Wrangle_act We RateDogs

Table of Contents

Introduction

```
The dataset that is wrangled is the tweet archive of Twitter user
@dog_rates, also known as WeRateDogs.
```

Gathering Data

Gather each of the three pieces of data

- **twitter-archive-enhanced.csv** the WeRateDogs Twitter archive.
- **image_predictions.tsv** the tweet image predictions, i.e., what breed of dog (or other object, animal, etc.) is present in each tweet according to a neural network.
- **tweet_json.txt**
  - Each tweet's retweet count and favorite ("like") count at minimum, and any additional data you find interesting. Using the tweet IDs in the WeRateDogs Twitter archive, query the Twitter API for each tweet's JSON data using Python's Tweepy library and store each tweet's entire set of JSON data in a file called tweet_json.txt file.
  - Each tweet's JSON data should be written to its own line. Then read this .txt file line by line into a pandas DataFrame with (at minimum) tweet ID, retweet count, and favorite count.

1. WeRateDogs Twitter archive: twitter-archive-enhanced.csv

In [2]:
```
# Dependencies
import pandas as pd
import numpy as np
import tweepy
import requests
import json
import os
import re
import matplotlib.pyplot as plt
import datetime
%matplotlib inline

```

In [3]:
```
# import csv file as a dataframe
tweet_archive_df = pd.read_csv("twitter-archive-enhanced.csv")

```

2. The tweet image predictions image_predictions.tsv

In [4]:
```
# URL downloaded programatically
file_url = 'https://d17h27t6h515a5.cloudfront.net/topher/2017/August/599

response = requests.get(file_url)

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

In [5]:
```

# Read Tsv files
image_prediction_df = pd.read_csv('image-predictions.tsv', sep='\t' )
```

3. tweet_json.txt

In [6]:
```
CONSUMER_KEY = "############"
CONSUMER_SECRET = "###############"
OAUTH_TOKEN = "#############"
OAUTH_TOKEN_SECRET = "############"
#https://stackoverflow.com/questions/28384588/twitter-api-get-tweets-wit
auth = tweepy.OAuthHandler(CONSUMER_KEY, CONSUMER_SECRET)
auth.set_access_token(OAUTH_TOKEN, OAUTH_TOKEN_SECRET)
api = tweepy.API(auth,
                 parser = tweepy.parsers.JSONParser(),
                 wait_on_rate_limit = True,
                 wait_on_rate_limit_notify = True)


```

In [7]:

```python
#Download Tweepy status object based on Tweet ID and store in list
tweets_list = []
# Tweets that can't be found are saved in the list below:
cant_find_tweetsid_list = []
for tweet_id in tweet_archive_df['tweet_id']:
    try:
        tweets_list.append(api.get_status(tweet_id))
    except:
        cant_find_tweetsid_list.append(tweet_id)
        print(f"We cann't find those id: {tweet_id}")
```

```
We cann't find those id: 872668790621863937
We cann't find those id: 872261713294495745
We cann't find those id: 869988702071779329
We cann't find those id: 866816280283807744
We cann't find those id: 861769973181624320
We cann't find those id: 856602993587888130
We cann't find those id: 845459076796616705
We cann't find those id: 844704788403113984
We cann't find those id: 842892208864923648
We cann't find those id: 837012587749474308
We cann't find those id: 827228250799742977
We cann't find those id: 812747805718642688
We cann't find those id: 802247111496568832
We cann't find those id: 775096608509886464
We cann't find those id: 770743923962707968
Rate limit reached. Sleeping for: 709
We cann't find those id: 754011816964026368
We cann't find those id: 680055455951884288
Rate limit reached. Sleeping for: 710
```

In [18]:

```python
print(f"We get data for {len(tweets_list)} tweet ids")
print(f"There are {len(cant_find_tweetsid_list)} tweet ids we cann't get
```

```
We get data for 2337 tweet ids
There are 19 tweet ids we cann't get data
```

In [40]:

```
# Create a dataframe from tweets_list
RateDogs_df = pd.DataFrame(tweets_list)
RateDogs_df = df[['created_at', 'id', 'text', 'in_reply_to_user_id_str',
RateDogs_df.head()
```

Out[40]:

	created_at	id	text	in_reply_to_user_id_str	user	retweet_
0	Tue Aug 01 16:23:56 +0000 2017	892420643555336193	This is Phineas. He's a mystical boy. Only eve...	None	{'id': 4196983835, 'id_str': '4196983835', 'na...	
1	Tue Aug 01 00:17:27 +0000 2017	892177421306343426	This is Tilly. She's just checking pup on you....	None	{'id': 4196983835, 'id_str': '4196983835', 'na...	
2	Mon Jul 31 00:18:03 +0000 2017	891815181378084864	This is Archie. He is a rare Norwegian Pouncin...	None	{'id': 4196983835, 'id_str': '4196983835', 'na...	
3	Sun Jul 30 15:58:51 +0000 2017	891689557279858688	This is Darla. She commenced a snooze mid meal...	None	{'id': 4196983835, 'id_str': '4196983835', 'na...	
4	Sat Jul 29 16:00:24 +0000 2017	891327558926688256	This is Franklin. He would like you to stop ca...	None	{'id': 4196983835, 'id_str': '4196983835', 'na...	

In [41]:

```
# Save dataframe as csv.file
RateDogs_df.to_csv("RateDogs.csv",index=False)
```

Assessing Data

Visial Assessment

In [7]:

```
# WeRateDogs Twitter archive dataframe
tweet_archive_df
```

Out[7]:

	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://twi
1	892177421306343426	NaN	NaN	2017-08-01 00:17:27 +0000	href="http://twi
2	891815181378084864	NaN	NaN	2017-07-31 00:18:03 +0000	href="http://twi
3	891689557279858688	NaN	NaN	2017-07-30 15:58:51 +0000	href="http://twi
4	891327558926688256	NaN	NaN	2017-07-29 16:00:24 +0000	href="http://twi
5	891087950875897856	NaN	NaN	2017-07-29 00:08:17 +0000	href="http://twi
6	890971913173991426	NaN	NaN	2017-07-28 16:27:12 +0000	href="http://twi
7	890729181411237888	NaN	NaN	2017-07-28 00:22:40 +0000	href="http://twi
8	890609185150312448	NaN	NaN	2017-07-27 16:25:51 +0000	href="http://twi
9	890240255349198849	NaN	NaN	2017-07-26 15:59:51 +0000	href="http://twi
10	890006608113172480	NaN	NaN	2017-07-26 00:31:25 +0000	href="http://twi
11	889880896479866881	NaN	NaN	2017-07-25 16:11:53 +0000	href="http://twi

	tweet_id	in_reply_to_status_id	in_reply_to_user_id	timestamp	
12	889665388333682689	NaN	NaN	2017-07-25 01:55:32 +0000	href="http://twi
13	889638837579907072	NaN	NaN	2017-07-25 00:10:02 +0000	href="http://twi
14	889531135344209921	NaN	NaN	2017-07-24 17:02:04 +0000	href="http://twi
15	889278841981685760	NaN	NaN	2017-07-24 00:19:32 +0000	href="http://twi
16	888917238123831296	NaN	NaN	2017-07-23 00:22:39 +0000	href="http://twi
17	888804989199671297	NaN	NaN	2017-07-22 16:56:37 +0000	href="http://twi
18	888554962724278272	NaN	NaN	2017-07-22 00:23:06 +0000	href="http://twi
19	888202515573088257	NaN	NaN	2017-07-21 01:02:36 +0000	href="http://twi
20	888078434458587136	NaN	NaN	2017-07-20 16:49:33 +0000	href="http://twi
21	887705289381826560	NaN	NaN	2017-07-19 16:06:48 +0000	href="http://twi
22	887517139158093824	NaN	NaN	2017-07-19 03:39:09 +0000	href="http://twi
23	887473957103951883	NaN	NaN	2017-07-19 00:47:34 +0000	href="http://twi
24	887343217045368832	NaN	NaN	2017-07-18 16:08:03 +0000	href="http://twi
25	887101392804085760	NaN	NaN	2017-07-18 00:07:08 +0000	href="http://twi

	tweet_id	in_reply_to_status_id	in_reply_to_user_id	timestamp	
26	886983233522544640	NaN	NaN	2017-07-17 16:17:36 +0000	href="http://twi
27	886736880519319552	NaN	NaN	2017-07-16 23:58:41 +0000	href="http://twi
28	886680336477933568	NaN	NaN	2017-07-16 20:14:00 +0000	href="http://twi
29	886366144734445568	NaN	NaN	2017-07-15 23:25:31 +0000	href="http://twi
...	...	...	...	...	
2326	666411507551481857	NaN	NaN	2015-11-17 00:24:19 +0000	href="http://twi
2327	666407126856765440	NaN	NaN	2015-11-17 00:06:54 +0000	href="http://twi
2328	666396247373291520	NaN	NaN	2015-11-16 23:23:41 +0000	href="http://twi
2329	666373753744588802	NaN	NaN	2015-11-16 21:54:18 +0000	href="http://twi
2330	666362758909284353	NaN	NaN	2015-11-16 21:10:36 +0000	href="http://twi
2331	666353288456101888	NaN	NaN	2015-11-16 20:32:58 +0000	href="http://twi
2332	666345417576210432	NaN	NaN	2015-11-16 20:01:42 +0000	href="http://twi
2333	666337882303524864	NaN	NaN	2015-11-16 19:31:45 +0000	href="http://twi
2334	666293911632134144	NaN	NaN	2015-11-16 16:37:02 +0000	href="http://twi
2335	666287406224695296	NaN	NaN	2015-11-16 16:11:11 +0000	href="http://twi
2336	666273097616637952	NaN	NaN	2015-11-16 15:14:19 +0000	href="http://twi
2337	666268910803644416	NaN	NaN	2015-11-16 14:57:41 +0000	href="http://twi

	tweet_id	in_reply_to_status_id	in_reply_to_user_id	timestamp	
2338	666104133288665088	NaN	NaN	2015-11-16 04:02:55 +0000	href="http://twi
2339	666102155909144576	NaN	NaN	2015-11-16 03:55:04 +0000	href="http://twi
2340	666099513787052032	NaN	NaN	2015-11-16 03:44:34 +0000	href="http://twi
2341	666094000022159362	NaN	NaN	2015-11-16 03:22:39 +0000	href="http://twi
2342	666082916733198337	NaN	NaN	2015-11-16 02:38:37 +0000	href="http://twi
2343	666073100786774016	NaN	NaN	2015-11-16 01:59:36 +0000	href="http://twi
2344	666071193221509120	NaN	NaN	2015-11-16 01:52:02 +0000	href="http://twi
2345	666063827256086533	NaN	NaN	2015-11-16 01:22:45 +0000	href="http://twi
2346	666058600524156928	NaN	NaN	2015-11-16 01:01:59 +0000	href="http://twi
2347	666057090499244032	NaN	NaN	2015-11-16 00:55:59 +0000	href="http://twi
2348	666055525042405380	NaN	NaN	2015-11-16 00:49:46 +0000	href="http://twi
2349	666051853826850816	NaN	NaN	2015-11-16 00:35:11 +0000	href="http://twi
2350	666050758794694657	NaN	NaN	2015-11-16 00:30:50 +0000	href="http://twi
2351	666049248165822465	NaN	NaN	2015-11-16 00:24:50 +0000	href="http://twi
2352	666044226329800704	NaN	NaN	2015-11-16 00:04:52 +0000	href="http://twi
2353	666033412701032449	NaN	NaN	2015-11-15 23:21:54 +0000	href="http://twi
2354	666029285002620928	NaN	NaN	2015-11-15 23:05:30 +0000	href="http://twi

	tweet_id	in_reply_to_status_id	in_reply_to_user_id	timestamp	
2355	666020888022790149	NaN	NaN	2015-11-15 22:32:08 +0000	href="http://twi

2356 rows × 17 columns

In [8]:
```python
# tweet image predictions dataframe
image_prediction_df
```

Out[8]:

	tweet_id	jpg_url	img_num	
0	666020888022790149	https://pbs.twimg.com/media/CT4udn0WwAA0aMy.jpg	1	We
1	666029285002620928	https://pbs.twimg.com/media/CT42GRgUYAA5iDo.jpg	1	
2	666033412701032449	https://pbs.twimg.com/media/CT4521TWwAEvMyu.jpg	1	
3	666044226329800704	https://pbs.twimg.com/media/CT5Dr8HUEAA-lEu.jpg	1	R
4	666049248165822465	https://pbs.twimg.com/media/CT5IQmsXIAAKY4A.jpg	1	
5	666050758794694657	https://pbs.twimg.com/media/CT5Jof1WUAEuVxN.jpg	1	Ber
