Wrangle report - Wrangle WeRateDogs (Twitter Data)

- Data Gathering
- Data Assessment
- Data Cleaning

Data Gathering :

Data has been gathered using three sources.

1. **WeRateDogs "Enhanced" Twitter Archived data in CSV format:**

This file contains basic tweets information for 2356 tweets.
I loaded the data from CSV file into df named twitter_archivedf usings pandas read_csv function.
There are 17 columns in this data frame that i created.

- **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: i**f 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"

2. **Image Prediction's File**

From the provided url :
'https://d17h27t6h515a5.cloudfront.net/topher/2017/August/599fd2ad_image-predictions/image-predictions.tsv' i extracted the data into img_predictions_df .
There are following columns in this data frame.
#### img_predictions_df (tweet image predictions) columns:

- **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

3. Since retweet count and favorites counts columns were missing from the archived data. I acquired it using Twitter's API. Since, I already had a Twitter developer account, i just used the authorisation credentials and used the tweepy library to connect to the twitter API. For the corresponding tweet ids in **Archived data frame** I queried the twitter API for each tweets JSON data. Meanwhile i appended each JSON data response and stored it in a DataFrame **tweets_df in three columns :**

- **Tweet id**
- **retweets**
- **Favorites**

Data Assessment :

In data Assessment following issues were identified

Quality Issues

archive df (twitter_archivedf)

- Many tweets are retweets ('retweeted_status_id', 'retweeted_status_user_id')
- some tweets are replies to other tweets ('in_reply_to_status_id', 'in_reply_to_user_id')

- Removed unnecessary columns such as retweeted_status_id, retweeted_status_user_id, retweeted_status_timestamp,in_reply_to_status_id, in_reply_to_user_id
- html tags in source column in place of utility name
- 'name', 'doggo', 'floofer', 'pupper', and 'puppo' columns have string values of "None" instead of NaN
- timestamp column value is a string object
- tweet_id is an integer
- Mismatch in dog ratings and the rating in the text.
- Calculating the actual rating

predictions df (img_predictions_df)

- The number of tweets in this df is less than the archived dataframe and tweet info dataframe.
- tweet_id has different data type

tweet df (tweets_df)

- The number of tweets in this df is lesser than the archive dataframe

Tidiness Issues

- columns 'doggo', 'floofer', 'pupper', and 'puppo' can be brought under one variable
- Assigning only one stage to each dog
- df_archive and tweet_df are part of one observational unit
- Identifying the dog breed for a given tweet

Data Cleaning:

In data cleaning all of the above issues have been addressed.

Issue: Many tweets are retweets ('retweeted_status_id', 'retweeted_status_user_id')
Solution: Filtered the data frame arch_clean using retweeted_status_id.isnull() and retweeted_status_user_id.isnull()

Issue: some tweets are replies to other tweets ('in_reply_to_status_id', 'in_reply_to_user_id')
Solution: Filtered the data frame arch_clean using in_reply_to_status_id.isnull() and in_reply_to_user_id.isnull()

Issue: Removed unnecessary columns such as retweeted_status_id, retweeted_status_user_id, retweeted_status_timestamp,in_reply_to_status_id, in_reply_to_user_id
Solution: Dropped the unnecessary column using pandas pandas.DataFrame.drop() method

Issue: html tags in source column in place of utility name
Solution: By using str.split() method on the source URL separated the texts in the url and then by applying the str.extract using regular expression extracted the source name.

Issue: name', 'doggo', 'floofer', 'pupper', and 'puppo' columns have string values of "None" instead of NaN
Solution: Used Lambda function and parsed through each row replacing “None” with NaN.

Issue: timestamp column value is a string object
Solution: type casting timestamp column to datetime using pd.to_datetime()

Issue: tweet_id is an integer
Solution: typecasting tweet_id column to object type using astype(‘str’)

Issue: Mismatch in dog ratings and the rating in the text.
Solution: Identifying the columns with denominator whose rating is not 10. Then visually identifying those tweet ids and matching the ratings in their text values with the numerator ratings and denominators ratings value, replacing both the column values where the text value doesn’t match them.

Issue: Calculating the actual rating
Solution: Calculating the actual rating based on the values in numerator rating column and denominator rating columns. Correcting the wrong rating values after checking them visually.

Issue: tweet_id has different data type in img_predictions_df
Solution: typecasting the tweet_id datatype to string datatype using .astype(str“)

Issue: columns 'doggo', 'floofer', 'pupper', and 'puppo' can be brought under one variable
Solution: created a new column “stage” in the arch_clean df and iterated three times. In the First iteration copied all the values from the different stage columns in stage. In the second stage using lambda function iterated over each row and for

each “nan” value replaced it with empty strings “ ”. And in the 3rd iteration checked if the value was “ ” string replaced it with “NaN” else kept it as it is.

Issue: Assigning only one stage to each dog
Solution: since there were more than one stage values for few rows in the stage column so using pd.DataFrame.replace() method replaced two values with the appropriate single value. Used following logic: assigning only one stage to each dog, thereby removing two stages such as 'doggopuppo','doggofloofer', 'doggopupper'.
when a dog is puppo, it means its an intermediate stage between full grown doggo and pupper, hence assigning doggopuppo as puppo. doggopupper doggo's mostly behave as if they are puppers and hence assigning doggopupper to doggo. doggofloofer can be any stage of the Dog, the name floofer means furry dog and hence assigning doggofloofer to floofer

Issue: df_archive and tweet_df are part of one observational unit
Solution: using pandas.merge() merged the two DataFrames on common tweet_ids.

Issue: Identifying the dog breed for a given tweet
Solution: Filtered out the rows in the img_prediction_clean DataFrame for NON dog breeds. when 1st prediction was true we appended the breed name and its prediction confidence in respective variables otherwise we check for the successive predictions

Data Storing:

Stored the data in .csv files as

- arch_final.csv
- imgpred_clean.csv