wrangle_report

March 18, 2022

1 Introduction:

Real-world data rarely comes clean. Using Python and its libraries, you will gather data from a variety of sources and in a variety of formats, assess its quality and tidiness, then clean it. This is called data wrangling. You will document your wrangling efforts in a Jupyter Notebook, plus showcase them through analyses and visualizations using Python (and its libraries) and/or SQL.

The dataset that you will be wrangling (and analyzing and visualizing) is the tweet archive of Twitter user [@dog_rates](https://twitter.com/dog_rates), also known as WeRateDogs. 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." WeRateDogs has over 4 million followers and has received international media coverage.

WeRateDogs downloaded their Twitter archive and sent it to Udacity via email exclusively for you to use in this project. This archive contains basic tweet data (tweet ID, timestamp, text, etc.) for all 5000+ of their tweets as they stood on August 1, 2017. More on this soon.

2 Wrangle Steps Overview

Your tasks in this project are as follows:

- 1: Gathering data
- 2: Assessing data
- 3: Cleaning data
- 4: Storing data

3 1- Gather Data:

Gathered all three pieces of data as described below in the wrangle_act.ipynb notebook. #### 1- The WeRateDogs Twitter archive: I Downloaded this file manually by clicking the following link: twitter_archive_enhanced.csv. Once it is downloaded, I uploaded it and read the data into a pandas DataFrame. #### 2- The tweet image predictions This file (image_predictions.tsv) is present in each tweet according to a

neural network. It is hosted on Udacity's servers and I downloaded it programmatically using the Requests library and the following URL: here #### 3- Data from the Twitter API I used tweet_json.txt provided by udacity since Tweeter refuse my API access

4 2- Assessing Data

In this section I defined 2 issues:

4.0.1 Quality issues

1. 'None' assigned instead of 'NaN' for empty missing data {visual assessment}
2. 'tweet_id' not a string. {programmatic assessment}
3. 'source' column contains tag html. {visual assessment}
4. timestamp not in type dtime {programmatic assessment}
5. column 'name' has values('a', 'Mo', 'Bo', 'O', 'Al', 'my', 'an', 'by', 'Ed', 'JD', 'Jo') {programmatic assessment}
6. Rating dinominator has different values instead of 10 {programmatic assessment}+{visual assessment}
7. expanded_urls has incorrect urls and duplicates such like:(https://twitter.com/dog_rates/status/673320132811366400/photo/1,https://twitter.com/dog_ra) {programmatic assessment}
8. Column names are incomprehensible to the reader such as ('P1', 'P2', 'P3') and contain strange Predictions(spatula, barrow, minibus,paper_towel,laptop) {visual assessment}
9. retweet_status has one value 'Original tweet' {visual assessment}
10. columns no need {'in_reply_to_status_id', 'in_reply_to_user_id', 'retweeted_status_id', 'retweeted_status_user_id',retweeted_status_timestamp', 'rating_denominator','img_num'}
11. Column names are not clear to the user { 'source', 'text', 'name'}

4.0.2 Tidiness issues

1. Dataframes must be one df with No retweet ids {visual assessment}
2. Dogtionary in 4 columns instead of one {visual assessment}
3. expanded_urls and url have same values {visual assessment}

5 3- Cleaning Data

- I Have made a copy of the original data before cleaning.
- I have used the Define-Code-Testframework.
- I have documented Define-Code-Testframework.
- I have documented each issue in a few Sentences.
- I have successfully cleaned all issues identified in the assessing phase.
- I have created a tidy master dataset with all pieces of gathered,cleaned data.

6 4- Storing Data

In this section I store the cleaned master DataFrame in a CSV file with the main one named twitter_archive_master.csv.