```
# import necessary libraries
import pandas as pd
import numpy as np
import json
import matplotlib.pyplot as plt
import seaborn as sns
from google.colab import drive
import requests
from bs4 import BeautifulSoup
import os
import tweepy
import time
import glob
import ast
from datetime import date
%matplotlib inline
```

GATHERING DATA

Reading the twitter_archive_enhance.csv file into dataframe

```
#read the twitter archive enhanced csv file
twitter_enhanced = pd.read_csv('sample_data/twitter-archive-enhanced.csv')
```

Downloading the image_predictions.tsv using the requests library

create folder

```
# create the folder name
folder_name = 'data_wrangling_project'

#set a condition that executes if folder name doesn't exist
if not os.path.exists(folder_name):
  os.makedirs(folder_name)
```

retrieve data

```
# specify the url
url = 'https://d17h27t6h515a5.cloudfront.net/topher/2017/August/599fd2ad_image-predictions/im
```

```
# retreive data from the url
response = requests.get(url)
```

▼ **open** file, **write** to file and **read** the extracted file

```
#get content of the tsv file
with open(os.path.join(folder_name, url.split('/')[-1]), 'wb') as image_predictions:

  #write the content to image_predictions
  image_predictions.write(response.content)

  #read the file
  image_predictions.read

  #print the result
  print(image_predictions)
```

```
    <_io.BufferedWriter name='data_wrangling_project/image-predictions.tsv'>
```

▼ **verify** that the file is in the folder

```
# verify the directory to ensure the file was extracted successfully
os.listdir(folder_name)
```

```
    ['image-predictions.tsv']
```

▼ **read** the **image-predictions.tsv** file into a pandas dataframe

```
# read the tsv file into dataframe: image_predictions
image_predictions = pd.read_csv('data_wrangling_project/image-predictions.tsv', sep='\t')
```

```
###
```

▼ Query the twitter API for the **tweet ID**, **retweet count** and **favorite count**

▼ input your consumer keys and access keys and authenticate them

```
# input your consumers keys and access tokens
consumer_key = 'consumer key'
consumer_secret = 'consumer secret key'
access_token = 'access token'
access_secret = 'access token secret'
```

```
# authenticate your keys
auth = tweepy.OAuthHandler(consumer_key, consumer_secret)
auth.set_access_token(access_token, access_secret)

#Create an API class
api = tweepy.API(auth, wait_on_rate_limit=True, wait_on_rate_limit_notify=True)
```

access twitter data through api and store in a list

```
#accessing twitter data through api
#create a list  to store the tweet data
tweet_list = []

#error list to store missing tweet id
tweet_list_error = []

#initialize start time
start_time = time.time()

# save ech tweet data queried in a txt file as a new line
with open('tweet_json.txt', mode='w') as tweet_json :
  # use for loop to iterate through the list of ids
  for id in twitter_enhanced['tweet_id'].values:

  # use the try-except error handler to cater for missing tweet ids
    try:
      tweet_1 = api.get_status(id, tweet_mode='extended')._json
      #tweet_2 = api.get_status(id, tweet_mode='extended')._json
      #print('success')
      json.dump(tweet_1, tweet_json)

      #write the file line by line
      tweet_json.write('\n')

      #get the necessary info from the tweet json file
      favorites = tweet_1['favorite_count'] # How many favorites the tweet had
      retweets = tweet_1['retweet_count'] # Count of the retweet
      user_favourites = tweet_1['user']['favourites_count'] # How many favorites the user had
      date_time = tweet_1['created_at'] # The date and time of the creation

      #append the info to a list
      tweet_list.append({'tweet_id': int(id),
                         'favorites': int(favorites),
                         'retweets': int(retweets),
                         'user_favourites': int(user_favourites),
                         'date_time': pd.to_datetime(date_time)})

    except tweepy.TweepError as e:
```

```
        tweet_list_error.append(id)

#initialize end time
end_time = time.time()
#display the execution time
display('time for execution is : {}'.format(end_time-start_time))
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
/usr/local/lib/python3.7/dist-packages/urllib3/connectionpool.py in _make_request(self,
conn, method, url, timeout, chunked, **httplib_request_kw)
    376             try:  # Python 2.7, use buffering of HTTP responses
--> 377                 httplib_response = conn.getresponse(buffering=True)
    378             except TypeError:  # Python 3

TypeError: getresponse() got an unexpected keyword argument 'buffering'

During handling of the above exception, another exception occurred:

KeyboardInterrupt                         Traceback (most recent call last)
```

13 frames

```
/usr/lib/python3.7/ssl.py in read(self, len, buffer)
    927         try:
    928             if buffer is not None:
--> 929                 return self._sslobj.read(len, buffer)
    930             else:
    931                 return self._sslobj.read(len)

KeyboardInterrupt:
```

read the file data into a dataframe

```
tweet_1['full_text']
```

```
'Here we have a Japanese Irish Setter. Lost eye in Vietnam (?). Big fan of relaxing on
stair  8/10 would pet https://t.co/BLDqew2Iij'
```

```
#for text in tweet_1['full_text']:
 # text = tweet_1['full_text'][-38:-37]
  #text_2 = tweet_1['full_text'][-36:-34]
  #tweet_list.append({
   #   'rating_numerator':int(text),
    #  'rating_denominator':int(text_2)
 # })


# convert the tweet_list to dataframe
json_tweets = pd.DataFrame(tweet_list, columns = ['tweet_id', 'date_time', 'favorites', 'retw
                                                  'user_favourites'])
```

```
# save the dataframe as csv file
json_tweets.to_csv('sample_data/json_tweets.csv', index = False)
```

```
# read the csv file
json_tweets_df = pd.read_csv('sample_data/json_tweets.csv')
```

Double-click (or enter) to edit

ACCESSING

Accessing visually

> we will be looking at the twitter_enhanced_df dataset first

```
#make a copy of each file
twitter_enhanced_df = twitter_enhanced.copy()
image_predictions_df = image_predictions.copy()
json_tweet_df = json_tweets_df.copy()
```

```
#print the head of twitter_enhanced_df
twitter_enhanced_df
```

	tweet_id	in_reply_to_status_id	in_reply_to_user_id	timestamp	
0	892420643555336193	NaN	NaN	2017-08-01 16:23:56 +0000	href="h
1	892177421306343426	NaN	NaN	2017-08-01 00:17:27 +0000	href="h
2	891815181378084864	NaN	NaN	2017-07-31 00:18:03 +0000	href="h
3	891689557279858688	NaN	NaN	2017-07-30 15:58:51 +0000	href="h
4	891327558926688256	NaN	NaN	2017-07-29 16:00:24 +0000	href="h
...	...	...	...	...	
2351	666049248165822465	NaN	NaN	2015-11-16 00:24:50 +0000	href="h
2352	666044226329800704	NaN	NaN	2015-11-16 00:04:52	href="h

- Column **in_reply_to_status_id, in_reply_to_user_id, retweeted_status_id retweeted_status_user_id and retweeted_status_timestamp** contains null values
- Column headers **doggo, fluffer, pupper and puppo** are values not headers

```
# sample the data
twitter_enhanced_df.sample(50)
```

	tweet_id	in_reply_to_status_id	in_reply_to_user_id	timestamp	
785	775085132600442880	NaN	NaN	2016-09-11 21:34:30 +0000	href="h
1104	735137028879360001	NaN	NaN	2016-05-24 15:55:00 +0000	href="h
1350	704113298707505153	NaN	NaN	2016-02-29 01:17:46 +0000	href="h
395	825535076884762624	NaN	NaN	2017-01-29 02:44:34 +0000	href="h
643	793195938047070209	NaN	NaN	2016-10-31 21:00:23 +0000	href="h
1308	707297311098011648	NaN	NaN	2016-03-08 20:09:54 +0000	href="h
263	842535590457499648	NaN	NaN	2017-03-17 00:38:32 +0000	href="h
2239	667937095915278337	NaN	NaN	2015-11-21 05:26:27 +0000	href="h
1152	725786712245440512	NaN	NaN	2016-04-28 20:40:11 +0000	href="h
271	841077006473256960	NaN	NaN	2017-03-13 00:02:39 +0000	href="h
1136	728387165835677696	NaN	NaN	2016-05-06 00:53:27 +0000	href="h
890	750566828574212096	NaN	NaN	2016-07-31	href="h

