wrangle_act

December 28, 2019

1 WE NOW BEGIN OUR DATA WRANGLING JOURNEY……

[1]:
```
# We first import all the required libraries.
import pandas as pd
import numpy as np
import requests
import tweepy
from tweepy import OAuthHandler
import json
from timeit import default_timer as timer
```

1.1 GATHERING THE DATA

[2]:
```
# We now import our provided twitter archive file with the help of pandas.
dog_ratings = pd.read_csv('twitter-archive-enhanced.csv')
```

[69]:
```
pd.set_option("display.max_columns", 8)
dog_ratings.head()
```

[69]:
```
             tweet_id  in_reply_to_status_id  in_reply_to_user_id  \
0  892420643555336193                    NaN                  NaN
1  892177421306343426                    NaN                  NaN
2  891815181378084864                    NaN                  NaN
3  891689557279858688                    NaN                  NaN
4  891327558926688256                    NaN                  NaN

                   timestamp  … doggo floofer  pupper  puppo
0  2017-08-01 16:23:56 +0000  …  None    None    None   None
1  2017-08-01 00:17:27 +0000  …  None    None    None   None
2  2017-07-31 00:18:03 +0000  …  None    None    None   None
3  2017-07-30 15:58:51 +0000  …  None    None    None   None
4  2017-07-29 16:00:24 +0000  …  None    None    None   None

[5 rows x 17 columns]
```

[4]:
```
# Now we will programatically download the image prediction tsv file from␣
↪udacity's server using requests library.
```

```
url = 'https://d17h27t6h515a5.cloudfront.net/topher/2017/August/
↪599fd2ad_image-predictions/image-predictions.tsv'
response = requests.get(url)
with open("image_predictions.tsv", mode = 'wb') as outfile:
    outfile.write(response.content)
```

```
[71]: # We now read image predictions file into a df.
pd.set_option("display.max_columns", 8)
image_predictions = pd.read_csv('image_predictions.tsv', sep = '\t', encoding =␣
↪'utf-8')
image_predictions.head()
```

```
[71]:               tweet_id                                          jpg_url  \
0  666020888022790149  https://pbs.twimg.com/media/CT4udn0WwAA0aMy.jpg
1  666029285002620928  https://pbs.twimg.com/media/CT42GRgUYAA5iDo.jpg
2  666033412701032449  https://pbs.twimg.com/media/CT4521TWwAEvMyu.jpg
3  666044226329800704  https://pbs.twimg.com/media/CT5Dr8HUEAA-lEu.jpg
4  666049248165822465  https://pbs.twimg.com/media/CT5IQmsXIAAKY4A.jpg

   img_num                       p1  …  p2_dog                  p3  \
0        1  Welsh_springer_spaniel  …    True    Shetland_sheepdog
1        1                 redbone  …    True  Rhodesian_ridgeback
2        1          German_shepherd  …    True           bloodhound
3        1     Rhodesian_ridgeback  …    True   miniature_pinscher
4        1      miniature_pinscher  …    True             Doberman

    p3_conf  p3_dog
0  0.061428    True
1  0.072010    True
2  0.116197    True
3  0.222752    True
4  0.154629    True

[5 rows x 12 columns]
```

```
[6]: # Now we will gather retweet count and favourite count at minimum from twitter␣
↪API.
# Query Twitter API for each tweet in the Twitter archive and save JSON in a␣
↪text file
# These are hidden to comply with Twitter's API terms and conditions
consumer_key = 'P66aFQDJeBss9E0RraUg5CUCf'
consumer_secret = 'bUxnTS4JT5qfVJlfzdrSy8VHvFdjNhRLyLsbKlnn0qf3qdS7xm'
access_token = '760894336099688448-ECAkIkrsCGdyrHU56TEBzk3P0Pncpyk'
access_secret = 'QAnYI2sjEPsW8xyrYP4EUpbLHc5AGyGjMEuxP96nMyCM8'

auth = OAuthHandler(consumer_key, consumer_secret)
```

```
auth.set_access_token(access_token, access_secret)

api = tweepy.API(auth, wait_on_rate_limit=True)
```

[ ]:

```
```

[ ]:

```
# Query Twitter's API for JSON data for each tweet ID in the Twitter archive
tweet_ids = dog_ratings.tweet_id.values
len(tweet_ids)

count = 0
fails_dict = {}
start = timer()
# Save each tweet's returned JSON as a new line in a .txt file
with open('tweet_json.txt', 'w') as outfile:
    for tweet_id in tweet_ids:
        count += 1
        print(str(count) + ": " + str(tweet_id))
        try:
            tweet = api.get_status(tweet_id, tweet_mode='extended')
            print("Success")
            json.dump(tweet._json, outfile)
            outfile.write('\n')
        except tweepy.TweepError as e:
            print("Fail")
            fails_dict[tweet_id] = e
            pass
end = timer()
print(end - start)
print(fails_dict)

# Hiding very large output of this cell.
%%hide output
```

[ ]:

```
```

[8]:

```
# now we will open the file tweet_json.txt and read it line by line into the␣
↪data frame.
# load json data into pandas DataFrame
text_file_path = 'tweet_json.txt'
tweet_json_df = pd.read_json(text_file_path, lines = True)
# extract columns that pertain to like and favorite counts
columns_of_interest = ['id', 'retweet_count', 'favorite_count']
tweet_likes = tweet_json_df[columns_of_interest]
```

1.2 ASSESSING THE DATA

[72]:
```
# we will now visually assess dog_ratings df for Quality (Dirty Data) and␣
↪tidiness(messy data) issues.
pd.set_option("display.max_columns", 8)
dog_ratings.head()
```

[72]:
```
             tweet_id  in_reply_to_status_id  in_reply_to_user_id  \
0  892420643555336193                    NaN                  NaN
1  892177421306343426                    NaN                  NaN
2  891815181378084864                    NaN                  NaN
3  891689557279858688                    NaN                  NaN
4  891327558926688256                    NaN                  NaN

                   timestamp  … doggo floofer  pupper  puppo
0  2017-08-01 16:23:56 +0000  …  None     None    None   None
1  2017-08-01 00:17:27 +0000  …  None     None    None   None
2  2017-07-31 00:18:03 +0000  …  None     None    None   None
3  2017-07-30 15:58:51 +0000  …  None     None    None   None
4  2017-07-29 16:00:24 +0000  …  None     None    None   None

[5 rows x 17 columns]
```