6	666051853826850816	https://pbs.twimg.com/media/CT5KoJ1WoAAJash.jpg	1	
7	666055525042405380	https://pbs.twimg.com/media/CT5N9tpXIAAifs1.jpg	1	
8	666057090499244032	https://pbs.twimg.com/media/CT5PY90WoAAQGLo.jpg	1	
9	666058600524156928	https://pbs.twimg.com/media/CT5Qw94XAAA_2dP.jpg	1	
10	666063827256086533	https://pbs.twimg.com/media/CT5Vg_wXIAAXfnj.jpg	1	
11	666071193221509120	https://pbs.twimg.com/media/CT5cN_3WEAAlOoZ.jpg	1	
12	666073100786774016	https://pbs.twimg.com/media/CT5d9DZXAAALcwe.jpg	1	
13	666082916733198337	https://pbs.twimg.com/media/CT5m4VGWEAAtKc8.jpg	1	
14	666094000022159362	https://pbs.twimg.com/media/CT5w9gUW4AAsBNN.jpg	1	
15	666099513787052032	https://pbs.twimg.com/media/CT51-JJUEAA6hV8.jpg	1	
16	666102155909144576	https://pbs.twimg.com/media/CT54YGiWUAEZnoK.jpg	1	
17	666104133288665088	https://pbs.twimg.com/media/CT56LSZWoAAlJj2.jpg	1	
18	666268910803644416	https://pbs.twimg.com/media/CT8QCd1WEAADXws.jpg	1	
19	666273097616637952	https://pbs.twimg.com/media/CT8T1mtUwAA3aqm.jpg	1	
20	666287406224695296	https://pbs.twimg.com/media/CT8g3BpUEAAuFjg.jpg	1	
21	666293911632134144	https://pbs.twimg.com/media/CT8mx7KW4AEQu8N.jpg	1	
22	666337882303524864	https://pbs.twimg.com/media/CT9OwFIWEAMuRje.jpg	1	
23	666345417576210432	https://pbs.twimg.com/media/CT9Vn7PWoAA_ZCM.jpg	1	
24	666353288456101888	https://pbs.twimg.com/media/CT9cx0tUEAAhNN_.jpg	1	
25	666362758909284353	https://pbs.twimg.com/media/CT9lXGsUcAAyUFt.jpg	1	
26	666373753744588802	https://pbs.twimg.com/media/CT9vZEYWUAAlZ05.jpg	1	coa
27	666396247373291520	https://pbs.twimg.com/media/CT-D2ZHWIAA3gK1.jpg	1	
28	666407126856765440	https://pbs.twimg.com/media/CT-NvwmW4AAugGZ.jpg	1	black-
29	666411507551481857	https://pbs.twimg.com/media/CT-RugiWIAELEaq.jpg	1	
...	...	...	...	
2045	886366144734445568	https://pbs.twimg.com/media/DE0BTnQUwAApKEH.jpg	1	
2046	886680336477933568	https://pbs.twimg.com/media/DE4fEDzWAAAyHMM.jpg	1	

	tweet_id	jpg_url	img_num	
2047	886736880519319552	https://pbs.twimg.com/media/DE5Se8FXcAAJFx4.jpg	1	
2048	886983233522544640	https://pbs.twimg.com/media/DE8yicJW0AAAvBJ.jpg	2	
2049	887101392804085760	https://pbs.twimg.com/media/DE-eAq6UwAA-jaE.jpg	1	
2050	887343217045368832	https://pbs.twimg.com/ext_tw_video_thumb/88734...	1	
2051	887473957103951883	https://pbs.twimg.com/media/DFDw2tyUQAAAFke.jpg	2	
2052	887517139158093824	https://pbs.twimg.com/ext_tw_video_thumb/88751...	1	
2053	887705289381826560	https://pbs.twimg.com/media/DFHDQBbXgAEqY7t.jpg	1	
2054	888078434458587136	https://pbs.twimg.com/media/DFMWn56WsAAkA7B.jpg	1	
2055	888202515573088257	https://pbs.twimg.com/media/DFDw2tyUQAAAFke.jpg	2	
2056	888554962724278272	https://pbs.twimg.com/media/DFTH_O-UQAACu20.jpg	3	
2057	888804989199671297	https://pbs.twimg.com/media/DFWra-3VYAA2piG.jpg	1	
2058	888917238123831296	https://pbs.twimg.com/media/DFYRgsOUQAARGhO.jpg	1	
2059	889278841981685760	https://pbs.twimg.com/ext_tw_video_thumb/88927...	1	
2060	889531135344209921	https://pbs.twimg.com/media/DFg_2PVW0AEHN3p.jpg	1	
2061	889638837579907072	https://pbs.twimg.com/media/DFihzFfXsAYGDPR.jpg	1	
2062	889665388333682689	https://pbs.twimg.com/media/DFi579UWsAAatzw.jpg	1	
2063	889880896479866881	https://pbs.twimg.com/media/DFl99B1WsAITKsg.jpg	1	
2064	890006608113172480	https://pbs.twimg.com/media/DFnwSY4WAAAMliS.jpg	1	
2065	890240255349198849	https://pbs.twimg.com/media/DFrEyVuW0AAO3t9.jpg	1	
2066	890609185150312448	https://pbs.twimg.com/media/DFwUU__XcAEpyXI.jpg	1	
2067	890729181411237888	https://pbs.twimg.com/media/DFyBahAVwAAhUTd.jpg	2	
2068	890971913173991426	https://pbs.twimg.com/media/DF1eOmZXUAALUcq.jpg	1	
2069	891087950875897856	https://pbs.twimg.com/media/DF3HwyEWsAABqE6.jpg	1	Chesa
2070	891327558926688256	https://pbs.twimg.com/media/DF6hr6BUMAAzZgT.jpg	2	
2071	891689557279858688	https://pbs.twimg.com/media/DF_q7IAWsAEuuN8.jpg	1	
2072	891815181378084864	https://pbs.twimg.com/media/DGBdLU1WsAANxJ9.jpg	1	
2073	892177421306343426	https://pbs.twimg.com/media/DGGmoV4XsAAUL6n.jpg	1	
2074	892420643555336193	https://pbs.twimg.com/media/DGKD1-bXoAAIAUK.jpg	1	

2075 rows × 12 columns

In [7]:
```
# DogRate data dataframe
Rate_dogs_df = pd.read_csv("RateDogs.csv")
Rate_dogs_df.head()
```

Out[7]:

	created_at	id	text	in_reply_to_user_id_str	user	retweet_
0	Tue Aug 01 16:23:56 +0000 2017	892420643555336193	This is Phineas. He's a mystical boy. Only eve...	NaN	{'id': 4196983835, 'id_str': '4196983835', 'na...	
1	Tue Aug 01 00:17:27 +0000 2017	892177421306343426	This is Tilly. She's just checking pup on you....	NaN	{'id': 4196983835, 'id_str': '4196983835', 'na...	
2	Mon Jul 31 00:18:03 +0000 2017	891815181378084864	This is Archie. He is a rare Norwegian Pouncin...	NaN	{'id': 4196983835, 'id_str': '4196983835', 'na...	
3	Sun Jul 30 15:58:51 +0000 2017	891689557279858688	This is Darla. She commenced a snooze mid meal...	NaN	{'id': 4196983835, 'id_str': '4196983835', 'na...	
4	Sat Jul 29 16:00:24 +0000 2017	891327558926688256	This is Franklin. He would like you to stop ca...	NaN	{'id': 4196983835, 'id_str': '4196983835', 'na...	

In [8]:
```
# Resave dataframe as tweet_json.txt
df = pd.read_csv("RateDogs.csv")
df.to_csv('tweet_json.txt', encoding = 'utf-8')
```

Programmatically Assessment

In [9]:
```python
# WeRateDogs Twitter archive dataframe
tweet_archive_df.info()
```