890	759566828574212096	NaN	NaN	01:50:18 +0000	href="h
696	786664955043049472	NaN	NaN	2016-10-13 20:28:35 +0000	href="h
452	818614493328580609	NaN	NaN	2017-01-10 00:24:38 +0000	href="h
1652	683481228088049664	NaN	NaN	2016-01-03 02:53:17 +0000	href="h
1093	737310737551491075	NaN	NaN	2016-05-30 15:52:33 +0000	href="h
992	748692773788876800	NaN	NaN	2016-07-01 01:40:41 +0000	href="h
2278	667435689202614272	NaN	NaN	2015-11-19 20:14:03 +0000	href="h
2303	666996132027977728	NaN	NaN	2015-11-18 15:07:24 +0000	href="h
95	873697596434513921	NaN	NaN	2017-06-11 00:25:14 +0000	href="h
1663	682808988178739200	6.827884e+17	4.196984e+09	2016-01-01 06:22:03 +0000	href="h
217	850380195714523136	NaN	NaN	2017-04-07 16:10:12 +0000	href="h
1990	672640509974827008	NaN	NaN	2015-12-04 04:56:09 +0000	href="h
				2017-02-16	

338	832369877331693569	NaN	NaN	23:23:38 +0000	href="h
786	774757898236878852	NaN	NaN	2016-09-10 23:54:11 +0000	href="h
1612	685321586178670592	NaN	NaN	2016-01-08 04:46:13 +0000	href="h
1573	687494652870668288	NaN	NaN	2016-01-14 04:41:12 +0000	href="h
1081	738885046782832640	NaN	NaN	2016-06-04 00:08:17 +0000	href="h
865	762316489655476224	NaN	NaN	2016-08-07 15:56:28 +0000	href="h
417	822489057087389700	NaN	NaN	2017-01-20 17:00:46 +0000	href="h
2177	669037058363662336	NaN	NaN	2015-11-24 06:17:19 +0000	href="h
1237	712309440758808576	NaN	NaN	2016-03-22 16:06:19 +0000	href="h
985	749075273010798592	NaN	NaN	2016-07-02 03:00:36 +0000	
2174	669216679721873412	NaN	NaN	2015-11-24 18:11:04 +0000	href="h
680	788908386943430656	NaN	NaN	2016-10-20 01:03:11 +0000	href="h
				2016-04-	

1205	715928423106027520	NaN	NaN	01 15:46:52 +0000	href="h
912	757596066325864448	NaN	NaN	2016-07-25 15:19:12 +0000	href="h
2307	666826780179869698	NaN	NaN	2015-11-18 03:54:28 +0000	href="h
1267	709566166965075968	NaN	NaN	2016-03-15 02:25:31 +0000	href="h
1299	707738799544082433	NaN	NaN	2016-03-10 01:24:13 +0000	
1251	710997087345876993	NaN	NaN	2016-03-19 01:11:29 +0000	href="h
1087	738156290900254721	NaN	NaN	2016-06-01 23:52:28 +0000	href="h
855	764857477905154048	NaN	NaN	2016-08-14 16:13:27 +0000	href="h
1836	676098748976615425	NaN	NaN	2015-12-13 17:57:57 +0000	href="h
1193	717537687239008257	NaN	NaN	2016-04-06 02:21:30 +0000	href="h
2255	667773195014021121	NaN	NaN	2015-11-20 18:35:10 +0000	
1881	675003128568291329	NaN	NaN	2015-12-10 17:24:21 +0000	href="h
				2016-02-	

1393	700029284593901568	NaN	NaN	17 18:49:22 +0000	href="h
52	882045870035918850	NaN	NaN	2017-07-04	href="h

- invalid dog names like *a* and value *None* present

Accessing programmatically

> accessing twitter_enhanced_df data set programmatically

```
# Get the dataset information
twitter_enhanced_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2356 entries, 0 to 2355
Data columns (total 17 columns):
 #   Column                      Non-Null Count  Dtype  
---  ------                      --------------  -----  
 0   tweet_id                    2356 non-null   int64  
 1   in_reply_to_status_id       78 non-null     float64
 2   in_reply_to_user_id         78 non-null     float64
 3   timestamp                   2356 non-null   object 
 4   source                      2356 non-null   object 
 5   text                        2356 non-null   object 
 6   retweeted_status_id         181 non-null    float64
 7   retweeted_status_user_id    181 non-null    float64
 8   retweeted_status_timestamp  181 non-null    object 
 9   expanded_urls               2297 non-null   object 
 10  rating_numerator            2356 non-null   int64  
 11  rating_denominator          2356 non-null   int64  
 12  name                        2356 non-null   object 
 13  doggo                       2356 non-null   object 
 14  floofer                     2356 non-null   object 
 15  pupper                      2356 non-null   object 
 16  puppo                       2356 non-null   object 
dtypes: float64(4), int64(3), object(10)
memory usage: 313.0+ KB
```

- erroneous data types: tweet_id should of type string, timestamp should be datetime type
- presence of duplicate rows since we have retweet information(retweeted_status_id, retweeted_status_timestamp, retweeted_status_user_id). We are only interested in the original tweets

```
# get the summary statistics
twitter_enhanced_df.describe()
```

	tweet_id	in_reply_to_status_id	in_reply_to_user_id	retweeted_status_id	r
count	2.356000e+03	7.800000e+01	7.800000e+01	1.810000e+02	
mean	7.427716e+17	7.455079e+17	2.014171e+16	7.720400e+17	
std	6.856705e+16	7.582492e+16	1.252797e+17	6.236928e+16	
min	6.660209e+17	6.658147e+17	1.185634e+07	6.661041e+17	
25%	6.783989e+17	6.757419e+17	3.086374e+08	7.186315e+17	
50%	7.196279e+17	7.038708e+17	4.196984e+09	7.804657e+17	
75%	7.993373e+17	8.257804e+17	4.196984e+09	8.203146e+17	
max	8.924206e+17	8.862664e+17	8.405479e+17	8.874740e+17	

- rating denominator min value of 0 is incorresct. it should be 10
- outliers in the rating_numerator e.g minimum value of 10 and maximum of 1776. there are other apart from these.

```
# check for rows where rating numerator > 17
twitter_enhanced_df[twitter_enhanced_df['rating_numerator'] > 17]
```

	tweet_id	in_reply_to_status_id	in_reply_to_user_id	timestamp	
188	855862651834028034	8.558616e+17	1.943518e+08	2017-04-22 19:15:32 +0000	href
189	855860136149123072	8.558585e+17	1.361572e+07	2017-04-22 19:05:32 +0000	href
290	838150277551247360	8.381455e+17	2.195506e+07	2017-03-04 22:12:52 +0000	href
313	835246439529840640	8.352460e+17	2.625958e+07	2017-02-24 21:54:03 +0000	href
340	832215909146226688	NaN	NaN	2017-02-16 13:11:49 +0000	href
433	820690176645140481	NaN	NaN	2017-01-15 17:52:40 +0000	href
516	810984652412424192	NaN	NaN	2016-12-19 23:06:23 +0000	href
695	786709082849828864	NaN	NaN	2016-10-13 23:23:56 +0000	href
763	778027034220126208	NaN	NaN	2016-09-20 00:24:34 +0000	href
902	758467244762497024	NaN	NaN	2016-07-28 01:00:57 +0000	href
979	749981277374128128	NaN	NaN	2016-07-04 15:00:45 +0000	href="h
1120	731156023742988288	NaN	NaN	2016-05-13	href