[73]:
```
pd.set_option("display.max_columns", 8)
dog_ratings.sample(5)
```

[73]:
```
               tweet_id  in_reply_to_status_id  in_reply_to_user_id  \
301  836677758902222849                    NaN                  NaN
277  840370681858686976                    NaN                  NaN
184  856526610513747968           8.558181e+17         4.196984e+09
170  859074603037188101                    NaN                  NaN
902  758467244762497024                    NaN                  NaN

                     timestamp  … doggo floofer  pupper  puppo
301  2017-02-28 20:41:37 +0000  …  None     None    None   None
277  2017-03-11 01:15:58 +0000  …  None     None    None   None
184  2017-04-24 15:13:52 +0000  …  None     None    None   None
170  2017-05-01 15:58:40 +0000  …  None     None    None   None
902  2016-07-28 01:00:57 +0000  …  None     None    None   None

[5 rows x 17 columns]
```

[74]:
```
# let us visually assess image_predictionsfor quality and tidiness issues.
pd.set_option("display.max_columns", 8)
image_predictions.head()
```

```
[74]:              tweet_id                                          jpg_url  \
      0  666020888022790149  https://pbs.twimg.com/media/CT4udn0WwAA0aMy.jpg
      1  666029285002620928  https://pbs.twimg.com/media/CT42GRgUYAA5iDo.jpg
      2  666033412701032449  https://pbs.twimg.com/media/CT4521TWwAEvMyu.jpg
      3  666044226329800704  https://pbs.twimg.com/media/CT5Dr8HUEAA-lEu.jpg
      4  666049248165822465  https://pbs.twimg.com/media/CT5IQmsXIAAKY4A.jpg

         img_num                      p1  …  p2_dog                  p3  \
      0        1  Welsh_springer_spaniel  …    True    Shetland_sheepdog
      1        1                 redbone  …    True  Rhodesian_ridgeback
      2        1         German_shepherd  …    True           bloodhound
      3        1     Rhodesian_ridgeback  …    True   miniature_pinscher
      4        1      miniature_pinscher  …    True             Doberman

          p3_conf  p3_dog
      0  0.061428    True
      1  0.072010    True
      2  0.116197    True
      3  0.222752    True
      4  0.154629    True

      [5 rows x 12 columns]
```

```
[75]: pd.set_option("display.max_columns", 8)
      image_predictions.sample(5)
```

```
[75]:                 tweet_id                                          jpg_url  \
      210   669993076832759809  https://pbs.twimg.com/media/CUxLJO8U8AAu6Zu.jpg
      1914  854120357044912130  https://pbs.twimg.com/media/C9px7jyVwAAnmwN.jpg
      373   672988786805112832  https://pbs.twimg.com/media/CVbvjKqW4AA_CuD.jpg
      371   672980819271634944  https://pbs.twimg.com/media/CVbodBOUsAAb7jZ.jpg
      415   674014384960745472  https://pbs.twimg.com/media/CVqUgTIUAAUA8Jr.jpg

            img_num                        p1  …  p2_dog                      p3  \
      210         1                piggy_bank  …   False               toy_poodle
      1914        4  black-and-tan_coonhound  …    True                 bluetick
      373         1          Lakeland_terrier  …    True  wire-haired_fox_terrier
      371         1                car_mirror  …    True                   beagle
      415         1                  Pembroke  …    True               Eskimo_dog

             p3_conf  p3_dog
      210   0.086502    True
      1914  0.021762    True
      373   0.038160    True
      371   0.112397    True
      415   0.068321    True
```

```
[5 rows x 12 columns]
```

[13]:
```
# Let us visually assess tweet_likes for quality and tidiness issues.
tweet_likes.head()
```

[13]:
```
                   id  retweet_count  favorite_count
0  892420643555336193           7840           36773
1  892177421306343426           5804           31666
2  891815181378084864           3843           23848
3  891689557279858688           8000           40095
4  891327558926688256           8650           38294
```

[14]:
```
tweet_likes.sample(5)
```

[14]:
```
                      id  retweet_count  favorite_count
2209  668221241640230912            192             502
1492  691090071332753408            339            1736
1367  700167517596164096            747            2666
1080  735274964362878976           5100           13156
1784  676916996760600576           1802            2979
```

[15]:
```
# let us now also programmatically analyse data for quality issues.
dog_ratings.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2356 entries, 0 to 2355
Data columns (total 17 columns):
tweet_id                      2356 non-null int64
in_reply_to_status_id         78 non-null float64
in_reply_to_user_id           78 non-null float64
timestamp                     2356 non-null object
source                        2356 non-null object
text                          2356 non-null object
retweeted_status_id           181 non-null float64
retweeted_status_user_id      181 non-null float64
retweeted_status_timestamp    181 non-null object
expanded_urls                 2297 non-null object
rating_numerator              2356 non-null int64
rating_denominator            2356 non-null int64
name                          2356 non-null object
doggo                         2356 non-null object
floofer                       2356 non-null object
pupper                        2356 non-null object
puppo                         2356 non-null object
dtypes: float64(4), int64(3), object(10)
memory usage: 313.0+ KB
```

[16]:
```
dog_ratings[dog_ratings.tweet_id.duplicated()]
```

[16]:
```
Empty DataFrame
Columns: [tweet_id, in_reply_to_status_id, in_reply_to_user_id, timestamp,
source, text, retweeted_status_id, retweeted_status_user_id,
retweeted_status_timestamp, expanded_urls, rating_numerator, rating_denominator,
name, doggo, floofer, pupper, puppo]
Index: []
```