```
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
```

In [10]:
```python
# Check duplicated columns
tweet_archive_df.duplicated().sum()
```

Out[10]:
```
0
```

In [11]:
```python
# Check values in the "rating_numerator" column
tweet_archive_df.rating_numerator.value_counts()
```

```
44         1
50         1
60         1

165        1
84         1
88         1
144        1
182        1
143        1
666        1
960        1
1776       1
17         1
27         1
45         1
99         1
121        1
204        1
Name: rating_numerator, dtype: int64
```

In [12]:

```
tweet_archive_df[tweet_archive_df['rating_numerator']== 420]
# 2074 (tweet_id = 670842764863651840) is not about dog, will be deleted
```

Out[12]:

	tweet_id	in_reply_to_status_id	in_reply_to_user_id	timestamp	
188	855862651834028034	8.558616e+17	194351775.0	2017-04-22 19:15:32 +0000	href="http://twi
2074	670842764863651840	NaN	NaN	2015-11-29 05:52:33 +0000	href="http://twi

In [13]:

```
tweet_archive_df[tweet_archive_df['rating_numerator']== 75]
# 340 and # 695 are the same dog
# 695(tweet_id = 786709082849828864)will be deleted manually
```

Out[13]:

	tweet_id	in_reply_to_status_id	in_reply_to_user_id	timestamp	
340	832215909146226688	NaN	NaN	2017-02-16 13:11:49 +0000	href="http://twitt
695	786709082849828864	NaN	NaN	2016-10-13 23:23:56 +0000	href="http://twitt

In [14]:

```
tweet_archive_df[tweet_archive_df['rating_numerator']== 960]
# 313 (tweet_id = 835246439529840640) didn't get a valid rating 960/00,
```

Out[14]:

	tweet_id	in_reply_to_status_id	in_reply_to_user_id	timestamp	
313	835246439529840640	8.352460e+17	26259576.0	2017-02-24 21:54:03 +0000	href="http://twitt

In [15]:
```python
tweet_archive_df[tweet_archive_df['tweet_id']== 775096608509886464] #784
```

Out[15]:

	tweet_id	in_reply_to_status_id	in_reply_to_user_id	timestamp	
784	775096608509886464	NaN	NaN	2016-09-11 22:20:06 +0000	href="http://twitt

In [16]:
```python
tweet_archive_df[tweet_archive_df['tweet_id']== 740373189193256964] #act
```

Out[16]:

	tweet_id	in_reply_to_status_id	in_reply_to_user_id	timestamp	
1068	740373189193256964	NaN	NaN	2016-06-08 02:41:38 +0000	href="http://twi

In [17]:
```python
tweet_archive_df[tweet_archive_df['tweet_id']== 682962037429899265] #act
```

Out[17]:

	tweet_id	in_reply_to_status_id	in_reply_to_user_id	timestamp	
1662	682962037429899265	NaN	NaN	2016-01-01 16:30:13 +0000	href="http://twi

In [18]:
```python
tweet_archive_df[tweet_archive_df['tweet_id']== 666287406224695296] #act
```

Out[18]:

	tweet_id	in_reply_to_status_id	in_reply_to_user_id	timestamp	
2335	666287406224695296	NaN	NaN	2015-11-16 16:11:11 +0000	href="http://twi

In [19]:
```
tweet_archive_df[tweet_archive_df['tweet_id']== 682808988178739200] #act
```
Out[19]:

	tweet_id	in_reply_to_status_id	in_reply_to_user_id	timestamp	
1663	682808988178739200	6.827884e+17	4.196984e+09	2016-01-01 06:22:03 +0000	href="http://twi

In [20]:
```
tweet_archive_df[tweet_archive_df['tweet_id']== 832088576586297345]  #no
```
Out[20]:

	tweet_id	in_reply_to_status_id	in_reply_to_user_id	timestamp	
342	832088576586297345	8.320875e+17	30582082.0	2017-02-16 04:45:50 +0000	href="http://twitt

In [21]:
```
tweet_archive_df[tweet_archive_df['tweet_id']== 810984652412424192]  #no
```
Out[21]:

	tweet_id	in_reply_to_status_id	in_reply_to_user_id	timestamp	
516	810984652412424192	NaN	NaN	2016-12-19 23:06:23 +0000	href="http://twitt

In [22]:
```
image_prediction_df.head()
```
Out[22]:

	tweet_id	jpg_url	img_num	
0	666020888022790149	https://pbs.twimg.com/media/CT4udn0WwAA0aMy.jpg	1	Welsh_sprin
1	666029285002620928	https://pbs.twimg.com/media/CT42GRgUYAA5iDo.jpg	1	
2	666033412701032449	https://pbs.twimg.com/media/CT4521TWwAEvMyu.jpg	1	Germa
3	666044226329800704	https://pbs.twimg.com/media/CT5Dr8HUEAA-lEu.jpg	1	Rhodesiai
4	666049248165822465	https://pbs.twimg.com/media/CT5IQmsXIAAKY4A.jpg	1	miniatu

In [23]:
```
# tweet image predictions dataframe
image_prediction_df.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
```

In [24]:
```
# Check duplicated rows
image_prediction_df.duplicated().sum()
```

Out[24]:
```
0
```

In [25]:
```
# Check duplicated image url
image_prediction_df['jpg_url'].duplicated().sum()
```

Out[25]:
```
66
```

In [26]:
```
# DogRate data dataframe
Rate_dogs_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2337 entries, 0 to 2336
Data columns (total 7 columns):
created_at                 2337 non-null object
id                         2337 non-null int64
text                       2337 non-null object
in_reply_to_user_id_str    77 non-null float64
user                       2337 non-null object
retweet_count              2337 non-null int64
favorite_count             2337 non-null int64
dtypes: float64(1), int64(3), object(3)
memory usage: 127.9+ KB
```

In [30]:
```
Rate_dogs_df.duplicated().sum()
```

Out[30]:
```
0
```

Quality

WeRateDogs Twitter archive dataframe

1. Delete the columns we don't need : "in_reply_to_status_id", 'in_reply_to_user_id ', "retweeted_status_id","source" 'retweeted_status_user_id','retweeted_status_timestamp'
2. Erroneous datatypes (doggo, floofer, pupper and puppo columns)
3. convert timestamp to datetime
4. Correct numerators and denominators
5. Delete some deleted twitter

tweet image predictions dataframe

1. Drop 66 jpg_url duplicated
2. Create 1 column for image prediction and 1 column for confidence level
3. Delete columns that won't be used for analysis
4. Extract dog_type and confidence from p1,p2,p3

DogRate data dataframe

1. Simplify column 'user'
2. Delete columns that won't be used for analysis

Tidiness

Convert three dataframe to one dataset

Cleaning Data

In [27]:
```
# WeRateDogs Twitter archive dataframe
tweet_archive_clean_df = tweet_archive_df.copy()
tweet_archive_clean_df.head(2)
```

Out[27]:

	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://twitter.
1	892177421306343426	NaN	NaN	2017-08-01 00:17:27 +0000	href="http://twitter.

In [33]:
```
# Delete unused column and add
tweet_archive_clean_df =tweet_archive_clean_df[['tweet_id','timestamp','
                                               'rating_denominator','na
```

In [34]:
```
tweet_archive_clean_df.head(2)
```

Out[34]:

	tweet_id	timestamp	text	expanded_urls	ratin
0	892420643555336193	2017-08-01 16:23:56 +0000	This is Phineas. He's a mystical boy. Only eve...	https://twitter.com/dog_rates/status/892420643...	
1	892177421306343426	2017-08-01 00:17:27 +0000	This is Tilly. She's just checking pup on you....	https://twitter.com/dog_rates/status/892177421...	

In [35]:

