the cleaning steps on them. The programmatic data cleaning was performed as

1. Define: Converting the issues into defined cleaning tasks.
2. Code: Convert definitions to code.
3. Test: To verify cleaning operations worked.

Storing Data:

The cleaned data was stored in *'df_twitter_archive_clean'* with 1987 rows and 19 columns. The data is cleaned and ready for Analysis & Visualization