Project: Wrangling and Analyze Data

Wrangle Report

This report describes the wrangling process of We Rate Dogs project.

Removing errors and combining complex data sets is the process of Data Wrangling. It makes the data more accessible and easier to analyse. Data Wrangling steps, in the correct order, are:

1. Data Gathering
2. Assessing Data
3. Data Cleaning

To start with, I first imported Libraries and packages that would be need for the project.

In [1]:
```
# Import neccesary libraries and modules that will be used in the notebook

import pandas as pd
import requests
import os
import numpy as np
import tweepy
import json
from timeit import default_timer as timer
from datetime import timedelta
%matplotlib inline
from matplotlib import pyplot as plt
import seaborn as sns
from PIL import Image
from io import BytesIO
from wordcloud import WordCloud
import random
```

Data Gathering

The data for the project was pulled from 3 different sources. First, the twitter-archive-enhanced dataset for WeRateDogs was downloaded from Udacity's website and read into a pandas dataframe. The dataset contains over 5,000 tweets and some information about the tweets.

In [2]:
```
twitter_archive = pd.read_csv('twitter-archive-enhanced.csv')
```

Next, the image predictions dataset, in .tsv format, was downloaded programmatically using the requests library, then read to a second pandas dataframe, image_predict. This dataset contains information about the dogs in the pictures attached to tweets

In [3]:
```
url ='https://d17h27t6h515a5.cloudfront.net/topher/2017/August/599fd2ad_image-prediction
response=requests.get(url)
folder_name = 'C:/Users/Zion/Documents/Udacity_Wrangling'
```

```
with open(os.path.join(folder_name, url.split('/')[-1]), mode = 'wb') as file:
    file.write(response.content)
```

In [4]:
```
image_predict = pd.read_csv('image-predictions.tsv', sep= '\t')
```

Lastly, the dataset containing meta-information about the tweets in the twitter-archive dataframe was pulled using Tweepy to query Twitter's API. The pulled data in JSON format was then converted to a dataframe.

In [ ]:
```
auth = tweepy.OAuth2BearerHandler("MY_BEARER_TOKEN")
api = tweepy.API(auth, wait_on_rate_limit = True)
```

In [ ]:
```
tweets_text = 'tweet_json.txt'
count_true = 0
count_error = 0
id_error = []

start = time.time()

with open(tweets_text, 'w') as file:
    for id_of_tweet in twitter_archive['tweet_id']:
        try:
            tweet = api.get_status(id_of_tweet, tweet_mode='extended')
            json.dump(tweet._json, file)
            file.write('\n')
            count_true += 1
        except tweepy.TweepyException as e:
            count_error += 1
            id_error.append(id_of_tweet)
            continue

end = timer()

print('Successfully scrapped tweets are:' + str(count_true))
print('Unsuccessfully scrapped tweets are:' + str(count_error))
print(timedelta(seconds=end-start))
```

This will take about 20 - 30 minutes

In [5]:
```
# Load tweet data from JSON file

tweets = []
for line_of_tweet in open('tweet_json.txt', 'r'):
     tweets.append(json.loads(line_of_tweet))
```

In [6]:
```
# Load into dataframe

tweets_df = pd.DataFrame.from_dict(tweets)
tweets_df = tweets_df[['id', 'retweet_count', 'favorite_count']]
```

Assessing Data

Here, I looked for contentual and structural issues in the datasets which affects quality and tidiness respectively.

Quality issues

twitter_archive dataframe

1. Erroneous datatype for timestamp, in_reply_to_status_id and in_reply_to_user_id columns.
2. 181 Retweets (none original tweets) with unnecessary reply features columns present.
3. expanded_urls has duplicate values.
4. Some tweets do not have images attached; no value for expanded_urls.
5. Invalid Dog names like 'a', 'none', 'by', 'very' and so on.
6. Incomplete dog name; O.
7. Source column has unnecessary characters.
8. Wrong rating of 75/10

image_predict dataframe

1. Not all enteries for P1, P2 and P3 are dogs.
    - Some predicted images in image_predict do not have dogs in them.

1. tweet_id columns for the 3 dataframes should be in string format.

Tidiness issues

1. The doggo, floofer, pupper and puppo columns violate the first rule of tidiness: that each variable forms a column
2. No column for Dog breed in image_predicted dataframe.
3. rating_numerator and rating_denominator should be merged into one column.
4. twitter_archive dataframe should be merged with tweets dataframe, with unnecessary columns dropped.

Cleaning Data

In this section, I corrected the issues I documented during assessment. I first made copies of the datasets to prevent permanent data loss. I worked on each issue in 3 steps:

- Define: objective from the issue
- Code: Code to correct the issue
- Test: Check if the desired results are in place

In [36]:
```
# Make copies of original pieces of data
twitter_archive_clean = twitter_archive.copy()
image_predict_clean = image_predict.copy()
tweets_df_clean = tweets_df.copy()
```

Storing Data

Lastly, I stored the cleaned dataset for future purpose to a CSV file named "twitter_archive_master.csv".

In [73]:
```
# Store dataset
twitter_archive_cleaned.to_csv('twitter_archive_master.csv', index=False)
```