INTEROFFICE MEMORANDUM

TO: UDACITY DATA ANALYST COMMUNITY

FROM: PETER SCHULD

SUBJECT: WRANGLE REPORT
TWITTER WERATEDOGS
WRANGLE_ACT.IPYNB
TWITTER_ARCHIVE_MASTER.CSV

DATE: FEB. 10TH, 2020

INPUT FILES: TWITTER-ARCHIVE-ENHANCED.CSV [FILE PROVIDED BY TWITTER]
TWEET_JSON.TXT [ADDITIONAL DATA VIA THE TWITTER API]
IMAGE-PREDICTIONS.TSV [UDACITY IMAGE PREDICTIONS FILE]

WeRateDogs provided a CSV-file with their Twitter archive exclusively for Udacity to use. This archive contains basic tweet data (tweet ID, timestamp, text, etc.) for all 5000+ of their tweets as they stood on August 1, 2017.

Goal:

Wrangle WeRateDogs Twitter data to facilitate thorough analysis and visualizations.

Gathering Data

I have gathered each of the three pieces of data as described below in a Jupyter Notebook titled wrangle_act.ipynb:

The twitter-archive-enhanced.csv Twitter archive. The file was provided by download and I have used to pandas' read_csv() function to import the data into the **d1** dataframe.

Description of the data in the columns:

d1 dataframe columns (17 columns / 2356 observations)

- **[tweet_id]**: Unique Identifier of a Tweet (18-digit numerical) [Numerical]
- **[reply_to_status_id]**: In the context of someone else's tweet [Numerical]
- **[in_reply_to_user_id]**: Reply to a specific user [Numerical]
- **[timestamp]**: Date/Time of the tweet [Date]
- **[source]**: Origin of the tweet (e.g. web, mobile device) [Unicode]
- **[text]**: Text of the tweet [Unicode String]
- **[retweeted_status_id]**: Original tweet (if re-tweeted) [Numerical]. Retweets can be distinguished from typical Tweets by the existence of a retweeted_status!
- **[retweeted_status_user_id]**: User of the original tweet. [Numerical]
- **[retweeted_status_timestamp]**: Original date.time of the re-tweeted tweet. [Date]
- **[expanded_urls]**: Original URI (before URL minification) [Unicode String]
- **[rating_numerator]**: Rating numerator extracted from text. Should be on a scale from 0-10, but oftentimes is higher. [Numerical]
- **[rating_denominator]**: Rating denominator extracted from text. [Numerical]
- **[name]**: Dog's name extracted from text. [Unicode String] proper noun
- **[doggo]**: Doggo stage extracted from text. [Boolean]
- **[floofer]**: Floofer stage extracted from text. [Boolean]
- **[pupper]**: Pupper stage extracted from text. [Boolean]
- **[puppo]**: Puppo stage extracted from text. [Boolean]

Using the tweet IDs in the WeRateDogs Twitter archive, I have queried the Twitter API for each tweet's JSON data using Python's Tweepy library and store each tweet's entire set of JSON data in a file called TWEET_JSON.TXT file. Each tweet's JSON data was written to its own line. Then I read this .txt file line by line into a pandas dataframe called **d2** with [tweet ID], [retweet count], [favorite count] and [text_range]. I had to apply for a Twitter developer access to receive Twitter API keys necessary for the data download.

d2 dataframe columns (4 columns / 2332 observations):

- **[tweet_id]**: Unique Identifier of a Tweet [Unicode]
- **[favorite_count]**: Nullable. Indicates approximately how many times this Tweet has been liked by Twitter users. [Integer]
- **[retweet_count]**: Number of times this Tweet has been retweeted [Integer]
- **[display_text_range]**: identifies the start and end of the displayable content of the tweet [Unicode]

There are 24 observations (i.e. tweets) that have failed to download. The error messages 'No status found with that ID.' or 'Sorry, you are not authorized to see this status.' were provided.

