In [151]:
```
#Import useful libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
% matplotlib inline
import seaborn as sns
import requests
import os
import glob
import tweepy
import json
```

Gather

In [152]:
```
# Load twitter-archive-enhanced.csv into twitter_archive_df
twitter_archive_df=pd.read_csv('twitter-archive-enhanced.csv')
```

In [153]:
```
#save the image-predictions.tsv file and load it into image_pred_df
import requests
url='https://d17h27t6h515a5.cloudfront.net/topher/2017/August/599fd2ad_image-predictions/ir
result=requests.get(url)
```

In [154]:
```
with open('image-predictions.tsv',mode='wb') as file:
    file.write(result.content)
```

In [155]:
```
image_pred_df=pd.read_csv('image-predictions.tsv',sep='\t')
```

```
In [8]: #using Python's Tweepy library and store each tweet's entire set of JSON
        #data in a file called tweet_json.txt fil

        consumer_key='jL4Tb7YNEg5OjrQ03QsGQ0RxP'
        consumer_secret='14MjRVAuAtnCdOIY2xOAgqTbWVzzbeFEHXMg5L7oTtnGMxiX4f'
        access_token='1275518990278950912-XZIStE7P2HJaGy0n7WRlnmX8X1HB3U'
        access_secret='52t7kpuqBYKqN5CIp8XNyK4is4KviGPC5Qli09zVWwuOM'

        auth=tweepy.OAuthHandler(consumer_key,consumer_secret)
        auth.set_access_token(access_token,access_secret)
        api=tweepy.API(auth,wait_on_rate_limit=True,wait_on_rate_limit_notify=True)



        page_no_exist=[]
        retweet_count_and_favorite_count=[]

        with open('tweet_json.txt',mode='w')as file:
            for i in list(twitter_archive_df.tweet_id):
                try:
                    tweet=api.get_status(i)
                    file.write(json.dumps(tweet._json))
                    retweet_count_and_favorite_count.append({"tweet_id":i,
                                                            "retweet_count": tweet._json['retweet_
                                                            "favorite_count": tweet._json['favorit


                except:
                    page_no_exist.append(i)
```

```
Rate limit reached. Sleeping for: 692
Rate limit reached. Sleeping for: 736
```

```
In [156]: len(retweet_count_and_favorite_count),len(page_no_exist)
```

```
Out[156]: (2331, 25)
```

```
In [157]: # convert list into dataframe
          retweet_count_and_favorite_count=pd.DataFrame(retweet_count_and_favorite_count,
                                                        columns=['tweet_id','retweet_count','favorit
```

Assess

```
In [158]: # Mannual assessement of data
          twitter_archive_df
```

Out[158]:

	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.com/downlo
1	892177421306343426	NaN	NaN	2017-08-01 00:17:27 +0000	href="http://twitter.com/downlo
2	891815181378084864	NaN	NaN	2017-07-31 00:18:03 +0000	href="http://twitter.com/downlo
3	891689557279858688	NaN	NaN	2017-07-30 15:58:51 +0000	href="http://twitter.com/downlo

```
In [159]: # Display top 5 rows
          twitter_archive_df.head(5)
```

Out[159]:

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

In [160]:
```
# size of dataframe
twitter_archive_df.shape
```
Out[160]:
```
(2356, 17)
```

In [161]:
```
# Get duplicate data
sum(twitter_archive_df.duplicated())
```
Out[161]:
```
0
```

In [162]:
```
sum(twitter_archive_df.source.duplicated())
```
Out[162]:
```
2352
```

