Wrangle Report

28th January, 2019

1. Gathering Data
2. Assessing data
3. Cleaning data
4. Storing the Data

1. Gathering Data

As stated above Gathering the data is done in three steps. Gathering from CSV, Gathering from the link and gathering from multiple api requests. In this way I stored the data in three Data Frames df_tweet, df_images and df_api.

a) Gathering from CSV file
I used pd.read_csv() function to read the given csv named twitter-archive-enhanced.csv and saved in df_tweet.

b) Gathering from the url
As a part of this project we were given a link which had tsv containing all the details about the prediction of different types of breed of dogs which are present in the tweeted image. I used requests.get(url) to download the tsv from the remote server and then used pd.read_csv() function and saved it as df_image.

c) Gathering using api requests
This was the most challenging part of the gathering phase. Here first of all we needed to generate consumer_key, consumer_secret, access_token and access_secret from the twitter developer page. Then we passed these values as a saved variables We then generate the authentication.

```
auth = tweepy.OAuthHandler(consumer_key, consumer_secret)
auth.set_access_token(access_token, access_secret)
```

We then saved the tweet as a dictionary and eventually save them as a text file tweet_json.txt.

2. Assessing data

We then assess the data to find the number of columns present in the different data frames.

df_tweet:

tweet_id: the unique identifier for each tweet
in_reply_to_status_id: if the represented Tweet is a reply, this field will contain the integer representation of the original Tweet's ID
in_reply_to_user_id: if the represented Tweet is a reply, this field will contain the integer representation of the original Tweet's author ID
timestamp: time when this Tweet was created
source: utility used to post the Tweet, as an HTML-formatted string. e.g. Twitter for Android, Twitter for iPhone, Twitter Web Client
text: actual UTF-8 text of the status update retweeted_status_id: if the represented Tweet is a retweet, this field will contain the integer representation of the original Tweet's ID
retweeted_status_user_id: if the represented Tweet is a retweet, this field will contain the integer representation of the original Tweet's author ID
retweeted_status_timestamp: time of retweet
expanded_urls: tweet URL
rating_numerator: numerator of the rating of a dog. Note: ratings almost always greater than 10
rating_denominator: denominator of the rating of a dog. Note: ratings almost always have a denominator of 10
name: name of the dog
doggo: one of the 4 dog "stage"
floofer: one of the 4 dog "stage"
pupper: one of the 4 dog "stage"
puppo: one of the 4 dog "stage"

df_image:

tweet_id: the unique identifier for each tweet
jpg_url: dog's image URL
img_num: the image number that corresponded to the most confident prediction (numbered 1 to 4 since tweets can have up to four images)
p1: algorithm's #1 prediction for the image in the tweet
p1_conf: how confident the algorithm is in its #1 prediction
p1_dog: whether or not the #1 prediction is a breed of dog
p2: algorithm's #2 prediction for the image in the tweet
p2_conf: how confident the algorithm is in its #2 prediction
p2_dog: whether or not the #2 prediction is a breed of dog
p3: algorithm's #3 prediction for the image in the tweet

p3_conf: how confident the algorithm is in its #3 prediction
p3_dog: whether or not the #3 prediction is a breed of dog

df_api columns:

id: the unique identifier for each tweet
retweet_count: number of times this Tweet has been retweeted
favorite_count: indicates approximately how many times this Tweet has been liked by Twitter users

3. Cleaning data

a) Ensuring the Quality of the data.

- Checking all the NaNs and Handling the NaNs.
I first checked the occurrences of NaNs using the function isna() and removed them wherever possible.

- The tweet_ID is not the right data type and value in two DataFrames are of different types.
When I first tried to merge two data frames using tweet_id it did not allow merge due to the fact that
One of the tweet_id was of integer type and other was of object type.

- Erroneous data types and values for in_reply_to_status_id,in_reply_to_user_id.
In_reply_to_status_id,in_reply_to_user_id were stored as float value. There was no reason for them to be stored as a float values.

- We only want original ratings (no retweets).So the retweets shouldn't be there
Removed all the retweets and corresponding retweets rows as well.

- We only want ratings with images. Not all ratings have images.
I removed the ratings which contained no mages

- Some ratings are inaccurately picked up.
All the strings of format $/$ were picked up as rating so many a time wrong rating was picked up.

- Erroneous datatype for timestamp. Converting Object to DateTime Type.
This is common issue in nearly every dataset. Datetime is given as Object. Converting it to datetime object.

- Nulls represented as ‘None’ in columns 'name', 'doggo', 'floofer', 'pupper','puppo'.

Null values were represented as None. I replaced them with ‘’ empty string to use them in future for string concat.

- Some predictions are not dogs, there is no column for the most possible breed of a dog. Many of the prediction we predicting things other than dogs. To make meaningful sense they were eliminated

b) Tidiness

-'doggo', 'floofer', 'pupper','puppo' can be combined in one column.
I concatenated the the values in one single column and dropped the other columns. This helped me eliminate the use of redundant columns

- Combining Three DataFrames to one single DataFrame. Here rather than having 3 separate dataframes I combined all the data frames in df_comb having all clean data and meaningful columns.

4. Storing the Data

The last step was storing the data for future use/analysis. Using tocsv function in pandas i stored the DataFrame df_comb to twitter_archive_master.csv.