wrangle_report

June 28, 2022

0.1 Reporting: wrangle_report

0.1.1 Introduction

Data wrangling is the process of gathering data froma variety of sources aand a variety of formats, assess its quality and tidiness then clean it.

In this project, we used Python and its libraries to perform the wrangling process. We looked at the WeRateDogs dataset which 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 but the numerators are almost always greater than 10, 11/10, 12/10, 13/10 etc.

Step #1: Gathering in this step, we gather the three needed datasets: * Enhanced Twitter Archive which contains basic tweet data and is directly downloaded manually. * Additional data via the Twitter API. A query is used to gather this data. * Image Predictions file which is downloaded programatically using the Requests library and the given url.

In the jupyter notebook, we first imported the necessary libraries to enable the successful gathering of the above data through the indicated means. This is followed by reading the data into their respective dataframes using pandas function.

```
archive_df = pd.read_csv(’twitter_archive_enhanced.csv’)
tweets_df = pd.DataFrame(tweets_list, columns = [’id’, ’retweet_count’, ’favorite_count’])
img_pred = pd.read_csv(’image_predictions.tsv’, sep=’’)̑
```

Step #2: Assessing After gathering the three pieces of data through the aforementioned steps, we assess them visually and programatically for quality and tidiness issues.

In visual assessment, the data is displayed in the jupyter notebook and one scrolls through to identify any issue such as typos, missing data or incorrect data. In programmatic, codes are used. In this case, we used pandas function and methods to assess issues e.g. displaying information of the datasets to get any missing entries, datatypes of the variables or specific structural issues that may be present.

```
archive_df.info()
img_pred.info()
```

The following issues were obtained from the datasets:

Quality issues

1. twitter archive table - unnecessary in_reply_to_status_id column and in_reply_to_user_id column

2. twitter archive table - unnecessary retweeted_status_id column, retweeted_status_user_id column and retweeted_status_timestamp column as consideration is on original ratings

3. twitter archive table - Some dogs have ’a’ or ’None’ in the name column

4. twitter archive table - rating_numerator more than 10

5. twitter archive table - timestamp is an object rather than a timestamp

6. twitter archive table - The columns (doggo, floofer, pupper and puppo) are categorical variables yet placed in separate columns

7. image prediction table - Not all ratings are dog ratings

8. image prediction table - p1,p2 and p3 dog breed’s names have no standard capital letter or lowercase names

9. tweets table - column named id rather than tweet_id to make it standard as with the other tables

Tidiness issues

1. All tables - Incosistent number of observations twitter archive(2356), image prediction(2075) and tweets(2354)

2. All tables - Merging of the 3 dataframes to have one master table

Step #3: Cleaning Here, we clean the issues detected and highlighted while aasessing.

1. We first make a copy of each of the original data archive_df_clean = archive_df.copy()

   img_pred_clean = img_pred.copy()
   tweets_df_clean = tweets_df.copy()

2. We use the define-code-test framework, that is, * define - how one will clean the issue in words * code - converting the definition(s) into executable code(s) * test - examine the data to ensure the code is implemented corretly

   Cleaning steps taken to solve the documented issues

- Dropping the in_reply and retweeted_status columns to remain with original ratings only
- Replacing the given ’a’ and ’an’ with ’None’ to make it standard if a valid name is not given
- Converting the timestamp column from a string to datetime
- Melting the four columns (doggo, floofer, pupper and puppo) in two columns dog_class and stage
- Converting the names to lower case to make it standard
- Dropping rows that contain non-dog ratings using boolean columns that return a False value
- Renaming the id column name to tweet_id in the tweets dataframe
- Merging of the three dataframes to one master dataframe

Conclusion The master dataframe is totally not out of issues as data wrangling is an iterative process. We finished the wrangling process in this case with storing the wrangled data in a CSV file named `twitter_archive_master.csv` ready for analysis and visualization.

```
In [ ]:
```