In [163]:
```
twitter_archive_df.info()
```
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2356 entries, 0 to 2355
Data columns (total 17 columns):
tweet_id                      2356 non-null int64
in_reply_to_status_id         78 non-null float64
in_reply_to_user_id           78 non-null float64
timestamp                     2356 non-null object
source                        2356 non-null object
text                          2356 non-null object
retweeted_status_id           181 non-null float64
retweeted_status_user_id      181 non-null float64
retweeted_status_timestamp    181 non-null object
expanded_urls                 2297 non-null object
rating_numerator              2356 non-null int64
rating_denominator            2356 non-null int64
name                          2356 non-null object
doggo                         2356 non-null object
floofer                       2356 non-null object
pupper                        2356 non-null object
puppo                         2356 non-null object
dtypes: float64(4), int64(3), object(10)
memory usage: 313.0+ KB
```

In [164]:
```
twitter_archive_df.describe()
```
Out[164]:

	tweet_id	in_reply_to_status_id	in_reply_to_user_id	retweeted_status_id	retweeted_status_user_id	r
count	2.356000e+03	7.800000e+01	7.800000e+01	1.810000e+02	1.810000e+02	
mean	7.427716e+17	7.455079e+17	2.014171e+16	7.720400e+17	1.241698e+16	
std	6.856705e+16	7.582492e+16	1.252797e+17	6.236928e+16	9.599254e+16	
min	6.660209e+17	6.658147e+17	1.185634e+07	6.661041e+17	7.832140e+05	
25%	6.783989e+17	6.757419e+17	3.086374e+08	7.186315e+17	4.196984e+09	
50%	7.196279e+17	7.038708e+17	4.196984e+09	7.804657e+17	4.196984e+09	
75%	7.993373e+17	8.257804e+17	4.196984e+09	8.203146e+17	4.196984e+09	
max	8.924206e+17	8.862664e+17	8.405479e+17	8.874740e+17	7.874618e+17	

In [165]:
```
# Get null values
twitter_archive_df.isnull().sum()
```

Out[165]:
```
tweet_id                         0
in_reply_to_status_id         2278
in_reply_to_user_id           2278
timestamp                        0
source                           0
text                             0
retweeted_status_id           2175
retweeted_status_user_id      2175
retweeted_status_timestamp    2175
expanded_urls                   59
rating_numerator                 0
rating_denominator               0
name                             0
doggo                            0
floofer                          0
pupper                           0
puppo                            0
dtype: int64
```

```
In [166]: # Manual assessement of dataframe
          image_pred_df
```

Out[166]:

	tweet_id	jpg_url	img_num	p
0	666020888022790149	https://pbs.twimg.com/media/CT4udn0WwAA0aMy.jpg	1	Welsh_springer_spanie
1	666029285002620928	https://pbs.twimg.com/media/CT42GRgUYAA5iDo.jpg	1	redbone