1120	731156023742988288	NaN	NaN	16:15:54 +0000	href
1202	716439118184652801	NaN	NaN	2016-04-03 01:36:11 +0000	href
1228	713900603437621249	NaN	NaN	2016-03-27 01:29:02 +0000	href
1254	710658690886586372	NaN	NaN	2016-03-18 02:46:49 +0000	href
1274	709198395643068416	NaN	NaN	2016-03-14 02:04:08 +0000	href
1351	704054845121142784	NaN	NaN	2016-02-28 21:25:30 +0000	href
1433	697463031882764288	NaN	NaN	2016-02-10 16:51:59 +0000	href
1634	684225744407494656	6.842229e+17	4.196984e+09	2016-01-05 04:11:44	href

```
#check for duplicate rows or values
sum(twitter_enhanced_df.duplicated())
```

```
0
```

```
#check the shape
twitter_enhanced_df.shape
```

```
(2356, 17)
```

Accessing the image_predictions_df dataset

> Accessing visually

```
# print head of the image_predictions_df
image_predictions_df.head(5)
```

	tweet_id	jpg_url	img_num	
0	666020888022790149	https://pbs.twimg.com/media/CT4udn0WwAA0aMy.jpg	1	Welsh_spri
1	666029285002620928	https://pbs.twimg.com/media/CT42GRgUYAA5iDo.jpg	1	
2	666033412701032449	https://pbs.twimg.com/media/CT4521TWwAEvMyu.jpg	1	Germ
3	666044226329800704	https://pbs.twimg.com/media/CT5Dr8HUEAA-lEu.jpg	1	Rhodesia
4	666049248165822465	https://pbs.twimg.com/media/CT5IQmsXIAAKY4A.jpg	1	miniat

- p1, p2, p3 in the image prdiction dataset having underscore

Accesssing programmatically

```
# check the dataset info
image_predictions_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2075 entries, 0 to 2074
Data columns (total 12 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   tweet_id  2075 non-null   int64  
 1   jpg_url   2075 non-null   object 
 2   img_num   2075 non-null   int64  
 3   p1        2075 non-null   object 
 4   p1_conf   2075 non-null   float64
 5   p1_dog    2075 non-null   bool   
 6   p2        2075 non-null   object 
 7   p2_conf   2075 non-null   float64
 8   p2_dog    2075 non-null   bool   
 9   p3        2075 non-null   object 
 10  p3_conf   2075 non-null   float64
 11  p3_dog    2075 non-null   bool   
dtypes: bool(3), float64(3), int64(2), object(4)
memory usage: 152.1+ KB
```

```
#check for indepth description
image_predictions_df.describe()
```

	tweet_id	img_num	p1_conf	p2_conf	p3_conf
count	2.075000e+03	2075.000000	2075.000000	2.075000e+03	2.075000e+03
mean	7.384514e+17	1.203855	0.594548	1.345886e-01	6.032417e-02
std	6.785203e+16	0.561875	0.271174	1.006657e-01	5.090593e-02
min	6.660209e+17	1.000000	0.044333	1.011300e-08	1.740170e-10
25%	6.764835e+17	1.000000	0.364412	5.388625e-02	1.622240e-02
50%	7.119988e+17	1.000000	0.588230	1.181810e-01	4.944380e-02

```
#check for duplicates
image_predictions_df.duplicated().sum()
```

```
0
```

```
image_predictions_df[image_predictions_df.p1 =='redbone' ]
```

	tweet_id	jpg_url	img_num	p
1	666029285002620928	https://pbs.twimg.com/media/CT42GRgUYAA5iDo.jpg	1	redbor
155	668815180734689280	https://pbs.twimg.com/media/CUgb21RXIAAlff7.jpg	1	redbor
278	670995969505435648	https://pbs.twimg.com/media/CU_bRIEWcAAUVC7.jpg	1	redbor
1191	739932936087216128	https://pbs.twimg.com/media/CkTFEe-W0AA90m1.jpg	1	redbor
1276	750071704093859840	https://pbs.twimg.com/media/CmjKOzVWcAAQN6w.jpg	2	redbor
1386	766069199026450432	https://pbs.twimg.com/media/CqGf3xaXYAEh3ak.jpg	1	redbor

```
# check for prediction greater than 0.5
image_predictions_df[image_predictions_df.p1_conf >= 0.5]
```

	tweet_id	jpg_url	img_num	
1	666029285002620928	https://pbs.twimg.com/media/CT42GRgUYAA5iDo.jpg	1	
2	666033412701032449	https://pbs.twimg.com/media/CT4521TWwAEvMyu.jpg	1	G

```
# check for prediction greater than 0.5 for p2_conf
image_predictions_df[image_predictions_df.p2_conf >= 0.5]
```

	tweet_id	jpg_url	img_num	p1	p1_conf	p1_dog	p2	p2_conf	p2_dog	p3	p3_conf	p3_
...	...									...	...	

```
# check for prediction greater than 0.5 for p3_conf
image_predictions_df[image_predictions_df.p3_conf >= 0.5]
```

	tweet_id	jpg_url	img_num	p1	p1_conf	p1_dog	p2	p2_conf	p2_dog	p3	p3_conf	p3_
2070	891327558926688256	https://pbs.twimg.com/media/DF6hr6BUMAAzZgT.jpg	2									

Accessing the json_tweet_df dataset

1239 rows × 12 columns

```
json_tweet_df.head()
```

	Unnamed: 0	tweet_id	favorites	retweets	user_favourites	date_time
0	0	892420643555336193	33727	6979	147178	2017-08-01 16:23:56+00:00
1	1	892177421306343426	29254	5280	147178	2017-08-01 00:17:27+00:00
2	2	891815181378084864	21987	3466	147178	2017-07-31 00:18:03+00:00

```
# check the dataset info
json_tweet_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2327 entries, 0 to 2326
Data columns (total 6 columns):
 #   Column          Non-Null Count  Dtype 
---  ------          --------------  ----- 
 0   Unnamed: 0      2327 non-null   int64 
 1   tweet_id        2327 non-null   int64 
 2   favorites       2327 non-null   int64 
 3   retweets        2327 non-null   int64 
 4   user_favourites 2327 non-null   int64 
 5   date_time       2327 non-null   object
dtypes: int64(5), object(1)
memory usage: 109.2+ KB
```

```
json_tweet_df.duplicated().sum()
```

```
json_tweet_df.duplicated().sum()
```

```
0
```

Quality issues

- incomplete records in the json_tweet
- Column in_reply_to_status_id, in_reply_to_user_id, retweeted_status_id retweeted_status_user_id and retweeted_status_timestamp contains null values
- erroneous data types
- rating denominator value of *None* and less than 10
- incorrect dog like *a* and **None**
- presence of duplicate rows
- outliers in the rating_numerator e.g minimum value of 0 and maximum of 1776. There are other apart from these.
- p1, p2, p3 in the image prdiction dataset having underscore
- only true prediction with their confidence level are needed
- presence of irrelevant columns

Tidiness issues

- Column headers doggo, fluffer, pupper and puppo are values not headers
- only one table should exist

CLEANING

Define

> missing records in the json_tweet dataset due to tweets corresponding to the missing tweet_ids. We will merge the twitter_enhanced_df with the json_tweet dataset on the tweet_id column using right join so any tweet_id in the json_tweet that's not in the twitter_enhanced_df will not get added

Code

```
# merge the twitter_enhanced_df json_tweet
twitter_archive = twitter_enhanced_df.merge(json_tweet_df, on ='tweet_id', how = 'right')
```

Test

```
#print the first few rows
twitter_archive.head()
```

	tweet_id	in_reply_to_status_id	in_reply_to_user_id	timestamp	
0	892420643555336193	NaN	NaN	2017-08-01 16:23:56 +0000	href="http:/
1	892177421306343426	NaN	NaN	2017-08-01 00:17:27 +0000	href="http:/
2	891815181378084864	NaN	NaN	2017-07-31 00:18:03 +0000	href="http:/
3	891689557279858688	NaN	NaN	2017-07-30 15:58:51 +0000	href="http:/
4	891327558926688256	NaN	NaN	2017-07-29 16:00:24 +0000	href="http:/