[17]:
```
dog_ratings.tweet_id.value_counts().sample(20)
```

[17]:
```
732732193018155009    1
732585889486888962    1
810284430598270976    1
822163064745328640    1
692417313023332352    1
888554962724278272    1
678255464182861824    1
883117836046086144    1
866334964761202691    1
670003130994700288    1
670668383499735048    1
786595970293370880    1
680940246314430465    1
684188786104872960    1
671486386088865792    1
711363825979756544    1
779124354206535695    1
705442520700944385    1
814638523311648768    1
866686824827068416    1
Name: tweet_id, dtype: int64
```

[18]:
```
image_predictions.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2075 entries, 0 to 2074
Data columns (total 12 columns):
tweet_id    2075 non-null int64
jpg_url     2075 non-null object
img_num     2075 non-null int64
p1          2075 non-null object
p1_conf     2075 non-null float64
p1_dog      2075 non-null bool
p2          2075 non-null object
p2_conf     2075 non-null float64
p2_dog      2075 non-null bool
```

```
p3          2075 non-null object
p3_conf     2075 non-null float64
p3_dog      2075 non-null bool
dtypes: bool(3), float64(3), int64(2), object(4)
memory usage: 152.1+ KB
```

[76]:
```
pd.set_option("display.max_columns", 8)
image_predictions.head()
```

[76]:
```
             tweet_id                                          jpg_url  \
0  666020888022790149  https://pbs.twimg.com/media/CT4udn0WwAA0aMy.jpg
1  666029285002620928  https://pbs.twimg.com/media/CT42GRgUYAA5iDo.jpg
2  666033412701032449  https://pbs.twimg.com/media/CT4521TWwAEvMyu.jpg
3  666044226329800704  https://pbs.twimg.com/media/CT5Dr8HUEAA-lEu.jpg
4  666049248165822465  https://pbs.twimg.com/media/CT5IQmsXIAAKY4A.jpg

   img_num                      p1  …  p2_dog                   p3  \
0        1  Welsh_springer_spaniel  …    True     Shetland_sheepdog
1        1                 redbone  …    True   Rhodesian_ridgeback
2        1         German_shepherd  …    True            bloodhound
3        1     Rhodesian_ridgeback  …    True    miniature_pinscher
4        1      miniature_pinscher  …    True              Doberman

    p3_conf  p3_dog
0  0.061428    True
1  0.072010    True
2  0.116197    True
3  0.222752    True
4  0.154629    True

[5 rows x 12 columns]
```

[20]:
```
image_predictions[image_predictions.tweet_id.duplicated()]
```

[20]:
```
Empty DataFrame
Columns: [tweet_id, jpg_url, img_num, p1, p1_conf, p1_dog, p2, p2_conf, p2_dog,
p3, p3_conf, p3_dog]
Index: []
```

[21]:
```
tweet_likes.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2333 entries, 0 to 2332
Data columns (total 3 columns):
id                  2333 non-null int64
retweet_count       2333 non-null int64
favorite_count      2333 non-null int64
dtypes: int64(3)
```

```
memory usage: 54.8 KB
```

[22]:
```
tweet_likes[tweet_likes.id.duplicated()]
```

[22]:
```
Empty DataFrame
Columns: [id, retweet_count, favorite_count]
Index: []
```

2 QUALITY ISSUES

2.1 dog_rating table

1. column 'timestamp' includes '+0000' in the end which is of no use so we can strip that away.
2. 'retweeted_status_id', 'retweeted_status_user_id', 'retweeted_status_timestamp', are not required so they can be dropped.
3. Erroneous data types for following columns 'tweet_id', 'timestamp'
4. missing 'expanded_urls'
5. certain ids are having missing names in the 'name' column.
6. Remove the rows that belong to retweets.
7. Remove elements from name column of the table that do not represent name of a dog.

2.2 image_predictions table

1. dog breed names, certain have capital and certain have small letters, so it is better to make each of them lower case
2. erroneous datatype 'tweet_id', 'image_num'

2.3 tweet_likes table

1. erroneous datatype in column 'id'

3 TIDINESS ISSUES

1. the 'doggo', 'pupper', 'foofer' and 'puppo' columns can be just a single column named 'dog_stages'
2. The dog_ratings , image_predictions and tweet_likes represent the same observational unit so they they can all be merged as a single data frame.

[77]:
```
pd.set_option("display.max_columns", 8)
dog_ratings.sample()
```

[77]:
```
               tweet_id  in_reply_to_status_id  in_reply_to_user_id  \
737  780858289093574656                    NaN                  NaN

                     timestamp  … doggo floofer  pupper  puppo
737  2016-09-27 19:54:58 +0000  …  None    None    None   None

[1 rows x 17 columns]
```

```
[ ]:
```

4 CLEANING THE DATA

```
[24]: # In order to clean the data we first of all will create the copy of all the␣
       ↪data frames in order to retain the unmodified dataset.
      dog_ratings_clean = dog_ratings.copy()
      image_pred_clean = image_predictions.copy()
      tweet_likes_clean = tweet_likes.copy()
```

```
[78]: pd.set_option("display.max_columns", 8)
      dog_ratings_clean.sample(5)
```

```
[78]:                 tweet_id  in_reply_to_status_id  in_reply_to_user_id  \
      8961  674752233200820224                    NaN                  NaN
      2970  796759840936919040                    NaN                  NaN
      1724  680085611152338944                    NaN                  NaN
      337   832397543355072512                    NaN                  NaN
      399   824796380199809024                    NaN                  NaN

                           timestamp  … rating_denominator    name  dog_stages  \
      8961  2015-12-10 00:47:23 +0000  …                 10    None       puppo
      2970  2016-11-10 17:02:03 +0000  …                 10   Romeo     floofer
      1724  2015-12-24 18:00:19 +0000  …                 10      by       doggo
      337   2017-02-17 01:13:34 +0000  …                 10   Eevee       doggo
      399   2017-01-27 01:49:15 +0000  …                 10  Bailey       doggo

            stage_name
      8961        None
      2970        None
      1724        None
      337         None
      399         None

      [5 rows x 15 columns]
```

4.1 Note - First of all we will tackle the missing data and then tidiness and quality of data respectively…….

5 Missing Data

1. Missing 'expanded_urls' in dog_ratings table.
2. Certain ids have missing names in 'name' column in dog_ratings table.