2	666033412701032449	https://pbs.twimg.com/media/CT4521TWwAEvMyu.jpg	1	German_shepher
3	666044226329800704	https://pbs.twimg.com/media/CT5Dr8HUEAA-lEu.jpg	1	Rhodesian_ridgebac
4	666049248165822465	https://pbs.twimg.com/media/CT5IQmsXIAAKY4A.jpg	1	miniature_pinsche
5	666050758794694657	https://pbs.twimg.com/media/CT5Jof1WUAEuVxN.jpg	1	Bernese_mountain_do
6	666051853826850816	https://pbs.twimg.com/media/CT5KoJ1WoAAJash.jpg	1	box_turtl
7	666055525042405380	https://pbs.twimg.com/media/CT5N9tpXIAAifs1.jpg	1	chov
8	666057090499244032	https://pbs.twimg.com/media/CT5PY90WoAAQGLo.jpg	1	shopping_car
9	666058600524156928	https://pbs.twimg.com/media/CT5Qw94XAAA_2dP.jpg	1	miniature_poodl
10	666063827256086533	https://pbs.twimg.com/media/CT5Vg_wXIAAXfnj.jpg	1	golden_retrieve
11	666071193221509120	https://pbs.twimg.com/media/CT5cN_3WEAAlOoZ.jpg	1	Gordon_sette
12	666073100786774016	https://pbs.twimg.com/media/CT5d9DZXAAALcwe.jpg	1	Walker_houn
13	666082916733198337	https://pbs.twimg.com/media/CT5m4VGWEAAtKc8.jpg	1	pu
14	666094000022159362	https://pbs.twimg.com/media/CT5w9gUW4AAsBNN.jpg	1	bloodhoun
15	666099513787052032	https://pbs.twimg.com/media/CT51-JJUEAA6hV8.jpg	1	Lhas
16	666102155909144576	https://pbs.twimg.com/media/CT54YGiWUAEZnoK.jpg	1	English_sette
17	666104133288665088	https://pbs.twimg.com/media/CT56LSZWoAAlJj2.jpg	1	he
18	666268910803644416	https://pbs.twimg.com/media/CT8QCd1WEAADXws.jpg	1	desktop_compute
19	666273097616637952	https://pbs.twimg.com/media/CT8T1mtUwAA3aqm.jpg	1	Italian_greyhoun
20	666287406224695296	https://pbs.twimg.com/media/CT8g3BpUEAAuFjg.jpg	1	Maltese_do
21	666293911632134144	https://pbs.twimg.com/media/CT8mx7KW4AEQu8N.jpg	1	three-toed_sloth
22	666337882303524864	https://pbs.twimg.com/media/CT9OwFIWEAMuRje.jpg	1	o
23	666345417576210432	https://pbs.twimg.com/media/CT9Vn7PWoAA_ZCM.jpg	1	golden_retrieve
24	666353288456101888	https://pbs.twimg.com/media/CT9cx0tUEAAhNN_.jpg	1	malamut
25	666362758909284353	https://pbs.twimg.com/media/CT9lXGsUcAAyUFt.jpg	1	guinea_pi
26	666373753744588802	https://pbs.twimg.com/media/CT9vZEYWUAAlZ05.jpg	1	soft coated_wheaten_terrie
27	666396247373291520	https://pbs.twimg.com/media/CT-D2ZHWIAA3gK1.jpg	1	Chihuahu
28	666407126856765440	https://pbs.twimg.com/media/CT-NvwmW4AAugGZ.jpg	1	black-and-tan_coonhoun
29	666411507551481857	https://pbs.twimg.com/media/CT-RugiWIAELEaq.jpg	1	coh
...	...	...	...	..
2045	886366144734445568	https://pbs.twimg.com/media/DE0BTnQUwAApKEH.jpg	1	French_bulldo
2046	886680336477933568	https://pbs.twimg.com/media/DE4fEDzWAAAyHMM.jpg	1	convertibl
2047	886736880519319552	https://pbs.twimg.com/media/DE5Se8FXcAAJFx4.jpg	1	kuvas
2048	886983233522544640	https://pbs.twimg.com/media/DE8yicJW0AAAvBJ.jpg	2	Chihuahu

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

2075 rows × 12 columns