```python
# Fix Erroneous datatypes (doggo, floofer, pupper and puppo columns)

# Add one column 'dogs_stage' in tweet_archive_clean_df
# Conbine data of four columns 'doggo','floofer','pupper','puppo' in one
tweet_archive_clean_df['dogs_stage'] = tweet_archive_clean_df['doggo']+"
                                        tweet_archive_clean_df['pupper']
# Convert "None" to ""
for i in range(len(tweet_archive_clean_df)):
    if tweet_archive_clean_df.loc[i,'dogs_stage']=="None None None None"
        tweet_archive_clean_df.loc[i,'dogs_stage']="None"
    else:
        tweet_archive_clean_df.loc[i,'dogs_stage']=tweet_archive_clean_d
```

In [36]:

```python
tweet_archive_clean_df.dogs_stage.value_counts()
```

Out[36]:

```
None                 1976
pupper                245
doggo                  83
puppo                  29
doggo  pupper          12
floofer                 9
doggo floofer           1
doggo   puppo           1
Name: dogs_stage, dtype: int64
```

In [37]:

```python
# Drop four columns 'doggo','floofer','pupper','puppo'
tweet_archive_clean_df = tweet_archive_clean_df.drop(columns=['doggo','f
tweet_archive_clean_df.head(2)
```

Out[37]:

	tweet_id	timestamp	text	expanded_urls	ratin
0	892420643555336193	2017-08-01 16:23:56 +0000	This is Phineas. He's a mystical boy. Only eve...	https://twitter.com/dog_rates/status/892420643...	
1	892177421306343426	2017-08-01 00:17:27 +0000	This is Tilly. She's just checking pup on you....	https://twitter.com/dog_rates/status/892177421...	

In [38]:
```
# Correct numerators and denominators
# 313 manually change rating_numerator to 13, change rating_denominator
tweet_archive_clean_df.loc[313,'rating_numerator']=13
tweet_archive_clean_df.loc[313,'rating_denominator']=10
# 1068 actual rating 14/10 need to change manually
tweet_archive_clean_df.loc[1068,'rating_numerator']=14
tweet_archive_clean_df.loc[1068,'rating_denominator']=10
# 1662 actual rating 10/10 need to change manually
tweet_archive_clean_df.loc[1662,'rating_numerator']=10
tweet_archive_clean_df.loc[1662,'rating_denominator']=10
# 2335 actual rating 9/10 need to change manually
tweet_archive_clean_df.loc[2335,'rating_numerator']=9
tweet_archive_clean_df.loc[2335,'rating_denominator']=10
# 1663 actual rating 20/16 need to change manually
tweet_archive_clean_df.loc[1663,'rating_numerator']=20
tweet_archive_clean_df.loc[1663,'rating_denominator']=16


```

In [39]:
```
#Delete twitter
tweet_archive_clean_df = tweet_archive_clean_df[tweet_archive_clean_df["
tweet_archive_clean_df = tweet_archive_clean_df[tweet_archive_clean_df["
tweet_archive_clean_df = tweet_archive_clean_df[tweet_archive_clean_df["
tweet_archive_clean_df = tweet_archive_clean_df[tweet_archive_clean_df["
tweet_archive_clean_df = tweet_archive_clean_df[tweet_archive_clean_df["
```

In [40]:
```
#convert timestamp to datetime and change the column name to "date"
tweet_archive_clean_df['timestamp'] = pd.to_datetime(tweet_archive_clean
tweet_archive_clean_df = tweet_archive_clean_df.rename(columns={'timesta

```

In [41]:
```
tweet_archive_clean_df.head(2)
```

Out[41]:

	tweet_id	date	text	expanded_urls	rating_nur
0	892420643555336193	2017-08-01	This is Phineas. He's a mystical boy. Only eve...	https://twitter.com/dog_rates/status/892420643...	
1	892177421306343426	2017-08-01	This is Tilly. She's just checking pup on you....	https://twitter.com/dog_rates/status/892177421...	

In [17]:
```
# tweet image predictions dataframe
image_prediction_clean = image_prediction_df.copy()
image_prediction_clean.head()
```

Out[17]:

	tweet_id	jpg_url	img_num	
0	666020888022790149	https://pbs.twimg.com/media/CT4udn0WwAA0aMy.jpg	1	Welsh_sprin
1	666029285002620928	https://pbs.twimg.com/media/CT42GRgUYAA5iDo.jpg	1	
2	666033412701032449	https://pbs.twimg.com/media/CT4521TWwAEvMyu.jpg	1	Germa
3	666044226329800704	https://pbs.twimg.com/media/CT5Dr8HUEAA-lEu.jpg	1	Rhodesiaı
4	666049248165822465	https://pbs.twimg.com/media/CT5IQmsXIAAKY4A.jpg	1	miniatu

In [18]:
```
# Extract dog_type and confidence from p1,p2,p3
# Create two coloum "dog_type" and "confidence"
image_prediction_clean["dog_type"]=""
image_prediction_clean["confidence"]=""