Since we cannot obtain data for the above issues, we will leave them as it is

```
[ ]:
```

[26]:
```
# Now let us test is all retweets are removed. For this there should be no non␣
↪null values in retweeted_status_id,retweeted_status_user_id,␣
↪retweeted_status_timestamp.
dog_ratings_clean.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2356 entries, 0 to 2355
Data columns (total 17 columns):
tweet_id                      2356 non-null int64
in_reply_to_status_id         78 non-null float64
in_reply_to_user_id           78 non-null float64
timestamp                     2356 non-null object
source                        2356 non-null object
text                          2356 non-null object
retweeted_status_id           181 non-null float64
retweeted_status_user_id      181 non-null float64
retweeted_status_timestamp    181 non-null object
expanded_urls                 2297 non-null object
rating_numerator              2356 non-null int64
rating_denominator            2356 non-null int64
name                          2356 non-null object
doggo                         2356 non-null object
floofer                       2356 non-null object
pupper                        2356 non-null object
puppo                         2356 non-null object
dtypes: float64(4), int64(3), object(10)
memory usage: 313.0+ KB
```

Therefore our test is complete and we havew removed all retweets.

6 Tidiness Issues

[ ]:
```

```

## 1. `dog_ratings` table : The 'doggo', 'pupper', 'floofer' and 'puppo' columns can be just a single column named 'dog_stages'

6.0.1 Define

The 'doggo', 'pupper', 'floofer' and 'puppo' columns can be just a single column named 'dog_stages'.For the purpose we will use pd.melt function of pandas

6.0.2 Code

[79]:
```
pd.set_option("display.max_columns", 8)
dog_ratings_clean.head(1)
```

```
[79]:               tweet_id  in_reply_to_status_id  in_reply_to_user_id  \
      0  892420643555336193                    NaN                  NaN

                        timestamp  … rating_denominator     name  dog_stages  \
      0  2017-08-01 16:23:56 +0000  …                 10  Phineas       doggo

        stage_name
      0       None

      [1 rows x 15 columns]
```

```
[28]: dog_ratings_clean = pd.melt(dog_ratings_clean, id_vars =  ['tweet_id',␣
       ↪'in_reply_to_status_id', 'in_reply_to_user_id', 'timestamp', 'source',␣
       ↪'text', 'retweeted_status_id', 'retweeted_status_user_id',␣
       ↪'retweeted_status_timestamp', 'expanded_urls', 'rating_numerator',␣
       ↪'rating_denominator', 'name'], var_name = 'dog_stages', value_name =␣
       ↪'stage_name')
```

```
[ ]:
```

6.0.3 TEST

```
[80]: pd.set_option("display.max_columns", 8)
      dog_ratings_clean.sample(5)
```

```
[80]:                 tweet_id  in_reply_to_status_id  in_reply_to_user_id  \
      5178  817171292965273600                    NaN                  NaN
      7676  797971864723324932                    NaN                  NaN
      48    882992080364220416                    NaN                  NaN
      5332  796125600683540480                    NaN                  NaN
      7056  666071193221509120                    NaN                  NaN

                           timestamp  … rating_denominator   name  dog_stages  \
      5178  2017-01-06 00:49:53 +0000  …                 10  Tebow      pupper
      7676  2016-11-14 01:18:12 +0000  …                 10   None       puppo
      48    2017-07-06 15:58:11 +0000  …                 10  Rusty       doggo
      5332  2016-11-08 23:01:49 +0000  …                 10   None      pupper
      7056  2015-11-16 01:52:02 +0000  …                 10   None      pupper

           stage_name
      5178       None
      7676       None
      48         None
      5332       None
      7056       None

      [5 rows x 15 columns]
```

6.1 2. The dog_ratings , image_predictions and tweet_likes represent the same observational unit so they they can all be merged as a single data frame.

6.1.1 DEFINE

The dog_ratings , image_predictions and tweet_likes represent the same observational unit so they they can all be merged as a single data frame USING pandas pd.merge function.

6.1.2 CODE

```
[30]: master_df = pd.merge(dog_ratings_clean, image_pred_clean, on = 'tweet_id')
```

```
[81]: pd.set_option("display.max_columns", 8)
      master_df.head()
```

```
[81]:              tweet_id  in_reply_to_status_id  in_reply_to_user_id  \
      0  892420643555336193                    NaN                  NaN
      1  892420643555336193                    NaN                  NaN
      2  892420643555336193                    NaN                  NaN
      3  892420643555336193                    NaN                  NaN
      4  892177421306343426                    NaN                  NaN

                        timestamp  … p2_dog        p3   p3_conf  p3_dog
      0  2017-08-01 16:23:56 +0000  …  False    banana  0.076110   False
      1  2017-08-01 16:23:56 +0000  …  False    banana  0.076110   False
      2  2017-08-01 16:23:56 +0000  …  False    banana  0.076110   False
      3  2017-08-01 16:23:56 +0000  …  False    banana  0.076110   False
      4  2017-08-01 00:17:27 +0000  …   True  papillon  0.068957    True

      [5 rows x 26 columns]
```

```
[32]: tweet_likes.head()
```

```
[32]:                    id  retweet_count  favorite_count
      0  892420643555336193           7840           36773
      1  892177421306343426           5804           31666
      2  891815181378084864           3843           23848
      3  891689557279858688           8000           40095
      4  891327558926688256           8650           38294
```

```
[33]: # We will rename column name 'id' as 'tweet id' so that it can match and be␣
       ↪merged with other data frames.
      tweet_likes_clean = tweet_likes_clean.rename(columns = {'id' : 'tweet_id'})
      tweet_likes_clean.head()
```

```
[33]:              tweet_id  retweet_count  favorite_count
      0  892420643555336193           7840           36773
      1  892177421306343426           5804           31666
```

```
2  891815181378084864           3843          23848
3  891689557279858688           8000          40095
4  891327558926688256           8650          38294
```