```
In [167]: #Get random 5 rows of dataset
          image_pred_df.sample(5)
```

Out[167]:

	tweet_id	jpg_url	img_num	p1	p1_conf
286	671151324042559489	https://pbs.twimg.com/media/CVBokRSWsAADuXx.jpg	1	Rottweiler	0.781201
657	682303737705140231	https://pbs.twimg.com/media/CXgHoLnWAAA8i52.jpg	1	seat_belt	0.997659
228	670385711116361728	https://pbs.twimg.com/media/CU2wPyWWUAAb1MJ.jpg	1	whippet	0.178027
1003	708834316713893888	https://pbs.twimg.com/media/CdZI_bpWEAAm1fs.jpg	1	Eskimo_dog	0.283945
1137	728986383096946689	https://pbs.twimg.com/media/Ch3hOGWUYAE7w0y.jpg	2	Maltese_dog	0.952070

In [168]:

```
# Get manual assessement of dataset
retweet_count_and_favorite_count
```

Out[168]:

	tweet_id	retweet_count	favorite_count
0	892420643555336193	7468	35365
1	892177421306343426	5545	30612
2	891815181378084864	3670	23028
3	891689557279858688	7635	38632
4	891327558926688256	8243	36905
5	891087950875897856	2754	18609
6	890971913173991426	1791	10807
7	890729181411237888	16705	59550
8	890609185150312448	3813	25624
9	890240255349198849	6479	29233
10	890006608113172480	6495	28193
11	889880896479866881	4412	25652
12	889665388333682689	8854	44033
13	889638837579907072	3964	24754
14	889531135344209921	1998	13933
15	889278841981685760	4714	23102
16	888917238123831296	3972	26713
17	888804989199671297	3744	23462
18	888554962724278272	3061	18085
19	888078434458587136	3072	19982
20	887705289381826560	4780	27792
21	887517139158093824	10426	42559
22	887473957103951883	15916	62976
23	887343217045368832	9324	30887
24	887101392804085760	5278	28113
25	886983233522544640	6771	31949
26	886736880519319552	2821	10972
27	886680336477933568	3962	20651
28	886366144734445568	2802	19384
29	886267009285017600	4	110
...	...	...	...
2301	666411507551481857	286	396
2302	666407126856765440	31	98
2303	666396247373291520	73	155
2304	666373753744588802	78	169

	tweet_id	retweet_count	favorite_count
2305	666362758909284353	502	703
2306	666353288456101888	64	193
2307	666345417576210432	128	267
2308	666337882303524864	81	179
2309	666293911632134144	312	450
2310	666287406224695296	57	131
2311	666273097616637952	70	157
2312	666268910803644416	32	94
2313	666104133288665088	5811	13292
2314	666102155909144576	11	69
2315	666099513787052032	57	140
2316	666094000022159362	66	153
2317	666082916733198337	41	101
2318	666073100786774016	141	284
2319	666071193221509120	52	135
2320	666063827256086533	191	439
2321	666058600524156928	51	103
2322	666057090499244032	120	263
2323	666055525042405380	214	404
2324	666051853826850816	752	1098
2325	666050758794694657	51	121
2326	666049248165822465	39	96
2327	666044226329800704	124	265
2328	666033412701032449	39	109
2329	666029285002620928	41	119
2330	666020888022790149	448	2355

2331 rows × 3 columns

In [169]:
```
retweet_count_and_favorite_count.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2331 entries, 0 to 2330
Data columns (total 3 columns):
tweet_id          2331 non-null int64
retweet_count     2331 non-null int64
favorite_count    2331 non-null int64
dtypes: int64(3)
memory usage: 54.7 KB
```

In [170]:
```
retweet_count_and_favorite_count.describe()
```

Out[170]:

	tweet_id	retweet_count	favorite_count
count	2.331000e+03	2331.000000	2331.000000
mean	7.419079e+17	2624.619477	7387.945517
std	6.823170e+16	4439.925583	11473.145651
min	6.660209e+17	1.000000	0.000000
25%	6.782670e+17	533.000000	1284.000000
50%	7.182469e+17	1226.000000	3209.000000
75%	7.986692e+17	3044.500000	9037.500000
max	8.924206e+17	75421.000000	152412.000000

Quality

twitter_archive_df

- redundant retweeted rows
- redundant columns('in_reply_to_status_id','in_reply_to_user_id')
- wrong rating numerator in some rows
- some denominators are not equal to 10
- some name are given 'a' and 'an' instead of none
- wrong data types of 'timestamp','tweet_id'
- 'source' column has 2352 duplicated data
- image_pred_df*
- many prediction contains 3 false in image_pred_df which means those are useless data
- *tidiness**
- there should be only one column 'stage' instead of for columns 'doggo','floofer','pupper','puppo'
- retweet_count_and_favorite_count should be part of twitter_archive_df

-

Clean

In [171]:
```
# Copy original dataset before cleaning
twitter_clean=twitter_archive_df.copy()
image_clean=image_pred_df.copy()
retweet_count_and_favorite_count_clean=retweet_count_and_favorite_count.copy()
```

Step1:address completeness issue

No missing important data

Step2:address tidiness issues

- there should be only one column 'stage' instead of for columns 'doggo','floofer','pupper','puppo'

Define

Convert all four columns ['doggo','floofer','pupper','puppo'] into one 'stage' column and then drop the four columns

Code

In [172]:
```
twitter_clean['stage']=twitter_clean['doggo']+twitter_clean['floofer']+twitter_clean['puppe
```

In [173]:
```
twitter_clean.stage.value_counts()
```

Out[173]:
```
NoneNoneNoneNone        1976
NoneNonepupperNone       245
doggoNoneNoneNone         83
NoneNoneNonepuppo         29
doggoNonepupperNone       12
NoneflooferNoneNone        9
doggoflooferNoneNone       1
doggoNoneNonepuppo         1
Name: stage, dtype: int64
```

In [174]:
```
twitter_clean['stage']=twitter_clean['stage'].map(lambda x:x.replace('None',''))
```

In [175]:
```
twitter_clean.stage.value_counts()
```

Out[175]:
```
                1976
pupper           245
doggo             83
puppo             29
doggopupper       12
floofer            9
doggofloofer       1
doggopuppo         1
Name: stage, dtype: int64
```

In [176]:
```
twitter_clean.loc[twitter_clean['stage']=='doggopupper','stage']='doggo,pupper'
twitter_clean.loc[twitter_clean['stage']=='doggopuppo','stage']='doggo,puppo'
twitter_clean.loc[twitter_clean['stage']=='doggofloofer','stage']='doggo,floofer'
```

In [177]:
```
twitter_clean.drop(['doggo','floofer','pupper','puppo'],axis=1,inplace=True)
```

Test

In [178]:
```
twitter_clean.stage.value_counts()
```

Out[178]:
```
                 1976
pupper            245
doggo              83
puppo              29
doggo,pupper       12
floofer             9
doggo,floofer       1
doggo,puppo         1
Name: stage, dtype: int64
```

