Data Wrangling Report for WeRateDogs Twitter Account Data

Brief:

The dataset wrangled (and analyzed and visualized) in this data wrangling project from Udacity is the tweet archive of Twitter user @dog_rates, also known as WeRateDogs. WeRateDogs is a Twitter account that rates people's dogs with a humorous comment about the dog.

Objective:

- Gather and import data from three different sources and formats.
- Assess the data visually and programmatically.
- Wrangle the data using a three step process of defining the wrangle, writing code and testing the outcome. Eight quality issues and three tidiness issues were identified and processed in this project.
- Three insights and visualisations were produced for the analysis of the final dataset.
- Jupyter Notebooks and Python was used for the wrangling and analysis process and the following Python libraries were utilised:
    - re
    - pandas
    - requests
    - numpy
    - json
    - matplotlib.pyplot
    - tweepy

Gathering the data:

This data analysis consists of three datasets gathered from three different sources.

1. The WeRateDogs Twitter archive - this is a .csv file provided by Udacity. This archive contains basic tweet data (tweet ID, timestamp, text, etc.) for all 5000+ of WeRateDogs tweets as they stood on August 1, 2017
2. The tweet image predictions - this is a .tsv file also provided by Udacity. Images of the tweets were run through a neural network to classify breeds of dogs present in tweets.
3. Additional data from the Twitter API was gathered to produce the retweet count and favorite count of the tweets. Only the last 3000 tweets are available for download using the Twitter API and thus we have this data missing from the earlier tweets in the WeRateDogs Twitter archive.

Assessing the Data:

For the assessment of the data I visually assessed the three datasets using excel and viewing the dataframe in the Jupyter Notebook.

Some key findings:

- Missing values were sometimes written as ‘None’.
- URLs couldn’t be accessed in columns where more than one URL existed.
- Some of the names in the Tweet image predictions dataset were incorrectly extracted.
- Many columns existed between the three datasets, some of them were redundant as they were repeated among the datasets.
- The image prediction dataset was difficult to understand and data could be extracted from this dataset to produce some useful columns i.e. the dog breeds.
- In the image prediction dataset the ‘doggo’, ‘floofer’,’pupper’ and ‘puppo’ columns had many missing values and it had to be determined whether this data was useful and could be simplified into a single column.

The programmatic assessment included a standard set of steps using df.shape(), df.head(), df.info().

Some key findings:

- The json file from the Twitter API didn’t have a common column name to join the other two datasets to.
- Some of the columns were the incorrect datatype. I.e. twitter_ids were an int, timestamp was a string.
- The 'doggo', 'floofer', 'pupper', 'puppo' columns were investigated to see if any of the rows contained more than one of these four categories. If each row is only subject to one category, we can perhaps merge these columns into one. 14 rows have more than one of the four categories. Only 16.37 % of the data in these columns was available, therefore it was decided to not include these columns in the analysis.
- The dataset contained tweets retweeted by WeRateDog’s. These had to be removed.

Quality issues

Before addressing quality issues, copies of each dataframe was made and the dataframes were merged.

Observation	Solution
Dataset contains retweets and replies.	We can remove rows by using Pandas .isna() method on \`retweeted_status_id\`,

	\`in_reply_to_user_id\` and \`in_reply_to_status_id\` columns and only keeping rows that have a NaN value in this column.
tweet_id is an int not a string.	Column datatype was changed using the .astype(str) method.
timestamp is a string not a timedelta.	Column datatype was changed to a datetime64 object using pd.to_datetime. This allows for extraction of date parts.
Incorrect rating for some tweets.	Less than 1 % of the data is not equal to a denominator of 10. All these entries were removed, so that a consistent rating system can be observed, since some of the larger ratings were assigned to a group of dogs. Some of the numerator ratings were also extracted incorrectly and need to be identified using regex on the text column. Datatype of ratings needs to be float.
Some names extracted incorrectly.	Some of the tweets don't have dog names in them, but other words have been falsely identified as names, such as, 'a', 'an', 'the'. These words were identified in the name column by searching for names starting with a lowercase letter. Regular expressions were used to identify these and the .replace() function to replace them with NaN values.
'None' values should be converted to NaN.	Using the .replace('None',np.nan) function across the dataframe, all missing values were correctly assigned as NaN values.
Some urls are duplicated in the expanded_urls column and aren’t separate items in a list, making it difficult to access them.	The urls were split using a deliminator ',' so that they are separate items in a list. The list was then converted to a set, automatically dropping any duplicate urls.
Columns contain dog breeds as well as other objects.	A function was written and applied to the dataframe to identify rows containing dog breeds and replacing all irrelevant objects with NaN.

source column can be simplified.	Since there are only four common values in the \`source\` column, these values were replaced to be more readable and change the column datatype from a string (pandas object) to a category. This is done using .replace and .astype('category')

Tidiness issues

Observation	Solution
Datasets have duplicate columns and relevant data is not merged.	Join dataframes using a left join. This step took place before addressing quality issues so that quality issues pertaining to more than one dataset could be addressed simultaneously.
The doggo, pupper, floofer and puppo columns should be merged into one column.	The columns identifying dog stages can be merged into a single column. If some rows contain more than one dog stage, the stages can be separated by a ',' delimiter.
Unnecessary columns.	Remove and reorder columns not needed for analysis. I.e. most columns originally from the image_predictions dataframe were dropped as one column with dog_breeds was extracted from the data.
timestamp column not split into simpler counter parts.	Using pd.DatetimeIndex the month, day and year can be extracted and assigned as their own columns. This provides efficient access for future time series analysis.