for i in range(len(image_prediction_clean)):
    if image_prediction_clean.loc[i,"p1_dog"]==True:
        image_prediction_clean.loc[i,"dog_type"] = image_prediction_clea
        image_prediction_clean.loc[i,"confidence"] = image_prediction_cl
    elif image_prediction_clean.loc[i,"p2_dog"]==True:
        image_prediction_clean.loc[i,"dog_type"]=image_prediction_clean.
        image_prediction_clean.loc[i,"confidence"]=image_prediction_clea
    elif image_prediction_clean.loc[i,"p3_dog"]==True:
        image_prediction_clean.loc[i,"dog_type"]=image_prediction_clean.
        image_prediction_clean.loc[i,"confidence"]=image_prediction_clea
    else:
        image_prediction_clean.loc[i,"dog_type"]="None"
        image_prediction_clean.loc[i,"confidence"]="None"
```

In [19]:
```
# Drop columns we don't needed
image_prediction_clean = image_prediction_clean [['tweet_id','jpg_url','
image_prediction_clean.head()
```

Out[19]:

	tweet_id	jpg_url	img_num	
0	666020888022790149	https://pbs.twimg.com/media/CT4udn0WwAA0aMy.jpg	1	Welsh_sprin
1	666029285002620928	https://pbs.twimg.com/media/CT42GRgUYAA5iDo.jpg	1	
2	666033412701032449	https://pbs.twimg.com/media/CT4521TWwAEvMyu.jpg	1	Germa
3	666044226329800704	https://pbs.twimg.com/media/CT5Dr8HUEAA-lEu.jpg	1	Rhodesiaı
4	666049248165822465	https://pbs.twimg.com/media/CT5IQmsXIAAKY4A.jpg	1	miniatu

In [20]:
```
# Drop duplicated jpg url

image_prediction_clean = image_prediction_clean.drop_duplicates(subset='