[82]:
```
pd.set_option("display.max_columns", 8)
master_df_comp = pd.merge(master_df, tweet_likes_clean, on= 'tweet_id')
master_df_comp.head()
```

[82]:
```
             tweet_id  in_reply_to_status_id  in_reply_to_user_id  \
0  892420643555336193                    NaN                  NaN
1  892420643555336193                    NaN                  NaN
2  892420643555336193                    NaN                  NaN
3  892420643555336193                    NaN                  NaN
4  892177421306343426                    NaN                  NaN

                  timestamp  …   p3_conf p3_dog  retweet_count  \
0  2017-08-01 16:23:56 +0000  …  0.076110  False           7840
1  2017-08-01 16:23:56 +0000  …  0.076110  False           7840
2  2017-08-01 16:23:56 +0000  …  0.076110  False           7840
3  2017-08-01 16:23:56 +0000  …  0.076110  False           7840
4  2017-08-01 00:17:27 +0000  …  0.068957   True           5804

   favorite_count
0           36773
1           36773
2           36773
3           36773
4           31666

[5 rows x 28 columns]
```

7 Quality Issues

7.1 1. dog_rating table : Column ‘timestamp’ includes ‘+0000’ in the end which is of no use so we can strip that away.

7.1.1 Define

Column ‘timestamp’ includes ‘+0000’ in the end which is of no use so we can strip that away using pandas

7.1.2 Code

[35]:
```
master_df_comp.timestamp = dog_ratings_clean.timestamp.str[:-5]
```

7.1.3 Test

[83]:
```
pd.set_option("display.max_columns", 8)
master_df_comp.head()
```

[83]:
```
             tweet_id  in_reply_to_status_id  in_reply_to_user_id  \
0  892420643555336193                    NaN                  NaN
1  892420643555336193                    NaN                  NaN
2  892420643555336193                    NaN                  NaN
3  892420643555336193                    NaN                  NaN
4  892177421306343426                    NaN                  NaN

                   timestamp  …   p3_conf p3_dog  retweet_count  \
0  2017-08-01 16:23:56 +0000  …  0.076110  False           7840
1  2017-08-01 16:23:56 +0000  …  0.076110  False           7840
2  2017-08-01 16:23:56 +0000  …  0.076110  False           7840
3  2017-08-01 16:23:56 +0000  …  0.076110  False           7840
4  2017-08-01 00:17:27 +0000  …  0.068957   True           5804

   favorite_count
0           36773
1           36773
2           36773
3           36773
4           31666

[5 rows x 28 columns]
```

[ ]:
```

```

7.2 2. dog_rating table : Remove the rows that belong to retweets.

7.2.1 Define

[84]:
```
# So in order to remove the retweets, we have to remove the rows having any␣
↪entries in the 'retweeted_status_id', 'retweeted_status_user_id',␣
↪'retweeted_status_timestamp' columns.
# Following is the code for that.
pd.set_option("display.max_columns", 8)
master_df_comp.drop(master_df_comp[master_df_comp['retweeted_status_id'].
↪notnull()== True].index, inplace = True)
master_df_comp.drop(master_df_comp[master_df_comp['retweeted_status_user_id'].
↪notnull()== True].index, inplace = True)
master_df_comp.drop(master_df_comp[master_df_comp['retweeted_status_timestamp'].
↪notnull()== True].index, inplace = True)
master_df_comp.head()
```

[84]:
```
              tweet_id  in_reply_to_status_id  in_reply_to_user_id  \
0  892420643555336193                    NaN                  NaN
1  892420643555336193                    NaN                  NaN
2  892420643555336193                    NaN                  NaN
3  892420643555336193                    NaN                  NaN
4  892177421306343426                    NaN                  NaN

                  timestamp  …   p3_conf p3_dog  retweet_count  \
0  2017-08-01 16:23:56 +0000  …  0.076110  False           7840
1  2017-08-01 16:23:56 +0000  …  0.076110  False           7840
2  2017-08-01 16:23:56 +0000  …  0.076110  False           7840
3  2017-08-01 16:23:56 +0000  …  0.076110  False           7840
4  2017-08-01 00:17:27 +0000  …  0.068957   True           5804

   favorite_count
0           36773
1           36773
2           36773
3           36773
4           31666

[5 rows x 28 columns]
```

7.3 3. Erroneous data types for following columns 'tweet_id', 'timestamp' and 'image_num'.

7.3.1 DEFINE

Erroneous data types for following columns 'tweet_id', 'timestamp' and 'image_num'.

7.3.2 CODE

[38]:
```
master_df_comp.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 7948 entries, 0 to 8243
Data columns (total 28 columns):
tweet_id                      7948 non-null int64
in_reply_to_status_id         92 non-null float64
in_reply_to_user_id           92 non-null float64
timestamp                     7948 non-null object
source                        7948 non-null object
text                          7948 non-null object
retweeted_status_id           0 non-null float64
retweeted_status_user_id      0 non-null float64
retweeted_status_timestamp    0 non-null object
expanded_urls                 7948 non-null object
rating_numerator              7948 non-null int64
```

```
rating_denominator            7948 non-null int64
name                          7948 non-null object
dog_stages                    7948 non-null object
stage_name                    7948 non-null object
jpg_url                       7948 non-null object
img_num                       7948 non-null int64
p1                            7948 non-null object
p1_conf                       7948 non-null float64
p1_dog                        7948 non-null bool
p2                            7948 non-null object
p2_conf                       7948 non-null float64
p2_dog                        7948 non-null bool
p3                            7948 non-null object
p3_conf                       7948 non-null float64
p3_dog                        7948 non-null bool
retweet_count                 7948 non-null int64
favorite_count                7948 non-null int64
dtypes: bool(3), float64(7), int64(6), object(12)
memory usage: 1.6+ MB
```

Here we see that ‘tweet_id’ and ‘img_num’ can be of object datatype as they wouldnt require any calculation and ‘timestamp’ must be a datetime datatype.