The tweet image predictions are presented in each tweet according to a neural network. This file (image-predictions.tsv) is hosted on Udacity's servers and I have downloaded it programmatically using the **Requests library** and the following URL: https://d17h27t6h515a5.cloudfront.net/topher/2017/August/599fd2ad_image-predictions/image-predictions.tsv. Thereafter, I have imported the table using the **pandas' read_csv()** function.

image_df columns (12 columns / 2075 observations):

- **[tweet_id]**: Unique Identifier of a Tweet [Numerical]
- **[jpg_url]**: Image URL [Unicode]
- **[img_num]**: image number that corresponded to the most confident prediction (numbered 1 to 4 since tweets can have up to four images) [Integer]
- **[p1]**: the algorithm's #1 prediction for the image in the tweet [Unicode]
- **[p1_conf]**: how confident the algorithm is in its #1 prediction [Float]
- **[p1_dog]**: whether or not the #1 prediction is a breed of dog [Boolean]
- **[p2]**: the algorithm's #2 prediction for the image in the tweet [Unicode]
- **[p2_conf]**: how confident the algorithm is in its #2 prediction [Float]
- **[p2_dog]**: whether or not the #2 prediction is a breed of dog [Boolean]
- **[p3]**: the algorithm's #3 prediction for the image in the tweet [Unicode]
- **[p3_conf]**: how confident the algorithm is in its #3 prediction [Float]
- **[p3_dog]**: whether or not the #3 prediction is a breed of dog [Boolean]

Assessing Data

Next, I have assessed the data visually and programmatically for quality and tidiness issues. I have detected and documented **27 quality issues** and **3 tidiness issues** in the WRANGLE_ACT.IPYNB Jupyter Notebook.

SUMMARY of the Quality and Tidyness problems

Quality Rule: Completeness

1. The proper dog's name 'Howard' in **`df1`** [`name`] (row # 35) is not picked up from the text and the tweet is wrongly identified as a tweet without a dog's name.
2. **`df1`** [`rating`] in (row # 516) is 24/7 but `text` does not include a rating
3. **`df1`** [`rating`] in (row # 290) is 182/10 totally out of line with the rating scale (i.e. not valid rating)
4. **`df1`** [`rating`] in (row # 979) is 1776/10 totally out of line with the rating scale (i.e. not valid rating)
5. **`df1`** [`rating`] in (row # 2074) is 420/10 totally out of line with the rating scale (i.e. not valid rating)
6. **`image_df`** [`jpg_url` has 66 duplicate images (i.e. **retweets**) that appear twice in two rows each.
7. **`df1`** contains 181 **retweets** with entries in `retweeted_status_id` , `retweeted_status_user_id` and `retweeted_status_timestamp`

Quality Rule: Validity

8. **df1** [source] contain the characters *'<**a href**='* before valid HTTP address
9. **df1** [name] Some dog names are not proper nouns but adjectives like 'this' (row # 1120) or 'incredibly' (row # 542)
10. Invalid zero measurement of **df1** [rating_denominator] in (row # 313). The valid rating mentioned in the text is 13/10.
11. Incorrect value of '960' in **df1** [rating_numerator] (row # 313). The valid rating mentioned in the text is 13/10.

Quality Rule: Accuracy

12. Improper data type **integer** in column **df1** [rating_numerator] unable to display rating nuances expressed as **float** in the text. Example: Rating of 13.5/10 in (row # 45)
13. Erroneous rating in **df1** [rating_numerator] (row # 45). Correct rating of 13.5/10 mentioned in the text.
14. Erroneous rating in **df1** [rating_numerator] (row # 1068) confuses historical date 9/11 (i.e. September 11th, 2001) mentioned in the text with the rating, despite mentioning of a proper rating 14/10 in the tweet message.
15. **df1** [rating_numerator] / [rating_denominator] in (row # 340) is 75/10 but text states 9.75/10
16. **df1** [rating_numerator] / [rating_denominator] in (row # 695) is 75/10 but text states 9.75/10
17. **df1** [rating_numerator] / [rating_denominator] in (row # 763) is 27/10 but text states 11.27/10
18. **df1** [rating_numerator] / [rating_denominator] in (row # 1202) is 50/50 but text states 11/10
19. **df1** [rating_numerator] / [rating_denominator] in (row # 1712) is 26/10 but text states 11.26/10
20. **df1** [rating_numerator] / [rating_denominator] in some rows rate several dogs at once (e.g. 144/120)