# Recheck the duplicated jpg url rows
image_prediction_clean['jpg_url'].duplicated().sum()
```
Out[20]:
```
0
```

In [21]:
```
len(image_prediction_clean)
```
Out[21]:
```
2009
```

In [22]:
```
image_prediction_clean.head()
```
Out[22]:

	tweet_id	jpg_url	img_num	
0	666020888022790149	https://pbs.twimg.com/media/CT4udn0WwAA0aMy.jpg	1	Welsh_sprin
1	666029285002620928	https://pbs.twimg.com/media/CT42GRgUYAA5iDo.jpg	1	
2	666033412701032449	https://pbs.twimg.com/media/CT4521TWwAEvMyu.jpg	1	Germa
3	666044226329800704	https://pbs.twimg.com/media/CT5Dr8HUEAA-lEu.jpg	1	Rhodesiaı
4	666049248165822465	https://pbs.twimg.com/media/CT5IQmsXIAAKY4A.jpg	1	miniatu

In [23]:
```
image_prediction_clean.to_csv("dogtype.csv",index=False)
```

In [48]:
```
# DogRate data dataframe
Rate_dogs_clean = Rate_dogs_df.copy()
Rate_dogs_clean.head()
```

Out[48]:

	created_at	id	text	in_reply_to_user_id_str	user	retweet_
0	Tue Aug 01 16:23:56 +0000 2017	892420643555336193	This is Phineas. He's a mystical boy. Only eve...	NaN	{'id': 4196983835, 'id_str': '4196983835', 'na...	
1	Tue Aug 01 00:17:27 +0000 2017	892177421306343426	This is Tilly. She's just checking pup on you....	NaN	{'id': 4196983835, 'id_str': '4196983835', 'na...	
2	Mon Jul 31 00:18:03 +0000 2017	891815181378084864	This is Archie. He is a rare Norwegian Pouncin...	NaN	{'id': 4196983835, 'id_str': '4196983835', 'na...	
3	Sun Jul 30 15:58:51 +0000 2017	891689557279858688	This is Darla. She commenced a snooze mid meal...	NaN	{'id': 4196983835, 'id_str': '4196983835', 'na...	
4	Sat Jul 29 16:00:24 +0000 2017	891327558926688256	This is Franklin. He would like you to stop ca...	NaN	{'id': 4196983835, 'id_str': '4196983835', 'na...	

In [49]:
```
# Delete columns that won't be used for analysis
Rate_dogs_clean = Rate_dogs_clean[['id','retweet_count','favorite_count'
```

In [50]:
```
Rate_dogs_clean.head()
```

Out[50]:

	id	retweet_count	favorite_count
0	892420643555336193	8188	37547
1	892177421306343426	6058	32274
2	891815181378084864	4008	24314
3	891689557279858688	8343	40885
4	891327558926688256	9037	39085

Tidiness

In [51]:
```
#Convert tweet_id from str to int in Rate_dogs_clean dataframe
Rate_dogs_clean['id'] = Rate_dogs_clean['id'].astype(int)
# Rename 'id' to 'tweet_id'
Rate_dogs_clean = Rate_dogs_clean.rename(columns={'id':'tweet_id'})
Rate_dogs_clean.head()
```

Out[51]:

	tweet_id	retweet_count	favorite_count
0	892420643555336193	8188	37547
1	892177421306343426	6058	32274
2	891815181378084864	4008	24314
3	891689557279858688	8343	40885
4	891327558926688256	9037	39085

In [52]:
```
# Conbine three dataframes in one dataframe
Rate_dogs_combine = tweet_archive_clean_df.merge(image_prediction_clean,
Rate_dogs_combine = Rate_dogs_combine.merge(Rate_dogs_clean, how = 'left
```

In [53]:
```
Rate_dogs_combine.head()
```

Out[53]:

	tweet_id	date	text	expanded_urls	ratir
0	892420643555336193	2017-08-01	This is Phineas. He's a mystical boy. Only eve...	https://twitter.com/dog_rates/status/892420643...	
1	892177421306343426	2017-08-01	This is Tilly. She's just checking pup on you....	https://twitter.com/dog_rates/status/892177421...	
2	891815181378084864	2017-07-31	This is Archie. He is a rare Norwegian Pouncin...	https://twitter.com/dog_rates/status/891815181...	
			This is		

In [54]:
```
len(Rate_dogs_combine)
```

Out[54]:
```
2351
```

In [55]:
```
# Drop rows with missing data in column 'jpg_url'
Rate_dogs_combine = Rate_dogs_combine.dropna(subset=['jpg_url'])
# Drop column"img_num "
Rate_dogs_combine = Rate_dogs_combine.drop(columns=['img_num'])
```

In [56]:
```
Rate_dogs_combine.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2006 entries, 0 to 2350
Data columns (total 13 columns):
tweet_id              2006 non-null int64
date                  2006 non-null object
text                  2006 non-null object
expanded_urls         2006 non-null object
rating_numerator      2006 non-null int64
rating_denominator    2006 non-null int64
name                  2006 non-null object
dogs_stage            2006 non-null object
jpg_url               2006 non-null object
dog_type              2006 non-null object
confidence            2006 non-null object
retweet_count         2001 non-null float64
favorite_count        2001 non-null float64
dtypes: float64(2), int64(3), object(8)
memory usage: 219.4+ KB
```

Storing, Analyzing, and Visualizing

In [57]:
```
# Save dataframe as csv file
Rate_dogs_combine.to_csv("twitter_archive_master.csv", index= False)
```

In [58]:
```
Rate_dogs_combine_copy = Rate_dogs_combine.copy()
```

Insight one & visualization: Which breeds is the most common dog

In [59]:
```
# Drop row with empty value for column 'dog_type'
Rate_dogs_combine_copy_1 =Rate_dogs_combine_copy[Rate_dogs_combine_copy[

Rate_dogs_combine_copy_1['dog_type'].value_counts()
```

Out[59]:
```
golden_retriever                  157
Labrador_retriever                108
Pembroke                           95
Chihuahua                          91
pug                                63
toy_poodle                         51
chow                               48
Samoyed                            42
Pomeranian                         41
malamute                           33
Chesapeake_Bay_retriever           31
French_bulldog                     31
cocker_spaniel                     30
miniature_pinscher                 25
Eskimo_dog                         22
Staffordshire_bullterrier          21
Cardigan                           21
German_shepherd                    21
beagle                             20
Shih-Tzu                           20
Siberian_husky                     20
Maltese_dog                        19
Rottweiler                         19
Shetland_sheepdog                  19
kuvasz                             19
Lakeland_terrier                   18
Italian_greyhound                  17
basset                             17
West_Highland_white_terrier        16
American_Staffordshire_terrier     16
                                 ... 
bluetick                            4
Welsh_springer_spaniel              4
Tibetan_terrier                     4
Saluki                              4
giant_schnauzer                     4
Tibetan_mastiff                     4
toy_terrier                         3
Irish_water_spaniel                 3
Leonberg                            3
cairn                               3
komondor                            3
briard                              3
Greater_Swiss_Mountain_dog          3
Afghan_hound                        3
curly-coated_retriever              3
Brabancon_griffon                   3
black-and-tan_coonhound             2
wire-haired_fox_terrier             2
Australian_terrier                  2
Appenzeller                         2
Sussex_spaniel                      2
```

```
groenendael                    2
Irish_wolfhound                1
silky_terrier                  1
Bouvier_des_Flandres           1
standard_schnauzer             1
EntleBucher                    1
Japanese_spaniel               1
clumber                        1
Scotch_terrier                 1
Name: dog_type, Length: 113, dtype: int64
```

In [60]:

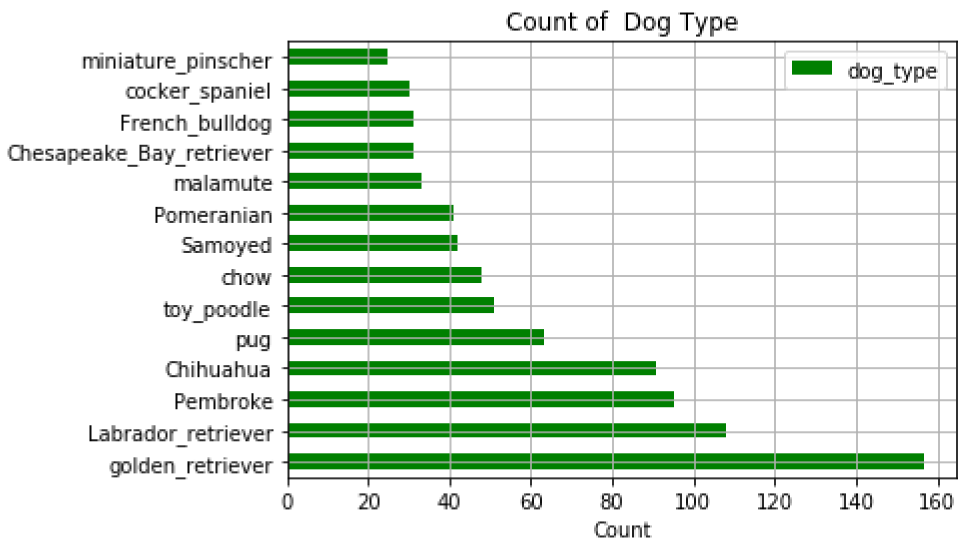
```python
# Create a bar plot

df_dog_type = pd.DataFrame(Rate_dogs_combine_copy_1['dog_type'].value_co
df_dog_type = df_dog_type[df_dog_type['dog_type']>=25].sort_values(by='d

df_dog_type.plot(kind = 'barh',color = "green")
# Set title x-axis label
plt.title('Count of  Dog Type')
plt.xlabel('Count')