[85]:
```
master_df_comp['timestamp'] = pd.to_datetime(master_df_comp['timestamp'])
```

[40]:
```
master_df_comp.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 7948 entries, 0 to 8243
Data columns (total 28 columns):
tweet_id                      7948 non-null int64
in_reply_to_status_id         92 non-null float64
in_reply_to_user_id           92 non-null float64
timestamp                     7948 non-null datetime64[ns]
source                        7948 non-null object
text                          7948 non-null object
retweeted_status_id           0 non-null float64
retweeted_status_user_id      0 non-null float64
retweeted_status_timestamp    0 non-null object
expanded_urls                 7948 non-null object
rating_numerator              7948 non-null int64
rating_denominator            7948 non-null int64
name                          7948 non-null object
dog_stages                    7948 non-null object
stage_name                    7948 non-null object
jpg_url                       7948 non-null object
img_num                       7948 non-null int64
p1                            7948 non-null object
```

```
p1_conf                       7948 non-null float64
p1_dog                        7948 non-null bool
p2                            7948 non-null object
p2_conf                       7948 non-null float64
p2_dog                        7948 non-null bool
p3                            7948 non-null object
p3_conf                       7948 non-null float64
p3_dog                        7948 non-null bool
retweet_count                 7948 non-null int64
favorite_count                7948 non-null int64
dtypes: bool(3), datetime64[ns](1), float64(7), int64(6), object(11)
memory usage: 1.6+ MB
```

[41]:
```
master_df_comp['img_num'] = master_df_comp['img_num'].astype(object)
```

[42]:
```
master_df_comp['tweet_id'] = master_df_comp['tweet_id'].astype(object)
```

[43]:
```
master_df_comp.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 7948 entries, 0 to 8243
Data columns (total 28 columns):
tweet_id                      7948 non-null object
in_reply_to_status_id         92 non-null float64
in_reply_to_user_id           92 non-null float64
timestamp                     7948 non-null datetime64[ns]
source                        7948 non-null object
text                          7948 non-null object
retweeted_status_id           0 non-null float64
retweeted_status_user_id      0 non-null float64
retweeted_status_timestamp    0 non-null object
expanded_urls                 7948 non-null object
rating_numerator              7948 non-null int64
rating_denominator            7948 non-null int64
name                          7948 non-null object
dog_stages                    7948 non-null object
stage_name                    7948 non-null object
jpg_url                       7948 non-null object
img_num                       7948 non-null object
p1                            7948 non-null object
p1_conf                       7948 non-null float64
p1_dog                        7948 non-null bool
p2                            7948 non-null object
p2_conf                       7948 non-null float64
p2_dog                        7948 non-null bool
p3                            7948 non-null object
p3_conf                       7948 non-null float64
p3_dog                        7948 non-null bool
```

```
retweet_count                 7948 non-null int64
favorite_count                7948 non-null int64
dtypes: bool(3), datetime64[ns](1), float64(7), int64(4), object(13)
memory usage: 1.6+ MB
```

7.3.3 TEST

```
[44]: master_df_comp.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 7948 entries, 0 to 8243
Data columns (total 28 columns):
tweet_id                      7948 non-null object
in_reply_to_status_id         92 non-null float64
in_reply_to_user_id           92 non-null float64
timestamp                     7948 non-null datetime64[ns]
source                        7948 non-null object
text                          7948 non-null object
retweeted_status_id           0 non-null float64
retweeted_status_user_id      0 non-null float64
retweeted_status_timestamp    0 non-null object
expanded_urls                 7948 non-null object
rating_numerator              7948 non-null int64
rating_denominator            7948 non-null int64
name                          7948 non-null object
dog_stages                    7948 non-null object
stage_name                    7948 non-null object
jpg_url                       7948 non-null object
img_num                       7948 non-null object
p1                            7948 non-null object
p1_conf                       7948 non-null float64
p1_dog                        7948 non-null bool
p2                            7948 non-null object
p2_conf                       7948 non-null float64
p2_dog                        7948 non-null bool
p3                            7948 non-null object
p3_conf                       7948 non-null float64
p3_dog                        7948 non-null bool
retweet_count                 7948 non-null int64
favorite_count                7948 non-null int64
dtypes: bool(3), datetime64[ns](1), float64(7), int64(4), object(13)
memory usage: 1.6+ MB
```

7.4 4. 'retweeted_status_id', 'retweeted_status_user_id', 'retweeted_status_timestamp', are not required so they can be dropped.

7.4.1 DEFINE

'retweeted_status_id', 'retweeted_status_user_id', 'retweeted_status_timestamp', are not required so they can be dropped since the retweets have now been removed.

7.4.2 CODE

[45]:
```
master_df_comp.drop(columns= ['retweeted_status_user_id',␣
↪'retweeted_status_id', 'retweeted_status_timestamp'], inplace = True)
```

7.5 5. dog breed names, certain have capital and certain have small letters, so it is better to make each of them lower case

7.5.1 DEFINE

dog breed names, certain have capital and certain have small letters, so it is better to make each of them lower case in 'p1', 'p2', 'p3' columns.

7.5.2 CODE

[86]:
```
pd.set_option("display.max_columns", 8)
master_df_comp.head(1)
```

[86]:
```
             tweet_id  in_reply_to_status_id  in_reply_to_user_id  \
0  892420643555336193                    NaN                  NaN

                   timestamp  …  p3_conf p3_dog  retweet_count  \
0 2017-08-01 16:23:56+00:00  …  0.07611  False           7840

   favorite_count
0           36773

[1 rows x 28 columns]
```

[47]:
```
master_df_comp['p1'] = master_df_comp['p1'].str.lower()
```

[48]:
```
master_df_comp['p2'] = master_df_comp['p2'].str.lower()
```

[49]:
```
master_df_comp['p3'] = master_df_comp['p3'].str.lower()
```

7.5.3 TEST

[87]:
```
pd.set_option("display.max_columns", 8)
master_df_comp.sample(5)
```

[87]:
```
                 tweet_id  in_reply_to_status_id  in_reply_to_user_id  \
6837  672523490734551040                    NaN                  NaN
3622  733460102733135873                    NaN                  NaN
7058  671362598324076544                    NaN                  NaN
3743  727524757080539137                    NaN                  NaN
4408  705591895322394625                    NaN                  NaN

                     timestamp  …   p3_conf p3_dog  retweet_count  \
6837 2015-12-03 21:11:09+00:00  …  0.061596   True            165
3622 2016-05-20 00:51:30+00:00  …  0.017379  False           1298
7058 2015-11-30 16:18:11+00:00  …  0.077301  False            291
3743 2016-05-03 15:46:33+00:00  …  0.003941   True           1237
4408 2016-03-04 03:13:11+00:00  …  0.035638   True           1164

      favorite_count
6837             621
3622            4251
7058            1083
3743            4518
4408            3215

[5 rows x 28 columns]
```

[ ]:
```

```

7.6 6.Remove elements from name column of the table that do not represent name of a dog.