Quality Rule: Consistency

21. **df1** data type of [timestamp] and [retweeted_status_timestamp] does not allow for datetime calculations (e.g. time difference).
22. [tweet_id] has inconsistent data types in different data frames (i.e. *string* in df2 and *integer* in **df1** and **image_df**).
23. **df2** data type of [favorite_count] and [retweet_count] does not allow numerical calculations (e.g. additions)
24. **df1** [retweeted_status_id] is in a *float* format rather than an *integer*
25. **df1** [retweeted_status_user_id] is in a *float* format rather than an *integer*
26. **df1** [in_reply_to_status_id] is in a *float* format rather than an *integer*
27. **df1** [in_reply_to_user_id] is in a *float* format rather than an *integer*

Tidiness Rule: Each variable forms a column

- **df1** [`rating_numerator`] and [`rating_denominator`] combined contain one variable rating in two columns. All ratings should be expressed on a scale of [0 to 10] (like 0 to 100%) but can get higher to reflect the unique WeRateDogs® rating.
- **df2** [`display_text_range`] contains two variables (`'start`_text_range' and `'end_text_range'`) in one column (rule: each type of observational unit forms a table)

Tidiness Rule: Each type of observational unit forms a table

- The tweet data is divided between the tables **df1**, **df2** and **image_id**. Cleaning includes merging individual pieces of data according to the rules of tidy data. All 3 tables share the type of observational unit (i.e. Original Twitter WeRateDogs tweets (no retweets) with ratings for dogs.

Cleaning Data

I have cleaned each of the 27 quality issues and 3 tidiness issues documented while assessing in wrangle_act.ipynb .

- Identify each step of the data cleaning process (defining, coding, and testing)
- Clean data using Python and pandas
- Test cleaning code visually and programmatically using Python

Define QUALITY (Completeness):

1. Change dog name for **df1** [name] (row # 35) to 'Howard'
2. Drop row # 516
3. Drop row # 290
4. Drop row # 979
5. Drop row # 2074
6. Identify and remove retweets in **`df1`**
7. Remove the empty columns [`retweeted_status_id`], [`retweeted_status_user_id`] and [`retweeted_status_timestamp`] from the **df1**

Define QUALITY (Validity):

8. Remove '<a href=' before every HTTP address in **df1** [source] using string slicing.
9. Some dog names in **df1** [name] are not proper nouns but adjectives like 'this' (row # 1120) or 'incredibly' (row # 542).
10. Change **df1** [`rating_denominator`] for (row # 313) to 10.

11. Change **df1** [`rating_numerator`] for (row # 313) to 13.

Define QUALITY (Accuracy):

12. Change **df1** [`rating_numerator`] data type from integer to float.
13. Change **df1** [`rating_numerator`] (row # 45) to 13.5.
14. Change **df1** [`rating_numerator`] (row # 1068) to 14.
    Change **df1** [`rating_denominator`] (row # 1068) to 10.
15. Change **df1** [`rating_numerator`] for (row # 340) to 9.75.
16. Change **df1** [`rating_numerator`] for (row # 695) to 9.75.
17. Change **df1** [`rating_numerator`] for (row # 763) to 11.27.
18. Change **df1** [`rating_numerator`] for (row # 1202) to 11.0.
    Change **df1** [`rating_denominator`] for (row # 1202) to 10.
19. Change **df1** [`rating_numerator`] for (row # 1712) to 11.26.
20. Divide all [`rating_numerator`] by [`rating_denominator`] and multiply by 10.

