Wrangle Report

INTRODUCTION

WeRateDogs is a twitter account that rates people's dogs with over 3.8 million followers. The ratings usually have a denominator of 10 with a numerator usually greater than the denominator, and this special rating system should take a big credit for the popularity of WeRateDogs.

In the Jupyter notebook, we will wrangle data of WeRateDogs followed by some analysis and visualizations. This wrangling process is divided into several sections:

- Gathering Data
- Assessing Data
- Cleaning Data
- Storing Data

GATHERING DATA

There are three various sources for the data:

- WeRateDogs Twitter archive: twitter_archive_enhanced.csv
- Tweet image prediction: image-predictions.tsv (https://d17h27t6h515a5.cloudfront.net/topher/2017/August/599fd2ad_image-predictions/)
- Tweet information: ID, retweet count, and favorite count

The archive of WeRateDogs account is already available in a CSV file. The only thing required is to read the CSV file in pandas's dataframe format (we_rate_dogs).

The tweet's image prediction is generated by a neutral network, and the results are stored in a TSV file. We download the TSV tweet image prediction file using Requests python library. We handle the Response object by setting 'utf-8' encoding and write the corresponding text into a txt file named as 'tweet_image_predictions', which is stored in the 'data' folder. The txt file is read into a dataframe (tweet_image_predictions).

There is extra information can be gathered from the twitter API. The Python twitter API, tweepy, is used for simpler and more robust data extraction. For each tweet ID in the we_rate_dogs and tweet_image_predicitons tables, the entire JSON data about the tweet is downloaded by setting a correct parser (JSONParser). The tweet JSON information is stored in a Python dictionary using the tweet ID as a key. Finally, we write the JSON information into a txt file (tweet_json.txt) in which each line represents one JSON record of the tweet. We then can extract the required information, the retweet and favorite counts of the tweet. The required information is finally transformed into Pandas dataframe (tweets_data).

Assessing Data

In this section, we will assess the data to identify the issues about data quality and tidiness on three datasets: we_rate_dogs, tweet_image_predictions, and tweets_data. There are several issues are identified based on a quick data assessing, and we discuss them in two dimensions: quality and tidiness.

WE_RATE_DOGS

Quality issues:

- Only the original ratings should be considered. We should be able to identify the original ratings based on the variables like retweeted_status_user_id etc.
- there is missing data in variables: name, doggo, floofer, pupper, and puppo.
- 'None' in the dataset should be replaced by 'NaN' for better handling.
- there are wrong data types: tweet_id and timestamp
- source: turns the HTML text into something with better meaning such as 'Twitter for iPhone'

Tidiness issues:

- doggo, floofer, pupper, and puppo should be one variable

TWEET_IMAGE_PREDICTIONS.

Quality issues:

- 'tweet_id' should string and 'p1', 'p2', and 'p3' should be categorical data.

Tidiness issues:

- 'p1', 'p2', and 'p3' should be one variables, i.e., the ranking of the prediction based on neutral network
- name of column such as 'p1_conf' and 'p1_dog' are not meaningful

TWEETS_DATA

This dataset should be a subset of we_rate_dogs table since it contains the information related with the tweets in WeRateDogs.

CLEANING DATA

There are total 8 cleaning processes on the datasets. The first 6 issues are related with the we_rate_dogs table, and the last two are related with the tweet_image_predictions table.

DEFINITION 1

We want to focus on the original ratings, so we ignore the data in which its retweeted_status_user_id and in_reply_to_user_id variables are null. Then, we can drop the variables related with the retweeted_status_user_id and in_reply_to_user_id.

DEFINITION 2
The 'source' variable should be categorical and has four elements. We first extract the correct elements from the HTML text, and then cast the data types to 'category'.

DEFINITION 3
We replace all the 'None' into 'NaN' in the we_rate_dogs dataframe.

DEFINITION 4
We correct the data types of 'tweet_id' and 'timestamp' in the we_rate_dogs dataframe.

DEFINITION 5
We create a new column from columns 'doggo', 'floofer', 'pupper', and 'puppo' and label it as 'dog_stages'. Since the 'dog_stages' contains the stages of the dog, in which the variable should be ordered. Therefore, we assign an order for ‘dog_stages': 'pupper' < 'puppo' < 'doggo' < 'floofer'. Then, we drop four original columns: 'pupper', 'puppo', 'doggo', and 'floofer'.

DEFINITION 6
We merge the tweets_data table into the we_rate_dogs table.

DEFINITION 7
We correct the data types in tweet_image_predictions table.

DEFINITION 8
There are 9 columns related with the dog predictions, and we would like to stack these columns by using the prediction rank of the images. After stacking, we have two new categorical columns called 'prediction_rank' which stores the rank of the prediction and 'breed' which stores the breed of the dog predictions. Also, we rename the columns '*conf' and '*dog' into 'confidence' and 'is_dog_breed'.

STORING DATA

Finally, we store two cleaned tables into 'we_rate_dogs.sqlite' database.