7.6.1 DEFINE

Remove elements from name column of the table that do not represent name of a dog

7.6.2 CODE

[51]:
```
master_df_comp['name'] = master_df_comp['name'].replace( to_replace = ['a',␣
↪'an' , 'the', 'by', 'very'], value = 'None')
```

[52]:
```
master_df_comp['name'].sample(25)
```

[52]:
```
2324       None
3477     Nollie
6824        Taz
6808     Norman
3330      Lenox
2189       Rory
8194       None
2202       Dale
1425      Buddy
```

```
378         Zoey
6326        None
6198        None
605         None
1536         Sky
6399     Tedders
3683        None
6428        None
4763      Cassie
560         None
318         None
1562        None
43          Koda
7989     Churlie
3695        None
4876       Brian
Name: name, dtype: object
```

7.6.3 TEST

[53]:
```
master_df_comp['name']
```

[53]:
```
0         Phineas
1         Phineas
2         Phineas
3         Phineas
4           Tilly
5           Tilly
6           Tilly
7           Tilly
8          Archie
9          Archie
10         Archie
11         Archie
12          Darla
13          Darla
14          Darla
15          Darla
16       Franklin
17       Franklin
18       Franklin
19       Franklin
20           None
21           None
22           None
23           None
24            Jax
```

```
25             Jax
26             Jax
27             Jax
28            None
29            None
          …
8214          None
8215          None
8216          None
8217          None
8218          None
8219          None
8220          None
8221          None
8222          None
8223          None
8224          None
8225          None
8226          None
8227          None
8228          None
8229          None
8230          None
8231          None
8232          None
8233          None
8234          None
8235          None
8236          None
8237          None
8238          None
8239          None
8240          None
8241          None
8242          None
8243          None
Name: name, Length: 7948, dtype: object
```

[54]:
```
# Let us now store our cleaned dataframe in a csv file
master_df_comp.to_csv('twitter_archive_master.csv')
```

7.7 OUR DATASET IS NOW CLEANED. LET US NOW TURN TOWARDS VISUALISATION AND INSIGHTS

8 VISUALISATION

[88]:
```
pd.set_option("display.max_columns", 8)
master_df_comp.head()
```

[88]:
```
             tweet_id  in_reply_to_status_id  in_reply_to_user_id  \
0  892420643555336193                    NaN                  NaN
1  892420643555336193                    NaN                  NaN
2  892420643555336193                    NaN                  NaN
3  892420643555336193                    NaN                  NaN
4  892177421306343426                    NaN                  NaN

                  timestamp  …   p3_conf p3_dog  retweet_count  \
0 2017-08-01 16:23:56+00:00  …  0.076110  False           7840
1 2017-08-01 16:23:56+00:00  …  0.076110  False           7840
2 2017-08-01 16:23:56+00:00  …  0.076110  False           7840
3 2017-08-01 16:23:56+00:00  …  0.076110  False           7840
4 2017-08-01 00:17:27+00:00  …  0.068957   True           5804

   favorite_count
0           36773
1           36773
2           36773
3           36773
4           31666

[5 rows x 28 columns]
```

[56]:
```
master_df_comp.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 7948 entries, 0 to 8243
Data columns (total 25 columns):
tweet_id                 7948 non-null object
in_reply_to_status_id    92 non-null float64
in_reply_to_user_id      92 non-null float64
timestamp                7948 non-null datetime64[ns]
source                   7948 non-null object
text                     7948 non-null object
expanded_urls            7948 non-null object
rating_numerator         7948 non-null int64
rating_denominator       7948 non-null int64
name                     7948 non-null object
dog_stages               7948 non-null object
stage_name               7948 non-null object
```

```
jpg_url                    7948 non-null object
img_num                    7948 non-null object
p1                         7948 non-null object
p1_conf                    7948 non-null float64
p1_dog                     7948 non-null bool
p2                         7948 non-null object
p2_conf                    7948 non-null float64
p2_dog                     7948 non-null bool
p3                         7948 non-null object
p3_conf                    7948 non-null float64
p3_dog                     7948 non-null bool
retweet_count              7948 non-null int64
favorite_count             7948 non-null int64
dtypes: bool(3), datetime64[ns](1), float64(5), int64(4), object(12)
memory usage: 1.4+ MB
```