```
    - retweet_count_and_favorite_count should be part of twitter_archive_df
```

Define - Merge retweet_count_and_favorite_count with twitter_archive on tweet_id

Code

In [179]:
```
twitter_clean.tweet_id=twitter_clean.tweet_id.astype(str)
retweet_count_and_favorite_count_clean.tweet_id=retweet_count_and_favorite_count_clean.twee
twitter_clean=pd.merge(twitter_clean,retweet_count_and_favorite_count_clean,on=['tweet_id']
```

Test

In [180]:
```
twitter_clean.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2356 entries, 0 to 2355
Data columns (total 16 columns):
tweet_id                      2356 non-null object
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
stage                         2356 non-null object
retweet_count                 2331 non-null float64
favorite_count                2331 non-null float64
dtypes: float64(6), int64(2), object(8)
memory usage: 312.9+ KB
```

```
In [181]: twitter_clean.retweet_count.describe()
```

```
Out[181]: count     2331.000000
          mean      2624.619477
          std       4439.925583
          min          1.000000
          25%        533.000000
          50%       1226.000000
          75%       3044.500000
          max      75421.000000
          Name: retweet_count, dtype: float64
```

```
In [182]: twitter_clean.favorite_count.describe()
```

```
Out[182]: count       2331.000000
          mean        7387.945517
          std        11473.145651
          min            0.000000
          25%         1284.000000
          50%         3209.000000
          75%         9037.500000
          max       152412.000000
          Name: favorite_count, dtype: float64
```

Step3: address quality issues

- redundant retweeted rows

Define

- Delete redundant retweeted columns'retweeted_status_id','retweeted_status_user_id','retweeted_status_timestamp'

Code

```
In [183]: twitter_clean.drop(['retweeted_status_id','retweeted_status_user_id','retweeted_status_time
```

Test

```
In [184]: twitter_clean.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2356 entries, 0 to 2355
Data columns (total 13 columns):
tweet_id                 2356 non-null object
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
expanded_urls            2297 non-null object
rating_numerator         2356 non-null int64
rating_denominator       2356 non-null int64
name                     2356 non-null object
stage                    2356 non-null object
retweet_count            2331 non-null float64
favorite_count           2331 non-null float64
dtypes: float64(4), int64(2), object(7)
memory usage: 257.7+ KB
```

- redundant columns('in_reply_to_status_id','in_reply_to_user_id')

Define

- Drop redundant columns ['in_reply_to_status_id','in_reply_to_user_id']

Code

```
In [185]: twitter_clean.drop(['in_reply_to_status_id','in_reply_to_user_id'],axis=1,inplace=True)
```

Test

```
In [186]: twitter_clean.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2356 entries, 0 to 2355
Data columns (total 11 columns):
tweet_id              2356 non-null object
timestamp             2356 non-null object
source                2356 non-null object
text                  2356 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
stage                 2356 non-null object
retweet_count         2331 non-null float64
favorite_count        2331 non-null float64
dtypes: float64(2), int64(2), object(7)
memory usage: 220.9+ KB
```

- wrong rating numerator in some rows

define

- Drop the rows where numerator ratings are greater than 20

Code

In [187]:
```
twitter_clean.drop(twitter_clean[twitter_clean.rating_numerator >=20].index,inplace=True)
```

Test

In [41]:
```
twitter_clean.rating_numerator.value_counts()
```

Out[41]:
```
12    558
11    464
10    461
13    351
9     158
8     102
7      55
14     54
5      37
6      32
3      19
4      17
1       9
2       9
0       2
15      2
17      1
Name: rating_numerator, dtype: int64
```

- some denominators are not equal to 10

Define

- Delete the rows where denominators are not equal to 10

Code

In [42]:
```
twitter_clean.drop(twitter_clean[twitter_clean.rating_denominator !=10].index,inplace=True
```

Test

In [43]:
```
twitter_clean.rating_denominator.value_counts()
```

Out[43]:
```
10    2324
Name: rating_denominator, dtype: int64
```

- some name are given 'a' and 'an' instead of none

Define

- Replace 'a' and 'an' with 'None' and then drop all the none values

Code

```
In [44]: twitter_clean.name.replace({'a':'None',
                                     'an':'None'},inplace=True)
```

Test