# Set grid
plt.grid()


plt.savefig('Dog Type.png')
```



golden_retriever is the most commom bread in this dataset.

Insight two: Retweets and Favorites with Time

In [61]:

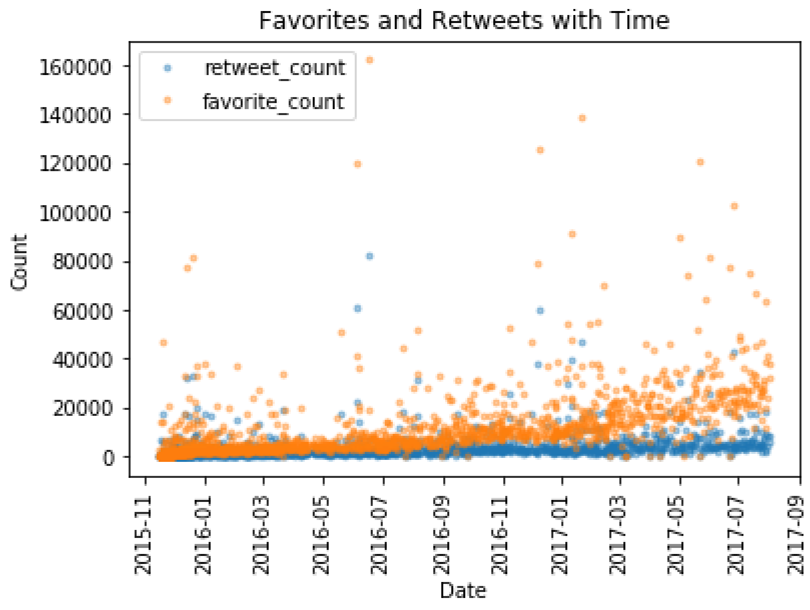
```
Rate_dogs_combine_copy_2 = Rate_dogs_combine_copy[['date','retweet_count
Rate_dogs_combine_copy_2 = Rate_dogs_combine_copy_2.set_index('date')
Rate_dogs_combine_copy_2.head()
```

Out[61]:

	retweet_count	favorite_count
date		
2017-08-01	8188.0	37547.0
2017-08-01	6058.0	32274.0
2017-07-31	4008.0	24314.0
2017-07-30	8343.0	40885.0
2017-07-29	9037.0	39085.0

In [62]:

```
# Create a scatter plot
Rate_dogs_combine_copy_2.plot(style = '.', alpha = 0.4)
plt.title('Favorites and Retweets with Time')
plt.xlabel('Date')
plt.xticks(rotation=90)
plt.ylabel('Count')
plt.savefig('Favorites and Retweets.png')
```



In [79]:
```
Rate_dogs_combine_copy_2.describe()
```

Out[79]:

	retweet_count	favorite_count
count	2001.000000	2001.000000
mean	2584.443778	8524.062969
std	4645.593911	12599.738516
min	11.000000	0.000000
25%	570.000000	1783.000000
50%	1241.000000	3823.000000
75%	2922.000000	10618.000000
max	82461.000000	162145.000000

Insight three & visualization: Rating and Retweet

In [75]:
```
Rate_dogs_combine_copy_3 = Rate_dogs_combine_copy[['retweet_count','rati
# Drop rows including null value
Rate_dogs_combine_copy_3 = Rate_dogs_combine_copy_3.dropna(how='any')
Rate_dogs_combine_copy_3['rating']=Rate_dogs_combine_copy_3['rating_nume
```

In [77]:
```
# the tweeter with the highest rating get 2604 retweeter
Rate_dogs_combine_copy_3[Rate_dogs_combine_copy_3['rating']==177.6]
```

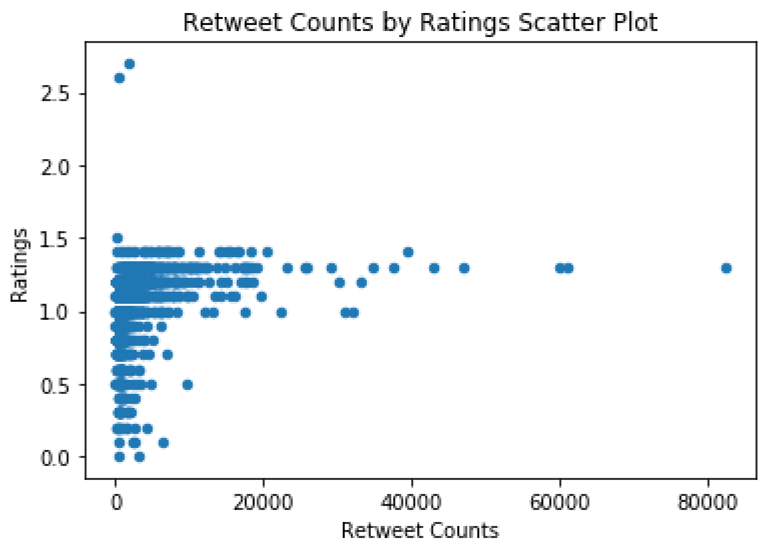
Out[77]:

	retweet_count	rating_numerator	rating_denominator	rating
975	2604.0	1776	10	177.6

In [67]:
```
# Exclude the highest rating tweeter
Rate_dogs_combine_copy_4= Rate_dogs_combine_copy_3[Rate_dogs_combine_cop
```

In [69]:
```python
# Generate scatter plot
Rate_dogs_combine_copy_4.plot(x='retweet_count', y='rating', kind='scatt
plt.xlabel('Retweet Counts')
plt.ylabel('Ratings')
plt.title('Retweet Counts by Ratings Scatter Plot')

fig = plt.gcf()
plt.savefig("Rating and Retweet.png")
```



In [ ]:
```python

```