5 rows × 22 columns

```
#print the shape
twitter_archive.shape
```

```
(2327, 22)
```

```
# make a copy of the twitter_archive
twitter_archive_df = twitter_archive.copy()
```

Define

> Presence of duplicate rows: rows with retweet information like retweeted_status_id, retweeted_status_user_id and retweeted_status_timestamp are duplicates as they are retweets. We are only interested in the original tweet information. we filter the dataframe for rows with null values. This gives a new dataframe with no retweet information

Code

```
# print shape of dataframe


# isolate the rows where retweet information is not null
tweet = twitter_archive_df[~twitter_archive_df[['retweeted_status_id', 'retweeted_status_user

# get the index of the new dataframe and store as list
tweet_1 = list( tweet.index)
tweet_1

# iterate through the list and drop any row number in the list present in the twitter_archive
for i in tweet_1:
  twitter_archive_df = twitter_archive_df.drop([i])
```

Test

```
# verify the result
twitter_archive_df[~twitter_archive_df[['retweeted_status_id', 'retweeted_status_user_id', 'r
```

tweet_id	in_reply_to_status_id	in_reply_to_user_id	timestamp	source	text	retwee

0 rows × 22 columns

Define

> Null values: **Column in_reply_to_status_id, in_reply_to_user_id, retweeted_status_id retweeted_status_user_id and retweeted_status_timestamp** contains too much Null

> values and since its not relevant to our analysis as we are intereted only in the original tweets, columns will be dropped

Code

```
# Drop irrelevant columns
twitter_archive_df = twitter_archive_df.drop(['in_reply_to_status_id', 'in_reply_to_user_id',
```

Test

```
twitter_archive_df
```

	tweet_id	timestamp	source	text	
0	892420643555336193	2017-08-01 16:23:56 +0000	<a href="http://twitter.com/download/iphone" r...	This is Phineas. He's a mystical boy. Only eve...	http
1	892177421306343426	2017-08-01 00:17:27 +0000	<a href="http://twitter.com/download/iphone" r...	This is Tilly. She's just checking pup on you....	http
2	891815181378084864	2017-07-31 00:18:03 +0000	<a href="http://twitter.com/download/iphone" r...	This is Archie. He is a rare Norwegian Pouncin...	http
3	891689557279858688	2017-07-30 15:58:51 +0000	<a href="http://twitter.com/download/iphone" r...	This is Darla. She commenced a snooze mid meal...	http
4	891327558926688256	2017-07-29 16:00:24	<a href="http://twitter.com/download/iphone"	This is Franklin. He would like	http

Define

> rating denominator of value *None* and values being less than the 10 and . All rating denominator should be 10 so we will replace values less than 10 with 10

2322	666049248165822465	2015-11-16 00:24:50 +0000	<a href="http://twitter.com/download/iphone"	1949 1st	http

Code

```
#mask = twitter_archive_df['rating_denominator'] != 10
# replace all values less than 10 with 10
twitter_archive_df['rating_denominator'] = twitter_archive_df['rating_denominator'].mask(twit

# Replace values None with 10
rating_denominator = {None: 10}
twitter_archive_df['rating_denominator'] =[rating_denominator[item] for item in twitter_archi
```

Test

2325	666029285002620928	2015-11-16 00:05:30 +0000	<a href="http://twitter.com/download/iphone"	brown	http

```
#verify for values less than 10
```

```
twitter_archive_df[twitter_archive_df['rating_denominator']<10]
```

tweet_id	timestamp	source	text	expanded_urls	rating_numerator	rating_denominato
				+0000	r...	Irish Setter.

```
# verify the result
twitter_archive_df[twitter_archive_df['rating_denominator'] == 'None']
```

tweet_id	timestamp	source	text	expanded_urls	rating_numerator	rating_denominato

```
twitter_archive_df.head()
```

	tweet_id	timestamp	source	text	
0	892420643555336193	2017-08-01 16:23:56 +0000	<a href="http://twitter.com/download/iphone" r...	This is Phineas. He's a mystical boy. Only eve...	https://t
1	892177421306343426	2017-08-01 00:17:27 +0000	<a href="http://twitter.com/download/iphone" r...	This is Tilly. She's just checking pup on you....	https://t
2	891815181378084864	2017-07-31 00:18:03 +0000	<a href="http://twitter.com/download/iphone" r...	This is Archie. He is a rare Norwegian Pouncin...	https://t
3	891689557279858688	2017-07-30 15:58:51 +0000	<a href="http://twitter.com/download/iphone" r...	This is Darla. She commenced a snooze mid meal...	https://t
4	891327558926688256	2017-07-29 16:00:24 +0000	<a href="http://twitter.com/download/iphone" r...	This is Franklin. He would like you to stop ca...	https://t

Define

> values in the p1, p2, p3 column of image_predictions dataset have underscore (_) which doesn't conform to the required schema as they are not column headers since

> the columns contains names

Code

```
# write a fuction that replace underscore(_) with space (' ')
def replace_char (df, column):

  df[column].astype('str')
  df[column] = df[column].str.replace('_', ' ')
  return df[column]


#image_predictions_df['p1'].astype('str')
#image_predictions_df['p1'].dtypes
replace_char (image_predictions_df, 'p1')
replace_char (image_predictions_df, 'p2')
replace_char (image_predictions_df, 'p3')
```

```
0                 Shetland sheepdog
1               Rhodesian ridgeback
2                         bloodhound
3                miniature pinscher
4                           Doberman
                    ...
2070    German short-haired pointer
2071                         spatula
2072                          kelpie
2073                        papillon
2074                          banana
Name: p3, Length: 2075, dtype: object
```

Test

```
image_predictions_df.head()
```

	tweet_id	jpg_url	img_num	p1
0	666020888022790149	https://pbs.twimg.com/media/CT4udn0WwAA0aMy.jpg	1	Welsh springer spaniel
1	666029285002620928	https://pbs.twimg.com/media/CT42GRgUYAA5iDo.jpg	1	redbone
2	666033412701032449	https://pbs.twimg.com/media/CT4521TWwAEvMyu.jpg	1	German shepherd

Define

> outliers in the rating_numerator e.g minimum value of 0 and maximum of 1776. There are other apart from these : rows with **rating numerator** of 0. There are only two rows with rating numerator of 0 and there is no information in the tweet about rating numerator so we drop them. other values like 1776 are valid so therefore should be left as it is

Code

```
# inspect the row(s) with rating numerator of 0
twitter_archive_df[twitter_archive_df['rating_numerator']==0]
```

	tweet_id	timestamp	source	text	
299	835152434251116546	2017-02-24 15:40:31 +0000	<a href="http://twitter.com/download/iphone" r...	When you're so blinded by your systematic plag...	https:/
988	746906459439529985	2016-06-26 03:22:31 +0000	<a href="http://twitter.com/download/iphone" r...	PUPDATE: can't see any. Even if I could, I cou...	https:/

```
# drop the rows
twitter_archive_df = twitter_archive_df.drop([299, 988])
```

Test

```
# verify if the rows have been deleted
twitter_archive_df[twitter_archive_df['rating_numerator']==0]
```

	tweet_id	timestamp	source	text	expanded_urls	rating_numerator	rating_denominato

Double-click (or enter) to edit

Define

> incorrect dog like a and None : dog name being **None**: the data is valid as no dog names were mentioned in the in the tweets involved, so we leave it that way but others like **a**, we check the tweet to extract the valid names and if none are present, we replace them with **None**

