Udacity Data Analyst Nanodegree – We Rate Dogs-Data Wrangling

Author: Michael Taverner
Date: 8/11/2020
Github Repo: https://github.com/Tavnuh/we-rate-dogs

This document outlines the steps taken to gather, analyse and clean the data relating to the We Rate Dogs datasets.

There were three data sets provided for this project:

1. twitter-archive-enhanced.csv – an archive of tweets and associated data from the WeRateDogs twitter account
2. image-predictions.tsv – a list of image predictions based on the tweet archive
3. tweet_json.txt – a list of data regarding retweet and favorite data. Please note that as Twitter would not accept my phone number, I was unfortunately unable to register for the required developer account to retrieve this data myself.

Data Gathering

CSV (twitter-archive-enhanced.csv) – imported into a Pandas DataFrame using pd.read_csv().

TSV (image-predictions.tsv) – pulled from Udacity servers using the Python requests library, saved into the CSV folder in my working directory and read back in as a DataFrame, using pd.read_csv() with '\t' as the separator.

JSON (tweet_json.txt) – While Pandas has a read_json() method, the brief required this JSON file to be read into a DataFrame line-by-line. This was done using a for loop to read each line, appending to a dictionary and creating a DataFrame with the resulting data.

Assessment

To gather the required 8 quality and 2 tidiness issues I used visual assessment (scrolling through the data in the Jupyter Notebook as well as a text editor (Notepad++)), which illuminated the bulk of the issues that were worked on, and programmatic assessment such as Pandas' df.info(), and a check that confirmed the presence of multiple Dog Stages in the relevant columns which informed a cleaning method in the Data Cleaning stage.

Clean – Operations

Quality:

1.df_archive source values contains HTML tags – While this was the issue as it presented itself, after splitting the string into it's components, I found the most valuable part of each string and replaced the value in that column with that component – eg. "Twitter for iPhone"

2.df_archive timestamp values contains extraneous characters "+0000" and is not timestamp datatype – The extraneous characters related to missing millisecond information, however given that all contained zeros theses were removed, and then the column datatype was transformed into a Pandas Datetime format using pd.to_datetime.

3.df_archive values in in_reply_to_status_id, in_reply_to_user_id, retweeted_status_id and retweeted_status_user_id are floats where they should be objects. They are numbers but have no mathematical value. – As these are ID numbers, they don't have any mathematical value and

therefore are better served as strings. These were cleaned with the .astype('str') method and a loop was used to save on code length.

4. df_archive contains Retweets that are not part of this analysis. – Anything row with a value in these columns were removed as per the brief that the data should not contain retweets.

5.Dog stage columns doggo, floofer, pupper, puppo contain the word "None" where there should be no value – using a similar loop to issue 3, I replaced with the string “None” with an empty string.

6. df_images p1, p2 and p3 values contain underscores – As with issue 3 and 5, the underscores were replaced looping through the applicable columns and replacing underscores with space characters.

7.df_images p1, p2 and p3 values contain a mix of upper and lowercase characters – Looped through the relevant columns and used the str.lower() method to convert all characters to lowercase.

8.df_tweet_info timestamp values not in correct datetime format – The datetime values were in the incorrect order, and to transform them into ISO-8601 format I split the strings into their individual components and assigned them to columns, then rebuilt the date column using string concatenate columns, and finally applied pd.datetime().

Tidiness:

1.Dog stage columns doggo,floofer,pupper,puppo is a single variable and should be represented in one column – string concatenation was used to combine the values into a single variable column. The values that contained two dog stages were located and split manually using a comma via the .replace() function.

2. df_tweet_info contains data that should be included in df_archive, as should df_images – the df.merge functions were used to merge the three DataFrames, and duplicate columns were dropped.