```
In [45]: twitter_clean.name.value_counts()
```

```
Out[45]: None        783
         Charlie      12
         Lucy         11
         Oliver       11
         Cooper       11
         Tucker       10
         Penny        10
         Lola         10
         Winston       9
         Bo            9
         the           8
         Sadie         8
         Daisy         7
         Bailey        7
         Toby          7
         Buddy         7
         Rusty         6
         Scout         6
         Stanley       6
         Milo          6
         Koda          6
         Oscar         6
         Dave          6
         Leo           6
         Bella         6
         Jack          6
         Jax           6
         Alfie         5
         George        5
         very          5
                    ...
         Atticus       1
         Tater         1
         Karll         1
         Yukon         1
         Tuck          1
         Jessiga       1
         Lacy          1
         Milky         1
         Jangle        1
         Jim           1
         Chef          1
         Mabel         1
         Ruffles       1
         Willow        1
         Swagger       1
         Tove          1
         Tango         1
         Malikai       1
         Spanky        1
         General       1
         Creg          1
         Tiger         1
         Tripp         1
         Ron           1
         Clarkus       1
         Jay           1
```

```
Meatball      1
Socks         1
Simba         1
Herb          1
Name: name, Length: 951, dtype: int64
```

- wrong data types of 'timestamp','tweet_id'

Define

- convert 'timestamp' datatype from 'object' to 'datetime'

Code

```
In [46]: twitter_clean.timestamp=twitter_clean.timestamp.astype('datetime64')
```

Test

```
In [47]: twitter_clean.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2324 entries, 0 to 2355
Data columns (total 11 columns):
tweet_id              2324 non-null object
timestamp             2324 non-null datetime64[ns]
source                2324 non-null object
text                  2324 non-null object
expanded_urls         2272 non-null object
rating_numerator      2324 non-null int64
rating_denominator    2324 non-null int64
name                  2324 non-null object
stage                 2324 non-null object
retweet_count         2300 non-null float64
favorite_count        2300 non-null float64
dtypes: datetime64[ns](1), float64(2), int64(2), object(6)
memory usage: 217.9+ KB
```

- 'source' column has 2352 duplicated data

Define

- Drop 'source' column as 95% data are duplicated

Code