Code

```
# create a mask to extract the invalid names
mask_name = twitter_archive_df.name.str.contains('^[a-z]', regex = True)

# print out the names
twitter_archive_df[mask_name].name.value_counts().sort_index()
```

```
a               55
actually         2
all              1
an               6
by               1
getting          2
his              1
incredibly       1
infuriating      1
just             3
life             1
light            1
mad              1
my               1
not              2
officially       1
old              1
one              4
quite            3
space            1
such             1
the              8
this             1
unacceptable     1
very             4
Name: name, dtype: int64
```

```
# check rows with these values and inspect the tweet text
twitter_archive_df[mask_name].sample(50)
```

	tweet_id	timestamp	source	te
2037	671147085991960577	2015-11-30 02:01:49 +0000	<a href="http://twitter.com/download/iphone" r...	This is Helveti Listerine nam Rufus. Th
1894	674082852460433408	2015-12-08 04:27:30 +0000	<a href="http://twitter.com/download/iphone" r...	This is Sagitar Baklava m Loves her ne
2117	669923323644657664	2015-11-26 16:59:01 +0000	<a href="http://twitter.com/download/iphone" r...	This is a spott Lipit Rumpelstiltsk name
997	746369468511756288	2016-06-24 15:48:42 +0000	<a href="http://twitter.com/download/iphone" r...	This is an Ira Speed Kangaro It is not a c
2235	667538891197542400	2015-11-20 03:04:08 +0000	<a href="http://twitter.com" rel="nofollow">Tw...	This is southwe Coriander nam Klint. Ha
2282	666781792255496192	2015-11-18 00:55:42 +0000	<a href="http://twitter.com/download/iphone" r...	This is purebred Baca nam Octaviath. Ca
629	792913359805018113	2016-10-31 02:17:31 +0000	<a href="http://twitter.com/download/iphone" r...	Here is a perfe example someone w has
2169	668815180734689280	2015-11-23 15:35:39 +0000	<a href="http://twitter.com/download/iphone" r...	This is a w Toblerone fro Papua Ne Guinea
2189	668507509523615744	2015-11-22 19:13:05 +0000	<a href="http://twitter.com/download/iphone" r...	This is Birmingha Quagmire nam Chuk. Love
2226	667773195014021121	2015-11-20 18:35:10 +0000	<a href="http://twitter.com" rel="nofollow">Tw...	This is a ra Hungarian Pin named Jessic
1756	677644091929329666	2015-12-18 00:18:36 +0000	<a href="http://twitter.com/download/iphone" r...	This is a d swinging. I rea enjoyed it so
2320	666051853826850816	2015-11-16	<a href="http://twitter.com/download/iphone"	This is an o dog. Hard on t

2320	666051853826850816	00:35:11 +0000	href= http://twitter.com/download/iphone" r...	dog. Hard on t outside but lo
2124	669661792646373376	2015-11-25 23:39:47 +0000	<a href="http://twitter.com/download/iphone" r...	This is a bra dog. Excelle free climber. T
2317	666058600524156928	2015-11-16 01:01:59 +0000	<a href="http://twitter.com/download/iphone" r...	Here is the Ra Paul of retrieve folks! He
2244	667470559035432960	2015-11-19 22:32:36 +0000	<a href="http://twitter.com" rel="nofollow">Tw...	This is a northe Wahoo nam Kohl. He runs
2304	666337882303524864	2015-11-16 19:31:45 +0000	<a href="http://twitter.com/download/iphone" r...	This is extremely ra horn Parthenon. No
2275	666983947667116034	2015-11-18 14:18:59 +0000	<a href="http://twitter.com/download/iphone" r...	This is a cu Ticondero named Pepe. N fee
2321	666050758794694657	2015-11-16 00:30:50 +0000	<a href="http://twitter.com/download/iphone" r...	This is a tru beautiful Engli Wilson Staf
960	748977405889503236	2016-07-01 20:31:43 +0000	<a href="http://twitter.com/download/iphone" r...	What jokest sent in a p without a dog
989	746872823977771008	2016-06-26 01:08:52 +0000	<a href="http://twitter.com/download/iphone" r...	This is a carr We only ra dogs. Plea or
1357	700747788515020802	2016-02-19 18:24:26 +0000	<a href="http://twitter.com/download/iphone" r...	We only ra dogs. Pls st sending in nc car
1972	672482722825261057	2015-12-03 18:29:09 +0000	<a href="http://twitter.com/download/iphone" r...	This is light sab pup. Ready fight off ev
1499	690360449368465409	2016-01-22 02:28:52 +0000	<a href="http://twitter.com/download/iphone" r...	Stop sending lobsters. This the final wa
		2016-02-20	<a	"Pupper is

1354	700864154249383937	02:06:50 +0000	href="http://twitter.com/download/iphone" r...	present to wor Here is a bow
2008	671561002136281088	2015-12-01 05:26:34 +0000	<a href="http://twitter.com/download/iphone" r...	This is the be thing I've ev seen so sprea
1093	730924654643314689	2016-05-13 00:56:32 +0000	<a href="http://twitter.com/download/iphone" r...	We only ra dogs. Pls st sending nc canine
1340	702539513671897089	2016-02-24 17:04:07 +0000	<a href="http://twitter.com/download/iphone" r...	This is a W Tusc Poofwigg Careful not
2319	666055525042405380	2015-11-16 00:49:46 +0000	<a href="http://twitter.com/download/iphone" r...	Here is Siberian heav armored po bear
1887	674307341513269249	2015-12-08 19:19:32 +0000	<a href="http://vine.co" rel="nofollow">Vine -...	This is lit changing. 12/ https://t.co/SroT
1179	715733265223708672	2016-04-01 02:51:22 +0000	<a href="http://twitter.com/download/iphone" r...	This is a tac We only ra dogs. Plea onl
2183	668587383441514497	2015-11-23 00:30:28 +0000	<a href="http://vine.co" rel="nofollow">Vine -...	Never forget th vine. You will r stop wato
1312	704859558691414016	2016-03-02 02:43:09 +0000	<a href="http://twitter.com/download/iphone" r...	Here is heartbreaki scene of incredible
1696	680085611152338944	2015-12-24 18:00:19 +0000	<a href="https://about.twitter.com/products/tw...	This is by far t most coordinat series of
1768	677269281705472000	2015-12-16 23:29:14 +0000	<a href="http://twitter.com/download/iphone" r...	This is t happiest pupp I've ever see 10
2316	666063827256086533	2015-11-16 01:22:45 +0000	<a href="http://twitter.com/download/iphone" r...	This is t happiest dog y will ever se Ve
		2015-12-	<a	Guys this rea

1708	679530280114372609	23 05:13:38 +0000	href="http://twitter.com/download/iphone" r...	needs to stc We've be ove
1848	675109292475830276	2015-12-11 00:26:12 +0000	<a href="http://twitter.com/download/iphone" r...	C'mon guy We've been ov this. We or rate
1334	703041949650034688	2016-02-26 02:20:37 +0000	<a href="http://twitter.com/download/iphone" r...	This is an Ea African Chalu Seal. We only
897	755206590534418437	2016-07-19 01:04:16 +0000	<a href="http://twitter.com/download/iphone" r...	This is one of t most inspiratior stories
184	855459453768019968	2017-04-21 16:33:22 +0000	<a href="http://twitter.com/download/iphone" r...	Guys, we or rate dogs. This quite clearly
974	747885874273214464	2016-06-28 20:14:22 +0000	<a href="http://twitter.com/download/iphone" r...	This is a migh rare blue-tail hammer sherk
1429	695095422348574720	2016-02-04 04:03:57 +0000	<a href="http://twitter.com/download/iphone" r...	This is just beautiful pupp good shit evo
2323	666044226329800704	2015-11-16 00:04:52 +0000	<a href="http://twitter.com/download/iphone" r...	This is purebred Pie Morgan. Loves Net
2220	667861340749471744	2015-11-21 00:25:26 +0000	<a href="http://twitter.com/download/iphone" r...	This is Shotok Macadamia n named Cheryl
1069	736225175608430592	2016-05-27 15:58:54 +0000	<a href="http://twitter.com/download/iphone" r...	We only ra dogs. Plea stop sending non-
1825	675534494439489536	2015-12-12 04:35:48 +0000	<a href="http://twitter.com/download/iphone" r...	Seriously guys Only send dogs. I only ra
1407	697259378236399616	2016-02-10 03:22:44 +0000	<a href="http://twitter.com/download/iphone" r...	Please st sending in sabe toothed tige T
		2016-02-	<a	This is