Define QUALITY (Consistency):

21. Change data type of **df1** [`timestamp`] from *string* to date *object*
22. Change data type of **df2** [tweet_id] from string to integer in line with **df1** and **image_df.**
23. Change data type of **df1** [`timestamp`] from string to datetime.
24. Change data type of **df1** [`favorite_count`] from string to integers.
25. Change data type of **df1** [`retweet_count`] from string to integers.
26. Change data type of **df1** [`in_reply_to_status_id`] from float to integer format.
27. Change data type of **df1** [`in_reply_to_user_id`] from float to integer format.

Define Tidiness (rule: each variable forms a column)

I. Insert a new column [rating**]** that takes the value from [`rating_numerator`**]** and devides it by the variable [`rating_denominator`**]**. Multiply the result by 10 to get the unique WeRateDogs rating. Drop the [`rating_numerator`**]** and [`rating_denominator`**]** columns when done.

Define Tidiness (rule: each variable forms a column)

II. Extract the [start_text_range**]** and [end_text_range**]** variables from the [display_text_range**]** column. Drop the [display_text_range] column when done.

Define Tidiness (rule: each type of observational unit forms a table)

III. Merge **df1** and **df2** using left, which uses only keys from left frame **df1**, similar to a SQL left outer join; preserve key order.

Merge **df1** and **image_df** using inner, which only uses the intersection of keys from both frames, similar to a SQL inner join; preserve the order of the left keys.

The result is one high quality and tidy master pandas dataframe **`df1_clean.`**

Storing Data

We store the cleaned dataframe **`df1_clean`** in the file **`twitter_archive_master.csv`**

`twitter_archive_master.csv` columns: (28 columns / 1991 observations)

- **[tweet_id]**: Unique Identifier of a Tweet (18-digit numerical) [Numerical]
- **[in_reply_to_status_id]**: In the context of someone else's tweet [Numeric]
- **[in_reply_to_user_id]**: Reply to a specific user [Integer]
- **[timestamp]**: Date/Time of the tweet [stored as string / change to Date Object]
- **[source]**: Origin of the tweet (e.g. web, mobile device) [Unicode String showing valid HTTP address]
- **[text]**: Text of the tweet [Unicode String]
- **[expanded_urls]**: Original URI (before URL minification) [Unicode String]
- **[rating]**: Rating numerator extracted from text. Should be on a scale from 0-10, but oftentimes is higher. [Float]
- **[name]**: Dog's name extracted from text. [Unicode String] proper noun
- **[doggo]**: Doggo stage extracted from text. [Boolean]
- **[floofer]**: Floofer stage extracted from text. [Boolean]
- **[pupper]**: Pupper stage extracted from text. [Boolean]
- **[puppo]**: Puppo stage extracted from text. [Boolean]
- **[favorite_count]**: Nullable. Indicates approximately how many times this Tweet has been liked by Twitter users. [Integer]
- **[retweet_count]**: Number of times this Tweet has been retweeted [Integer]
- **[start_text_range]**: identifies the start of the displayable content of the tweet [Integer]

- **[end_text_range]**: identifies the end of the displayable content of the tweet [Integer]
- **[jpg_url]**: Image URL [Unicode]
- **[img_num]**: image number that corresponded to the most confident prediction (numbered 1 to 4 since tweets can have up to four images) [Integer]
- **[p1]**: the algorithm's #1 prediction for the image in the tweet [Unicode]
- **[p1_conf]**: how confident the algorithm is in its #1 prediction [Float]
- **[p1_dog]**: whether or not the #1 prediction is a breed of dog [Boolean]
- **[p2]**: the algorithm's #2 prediction for the image in the tweet [Unicode]
- **[p2_conf]**: how confident the algorithm is in its #2 prediction [Float]
- **[p2_dog]**: whether or not the #2 prediction is a breed of dog [Boolean]
- **[p3]**: the algorithm's #3 prediction for the image in the tweet [Unicode]
- **[p3_conf]**: how confident the algorithm is in its #3 prediction [Float]
- **[p3_dog]**: whether or not the #3 prediction is a breed of dog [Boolean]