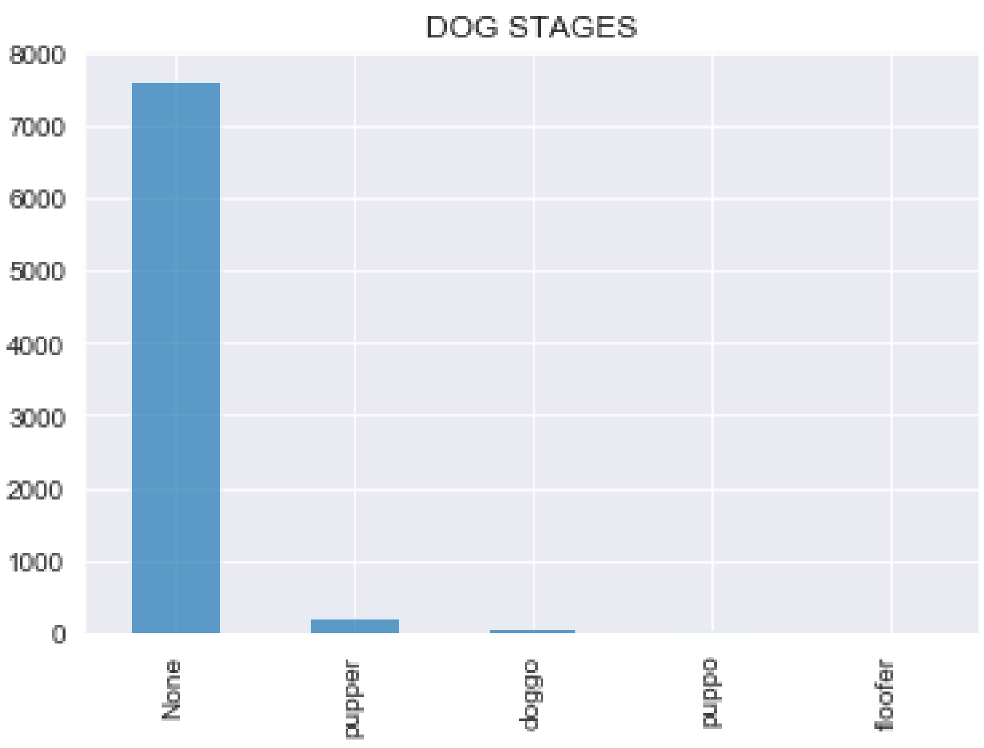
[57]:
```
stage_max = master_df_comp['stage_name'].value_counts()
stage_max_df = pd.DataFrame(stage_max)
stage_max_df
```

[57]:
```
         stage_name
None           7632
pupper          212
doggo            73
puppo            23
floofer           8
```

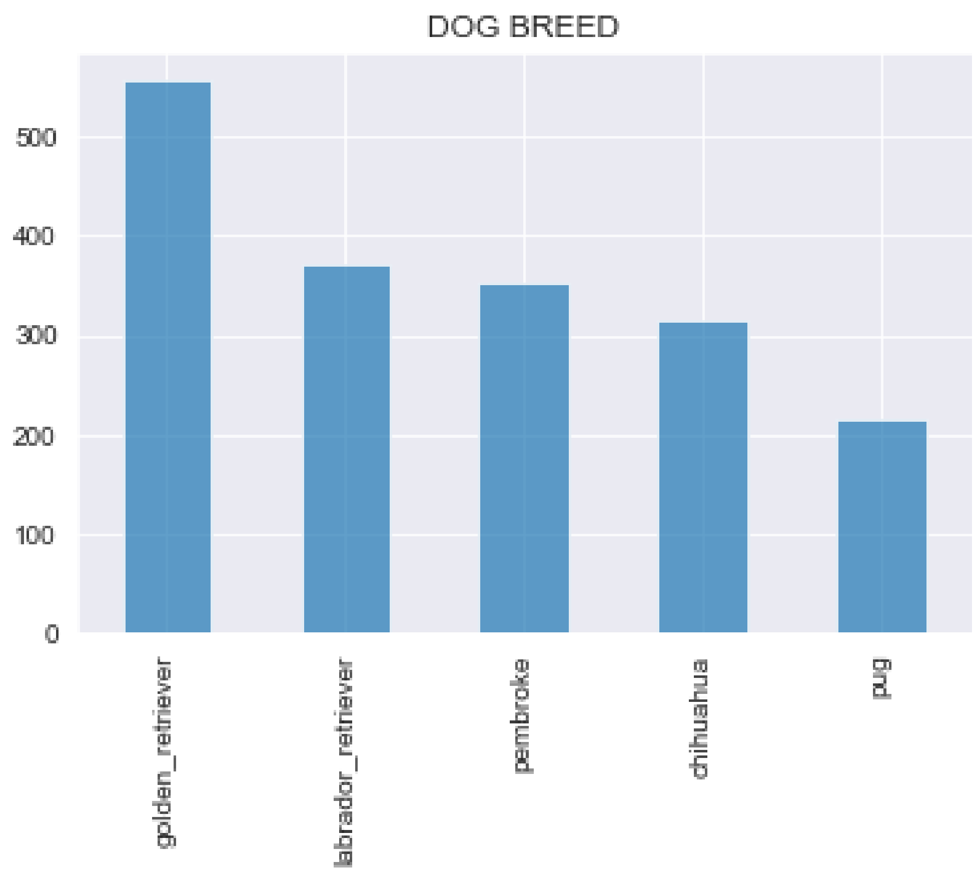
[58]:
```
breed_max = master_df_comp['p1'].value_counts().head()
breed_max_df = pd.DataFrame(breed_max)
breed_max_df
```

[58]:
```
                     p1
golden_retriever    556
labrador_retriever  372
pembroke            352
chihuahua           316
pug                 216
```

[62]:
```
%matplotlib inline
import seaborn as sns
sns.set_style('darkgrid')
stage_max_df.plot.bar( y ='stage_name', alpha = 0.7, title = 'DOG STAGES',␣
 ↪legend = '');
```



```
[63]: breed_max_df.plot.bar(y =  'p1', alpha= 0.7, legend = '', title = 'DOG BREED');
```



8.1 INSIGHTS

1. As we see from the above **visualisation no.1** that data wrangling specially for dog stages made it so convenient to see that most of the dogs could not be identified with a stage while maximum of the dogs whose stages could be defined lied in the pupper stage.
2. From the **second visualisation** we get that by **combining image_predicton table data together** made it easy to identify firdt prediction(p1) of the dog breed that came out to be golden retriever followed by labrador retriever and pembroke.
3. We see that by **removing the retweets** we are able to get a true picture of the data analysed as with retweets the counts would have been biased. we have also removed certain unrequired columns[‘retweeted_status_id’, ‘retweeted_status_user_id’, ‘retweeted_status_timestamp’] that contained retweet information.
4. We have also **altered the timestamp** to make it more presentable and also **changed the datatyoe od certain columns** [‘tweet_id’, ‘timestamp’ and ‘image_num’] which would make the work of data analysts who further utilise the dataset easier.

I believe that above wrangling and insights are in accordance with the project motivation and have formed a clean dataset.