1333	703079050210877440	26 04:48:02 +0000	href="http://twitter.com/download/iphone" r...	Butterr Cumberfloof. I not winc
		2016-05-		Say hello to m

> Upon careful analysis of the tweets, there seem to be no mention of dog names in the rows above. We replace all invalid names with **None**

```
# replace all invalid names with None
twitter_archive_df['name'] = twitter_archive_df['name'].str.replace('^[a-z]+', 'None', regex
```

Test

```
# verify the result
twitter_archive_df
```

	tweet_id	timestamp	source	text	
0	892420643555336193	2017-08-01 16:23:56 +0000	<a href="http://twitter.com/download/iphone" r...	This is Phineas. He's a mystical boy. Only eve...	http
1	892177421306343426	2017-08-01 00:17:27 +0000	<a href="http://twitter.com/download/iphone" r...	This is Tilly. She's just checking pup on you....	http
2	891815181378084864	2017-07-31 00:18:03 +0000	<a href="http://twitter.com/download/iphone" r...	This is Archie. He is a rare Norwegian Pouncin...	http
3	891689557279858688	2017-07-30 15:58:51 +0000	<a href="http://twitter.com/download/iphone" r...	This is Darla. She commenced a snooze mid meal...	http
4	891327558926688256	2017-07-29 16:00:24 +0000	<a href="http://twitter.com/download/iphone" r...	This is Franklin. He would like you to stop ca...	http

Double-click (or enter) to edit

Define

> only one table should exist. Merge twitter_archive_df with image_predictions_df

Code

```
# merge the twitter_archive_df with image_predictions_df
twitter_archive_df = twitter_archive_df.merge(image_predictions_df, on = 'tweet_id', how= 'in
```

Test

```
twitter_archive_df.head()
```

	tweet_id	timestamp	source	text	
0	892420643555336193	2017-08-01 16:23:56 +0000	<a href="http://twitter.com/download/iphone" r...	This is Phineas. He's a mystical boy. Only eve...	https://t
1	892177421306343426	2017-08-01 00:17:27 +0000	<a href="http://twitter.com/download/iphone" r...	This is Tilly. She's just checking pup on you....	https://t
2	891815181378084864	2017-07-31 00:18:03 +0000	<a href="http://twitter.com/download/iphone" r...	This is Archie. He is a rare Norwegian Pouncin...	https://t
3	891689557279858688	2017-07-30 15:58:51 +0000	<a href="http://twitter.com/download/iphone" r...	This is Darla. She commenced a snooze mid meal...	https://t
4	891327558926688256	2017-07-29 16:00:24 +0000	<a href="http://twitter.com/download/iphone" r...	This is Franklin. He would like you to stop ca...	https://t

5 rows × 26 columns

```
twitter_archive_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1984 entries, 0 to 1983
Data columns (total 26 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   tweet_id            1984 non-null   int64
 1   timestamp           1984 non-null   object
 2   source              1984 non-null   object
 3   text                1984 non-null   object
 4   expanded_urls       1984 non-null   object
 5   rating_numerator    1984 non-null   int64
 6   rating_denominator  1984 non-null   int64
 7   name                1984 non-null   object
 8   doggo               1984 non-null   object
 9   floofer             1984 non-null   object
 10  pupper              1984 non-null   object
```

```
 11  puppo               1984 non-null   object
 12  favorites           1984 non-null   int64
 13  retweets            1984 non-null   int64
 14  user_favourites     1984 non-null   int64
 15  jpg_url             1984 non-null   object
 16  img_num             1984 non-null   int64
 17  p1                  1984 non-null   object
 18  p1_conf             1984 non-null   float64
 19  p1_dog              1984 non-null   bool
 20  p2                  1984 non-null   object
 21  p2_conf             1984 non-null   float64
 22  p2_dog              1984 non-null   bool
 23  p3                  1984 non-null   object
 24  p3_conf             1984 non-null   float64
 25  p3_dog              1984 non-null   bool
dtypes: bool(3), float64(3), int64(7), object(13)
memory usage: 377.8+ KB
```

Define

> only true prediction with their confidence level are needed. Write a function that extract the true prediction and and their corresponding confidence level

Code

```
# We will store the fisrt true algorithm with it's level of confidence
prediction = []
confidence = []

# prediction_confidence function:
# search the first true algorithm and append it to a list with it's level of confidence
# if flase prediction_algorthm will have a value of Nan
def prediction_confidence(df):
    if df['p1_dog'] == True:
        prediction.append(df['p1'])
        confidence.append(df['p1_conf'])
    elif df['p2_dog'] == True:
        prediction.append(df['p2'])
        confidence.append(df['p2_conf'])
    elif df['p3_dog'] == True:
        prediction.append(df['p3'])
        confidence.append(df['p3_conf'])
    else:
        prediction.append('None')
        confidence.append(0)

twitter_archive_df.apply(prediction_confidence, axis=1)
```

```
twitter_archive_df['prediction'] = prediction
twitter_archive_df['confidence'] = confidence

# delete the original prediction and confidence level columns and other columns of image_prdi
twitter_archive_df = twitter_archive_df.drop(['img_num', 'p1','p1_conf', 'p1_dog', 'p2', 'p2_
```

Test

```
# verify the change
twitter_archive_df.head()
```

	tweet_id	timestamp	source	text	
0	892420643555336193	2017-08-01 16:23:56 +0000	<a href="http://twitter.com/download/iphone" r...	This is Phineas. He's a mystical boy. Only eve...	https://t
1	892177421306343426	2017-08-01 00:17:27 +0000	<a href="http://twitter.com/download/iphone" r...	This is Tilly. She's just checking pup on you....	https://t
2	891815181378084864	2017-07-31 00:18:03 +0000	<a href="http://twitter.com/download/iphone" r...	This is Archie. He is a rare Norwegian Pouncin...	https://t
3	891689557279858688	2017-07-30 15:58:51 +0000	<a href="http://twitter.com/download/iphone" r...	This is Darla. She commenced a snooze mid meal...	https://t
4	891327558926688256	2017-07-29 16:00:24 +0000	<a href="http://twitter.com/download/iphone" r...	This is Franklin. He would like you to stop ca...	https://t

column headers

> **doggo, floofer, pupper and puppo** should be values not column headers. A function is created that extract each dog stage count together with dog stage and append it to a

> list. A new row is then created in the twitter_archive_df containing this information

```
# create an empty list to store the dataframe
dog_stage = []

#write a function that store the count of each type of dog
def get_dog(df):
  # initiate a count
  count =0
  if df['doggo'] != 'None':
    count += 1
    dog_stage.append(df['doggo'])
  if df['floofer'] != 'None':
    count += 1
    dog_stage.append(df['floofer'])
  if df['pupper'] != 'None':
    count += 1
    dog_stage.append(df['pupper'])
  if df['puppo'] != 'None':
    count += 1
    dog_stage.append(df['puppo'])
  if count > 1:
        dog_stage.pop()
        dog_stage.pop()
        dog_stage.append('multiple')
  if count == 0:
        dog_stage.append('None')

twitter_archive_df.apply(get_dog, axis=1)
twitter_archive_df['dog_stage'] = dog_stage
```

```
# drop the columns doggo, floofer, pupper, puppo
twitter_archive_df = twitter_archive_df.drop(['doggo', 'floofer', 'pupper', 'puppo'], axis =
```

Test

```
twitter_archive_df.head()
```

	tweet_id	timestamp	source	text	
0	892420643555336193	2017-08-01 16:23:56 +0000	<a href="http://twitter.com/download/iphone" r...	This is Phineas. He's a mystical boy. Only eve...	https://t
1	892177421306343426	2017-08-01 00:17:27 +0000	<a href="http://twitter.com/download/iphone" r...	This is Tilly. She's just checking pup on you....	https://t
2	891815181378084864	2017-07-31 00:18:03 +0000	<a href="http://twitter.com/download/iphone" r...	This is Archie. He is a rare Norwegian Pouncin...	https://t
3	891689557279858688	2017-07-30 15:58:51 +0000	<a href="http://twitter.com/download/iphone" r...	This is Darla. She commenced a snooze mid meal...	https://t
4	891327558926688256	2017-07-29 16:00:24	<a href="http://twitter.com/download/iphone"	This is Franklin. He would like	https://t

```
# check dataset for any othe issues
twitter_archive_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1984 entries, 0 to 1983
Data columns (total 15 columns):
 #   Column              Non-Null Count  Dtype  
---  ------              --------------  -----  
 0   tweet_id            1984 non-null   int64  
 1   timestamp           1984 non-null   object 
 2   source              1984 non-null   object 
 3   text                1984 non-null   object 
 4   expanded_urls       1984 non-null   object 
 5   rating_numerator    1984 non-null   int64  
 6   rating_denominator  1984 non-null   int64  
 7   name                1984 non-null   object 
 8   favorites           1984 non-null   int64  
 9   retweets            1984 non-null   int64  
 10  user_favourites     1984 non-null   int64  
 11  jpg_url             1984 non-null   object 
 12  prediction          1984 non-null   object 
 13  confidence          1984 non-null   float64
 14  dog_stage           1984 non-null   object 
dtypes: float64(1), int64(6), object(8)
memory usage: 248.0+ KB
```

Define

> Erroneous data types : change datatype of tweet_id to string, timestamp to datetime, rating_numerattor to float and dog_stage to category

Code

```
# change data type
twitter_archive_df['tweet_id'] = twitter_archive_df['tweet_id'].astype('str')
twitter_archive_df['dog_stage'] = twitter_archive_df['dog_stage'].astype('category')
twitter_archive_df['timestamp'] = pd.to_datetime(twitter_archive_df['timestamp'])
```

Test

```
# verfy result
twitter_archive_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1984 entries, 0 to 1983
Data columns (total 15 columns):
 #   Column              Non-Null Count  Dtype              
---  ------              --------------  -----              
 0   tweet_id            1984 non-null   object             
 1   timestamp           1984 non-null   datetime64[ns, UTC]
 2   source              1984 non-null   object             
 3   text                1984 non-null   object             
 4   expanded_urls       1984 non-null   object             
 5   rating_numerator    1984 non-null   int64              
 6   rating_denominator  1984 non-null   int64              
 7   name                1984 non-null   object             
 8   favorites           1984 non-null   int64              
 9   retweets            1984 non-null   int64              
 10  user_favourites     1984 non-null   int64              
 11  jpg_url             1984 non-null   object             
 12  prediction          1984 non-null   object             
 13  confidence          1984 non-null   float64            
 14  dog_stage           1984 non-null   category           
dtypes: category(1), datetime64[ns, UTC](1), float64(1), int64(5), object(7)
memory usage: 234.7+ KB
```

Storing The Dataset

```
twitter_archive_df.to_csv('twitter_archive_master.csv', index=False)
```

Double-click (or enter) to edit

Analysis and Visualization

Read the file twitter_archive_master

> This data set to be used for analysis and visualization have been gathered, accessed both visually and programmatically and also cleaned.

```
# read the twitter_archive_master file
twitter_archive_master = pd.read_csv('twitter_archive_master.csv', parse_dates = ['timestamp'

# print the first few rows
twitter_archive_master.head()
```

What is the count of each dog stage?

2017-08-01 He's a

```
# find the count of each dog breed
count_of_breed_type = twitter_archive_master['dog_stage'].value_counts()
print(count_of_breed_type)
```

```
None        1679
pupper       203
doggo         62
puppo         22
multiple      11
floofer        7
Name: dog_stage, dtype: int64
```

00:18:03+00:00

> About 80% of the dogs posted had no dogstage mentioned but for those whose dog stage was mentioned, pupper was seen to be predominant which is about 67% of the dogs with dog stage

mid meal

What is the mean rating

16:00:24+00:00 would like

```
# fing the mean rating
twitter_archive_master.rating_numerator.mean()
```

```
10.587426326129666
```

> The average rating of dogs with or without dog stage is approximately 11. This means we should expect the rating of any dog to be around this value

What is the median rating

```
# find the median rating
twitter_archive_master['rating_numerator'].median()
```

```
11.0
```

> The mid rating is almost the same as the average rating

What value are dogs being rated most often?

```
# find the most frequent dog rating value
display(twitter_archive_master['rating_numerator'].mode())
```

```
0    12
dtype: int64
```

> This shows that more dogs are rated 12/10

count of each dog rating

```
# get count for each dog rating
twitter_archive_master['rating_numerator'].sort_values(ascending = False).value_counts()
```

```
12    471
10    425
11    413
13    275
9     151
8      95
7      52
14     38
5      34
6      32
3      19
4      16
2       9
1       5
15      1
Name: rating_numerator, dtype: int64
```

> Although the most common rating is 12, this result shows that margin between rating 12 and the next top two rating (10 and 11) is not too wide but it gets wider as we go from 13 downwards.

What is the highest and lowest retweets a tweet has ever gotten?

```
# find the highext retweet count
print('The highest retweets is {}'.format(twitter_archive_master['retweets'].max()))
```

```
# fing the lowest retweets
print('The lowest retweets is {}'.format(twitter_archive_master['retweets'].min()))
```

```
The highest retweets is 70427
The lowest retweets is 11
```

In what month and year did the highest retweet occur?

```
# extract month and year
month = twitter_archive_master['timestamp'].dt.month
year = twitter_archive_master['timestamp'].dt.year

# fing the highest retweet grouping by mnoth and year
twitter_archive_master.groupby([month, year])['retweets'].max()
```

```
timestamp  timestamp
1          2016         14845
           2017         39810
2          2016         13059
           2017         15383
3          2016         16597
           2017         13543
4          2016          7216
           2017         16094
5          2016         15470
           2017         30091
6          2016         70427
           2017         37304
7          2016         16264
           2017         15695
8          2016         26644
           2017          6979
9          2016         15470
10         2016         11677
11         2015         14570
           2016         21898
12         2015         28482
           2016         51485
Name: retweets, dtype: int64
```

> The highest retweet occurred in june 2016 with 70000+ retweets, followwed by december 2016 with 50000+ retweets and its lowest being about 7000 in august 2017

What is the Highest and lowest favorites or likes a tweet has gotten?

```
# find the highest number of likes
print('The highest number of likes is {}'.format(twitter_archive_master['favorites'].max()))
```

```
# find the lowest numberof likes
print('The lowest number of likes is {}'.format(twitter_archive_master['favorites'].min()))
```

```
The highest number of likes is 144398
The lowest number of likes is 0
```

In what month and year did the highest number of likes occur?

```
twitter_archive_master.groupby([month, year])['favorites'].max()
```

```
timestamp  timestamp
1          2016          33720
           2017         123766
2          2016          32986
           2017          62315
3          2016          30235
           2017          41117
4          2016          17320
           2017          41839
5          2016          49285
           2017         108583
6          2016         144398
           2017          92591
7          2016          41117
           2017          67085
8          2016          46141
           2017          33727
9          2016          28025
10         2016          28321
11         2015          42175
           2016          46655
12         2015          73278
           2016         111320
Name: favorites, dtype: int64
```

> WeRateDogs had it highest number of likes in june 2016 with about 144000+ likes which is the same day it had it highest retweets. It had it lowest likes in April 2016 which is about 17000+

Double-click (or enter) to edit

```
# find the
```

bold text

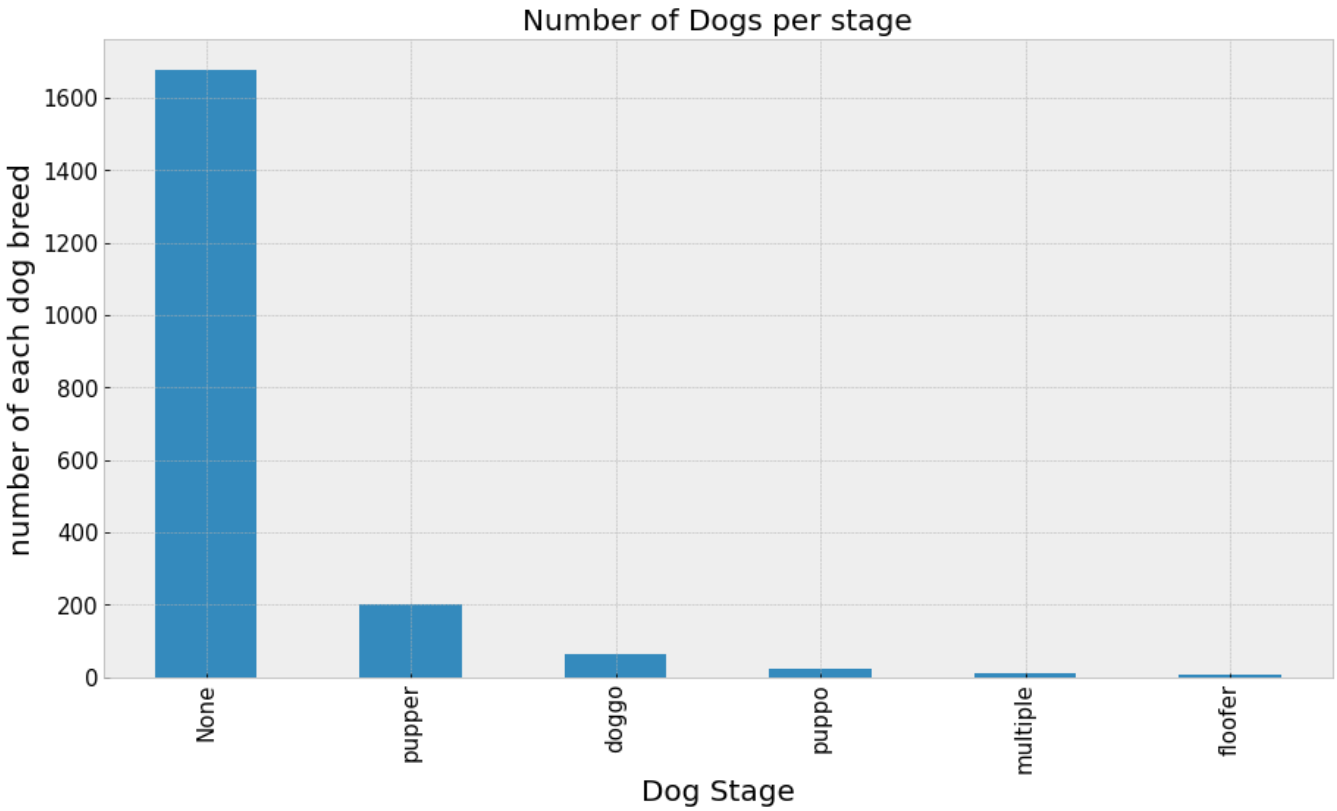
Visualizations

```
# visualize
# set style
plt.style.use('bmh')
# initialize subplot
fig, ax = plt.subplots()

#set plot size
fig.set_size_inches(15, 8)

# draw a bar plot
twitter_archive_master['dog_stage'].value_counts().plot(kind = 'bar')

# set axis labels and title
ax.set_title('Number of Dogs per stage', fontsize = 20)
ax.set_xlabel('Dog Stage', fontsize = 20)
ax.set_ylabel('number of each dog breed', fontsize =20 )
ax.tick_params(axis='both', which='major', labelsize=15)
plt.show()
```

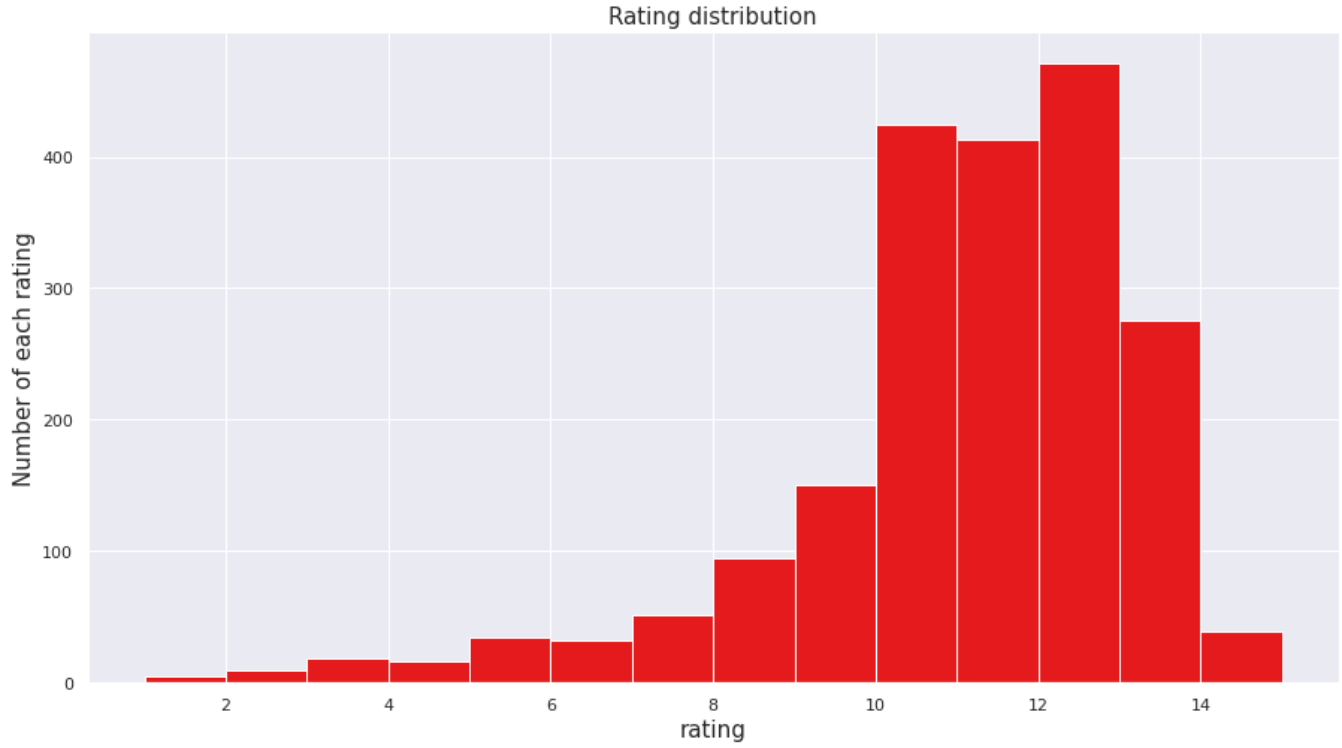


> Number of dogs with no dog stage have the highest population followed by dog stage pupper

Distribution of rating numerator

```
# draw a histogram showing the distribution of the rating numerator
twitter_archive_master['rating_numerator'].unique()
twitter_archive_master['rating_numerator'].hist( bins =14)
plt.xlabel('rating', fontsize = 15)
plt.ylabel('Number of each rating', fontsize = 15)
plt.title('Rating distribution', fontsize = 15)
```

```
Text(0.5, 1.0, 'Rating distribution')
```



> the histogram show the distribution of the dog rating which appears to be left-skewed as more rating are to the right side consisting of high ratings

How Does Number of likes Vary per month for Each Year

Relational plot

```
# set the color
sns.set_palette('Set1')

# extract month from timestamp
monthly_retweets = twitter_archive_master['timestamp'].dt.month

# draw a relational plot
g = sns.relplot(x= monthly_retweets, y = twitter_archive_master['favorites'],  col = twitter_

# set axis label and title
g.set(xlabel = 'time (month)', ylabel = 'Number of tweet likes')
g.set_xticklabels(size = 12)
g.fig.suptitle('Number of Tweet Likes Per Month For Each Year', y = 1.03)

#set tick labels
plt.xticks([1,2,3,4,5,6,7,8,9,10,11,12],['January', 'february', 'March', 'April', 'May', 'Jun
g.set_xticklabels(rotation = 90)


# set the plot size
plt.figure(figsize=(20,40))
plt.show()
```