```
In [48]: twitter_clean.drop('source',axis=1,inplace=True)
```

Test

In [49]:
```
twitter_clean.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2324 entries, 0 to 2355
Data columns (total 10 columns):
tweet_id              2324 non-null object
timestamp             2324 non-null datetime64[ns]
text                  2324 non-null object
expanded_urls         2272 non-null object
rating_numerator      2324 non-null int64
rating_denominator    2324 non-null int64
name                  2324 non-null object
stage                 2324 non-null object
retweet_count         2300 non-null float64
favorite_count        2300 non-null float64
dtypes: datetime64[ns](1), float64(2), int64(2), object(5)
memory usage: 199.7+ KB
```

- many prediction contains 3 false in image_pred_df which means those are useless data

Define

- find rows that have 3 False and Drop them

Code

In [50]:
```
False3=image_clean.query('p1_dog==False and p2_dog ==False and p3_dog==False').index
```

In [51]:
```
False3.shape[0]
```

Out[51]:
```
324
```

In [52]:
```
image_clean.drop(index=False3,inplace=True)
```

Test

In [53]:
```
image_clean.query('p1_dog==False and p2_dog ==False and p3_dog==False').index
```

Out[53]:
```
Int64Index([], dtype='int64')
```

Final Step:Store Data

In [54]:
```
# Store wrangle dataset twitter_clean into twitter_archive_master.csv
twitter_clean.to_csv('twitter_archive_master.csv',index=False)
```

In [55]:
```
# Store wrangle dataset image_clean into Image_pre_wrangled.csv
image_clean.to_csv('Image_pre_wrangled.csv',index=False)
```

Data Analyzing

In [56]:
```python
# Load twitter_archive_master.csv into df dataframe
df=pd.read_csv('twitter_archive_master.csv')
# Load Image_pre_wrangled.csv into df dataframe
df_image=pd.read_csv('Image_pre_wrangled.csv')
```

In [57]:
```python
# Display top 5 rows of th dataset
df.head()
```

Out[57]:

	tweet_id	timestamp	text	expanded_urls	rating_numerator	r
0	892420643555336193	2017-08-01 16:23:56	This is Phineas. He's a mystical boy. Only eve...	https://twitter.com/dog_rates/status/892420643...	13	
1	892177421306343426	2017-08-01 00:17:27	This is Tilly. She's just checking pup on you....	https://twitter.com/dog_rates/status/892177421...	13	
2	891815181378084864	2017-07-31 00:18:03	This is Archie. He is a rare Norwegian Pouncin...	https://twitter.com/dog_rates/status/891815181...	12	
3	891689557279858688	2017-07-30 15:58:51	This is Darla. She commenced a snooze mid meal...	https://twitter.com/dog_rates/status/891689557...	13	
4	891327558926688256	2017-07-29 16:00:24	This is Franklin. He would like you to stop ca...	https://twitter.com/dog_rates/status/891327558...	12	

In [58]:
```python
# Display top 5 rows of th dataset
df_image.head()
```

Out[58]:

	tweet_id	jpg_url	img_num	p1	p1_c
0	666020888022790149	https://pbs.twimg.com/media/CT4udn0WwAA0aMy.jpg	1	Welsh_springer_spaniel	0.465
1	666029285002620928	https://pbs.twimg.com/media/CT42GRgUYAA5iDo.jpg	1	redbone	0.506
2	666033412701032449	https://pbs.twimg.com/media/CT4521TWwAEvMyu.jpg	1	German_shepherd	0.596
3	666044226329800704	https://pbs.twimg.com/media/CT5Dr8HUEAA-lEu.jpg	1	Rhodesian_ridgeback	0.408
4	666049248165822465	https://pbs.twimg.com/media/CT5IQmsXIAAKY4A.jpg	1	miniature_pinscher	0.56

In [59]:
```
#Drop the column we do not need ['timestamp','text','expanded_url','name']
df.drop(['timestamp','text','expanded_urls'],axis=1,inplace=True)
# Confirm the change
df.head()
```

Out[59]:

	tweet_id	rating_numerator	rating_denominator	name	stage	retweet_count	favorite_count
0	892420643555336193	13	10	Phineas	NaN	7468.0	35365.0
1	892177421306343426	13	10	Tilly	NaN	5545.0	30612.0
2	891815181378084864	12	10	Archie	NaN	3670.0	23028.0
3	891689557279858688	13	10	Darla	NaN	7635.0	38632.0
4	891327558926688256	12	10	Franklin	NaN	8243.0	36905.0

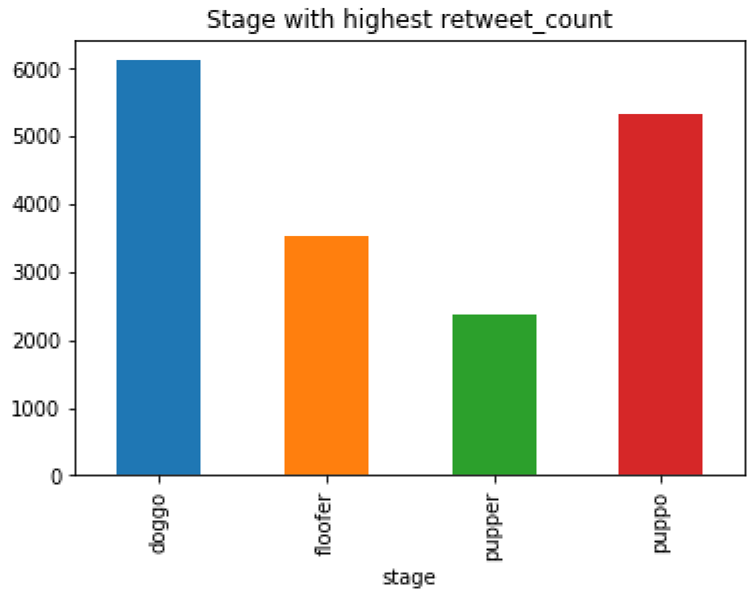
Explotary Data Analysis

Research Question 1: Which stage of dog got highest retweet_count and favorite_count

In [60]:
```
df_stage=df.query('stage=="doggo" or stage=="floofer" or stage=="pupper" or stage=="puppo"
df_stage.groupby('stage').retweet_count.mean()
```

Out[60]:
```
stage
doggo      6120.775000
floofer    3535.222222
pupper     2365.814050
puppo      5335.413793
Name: retweet_count, dtype: float64
```

In [98]:
```
df_stage.groupby('stage').retweet_count.mean().plot(kind='bar',title='Stage with highest re
```

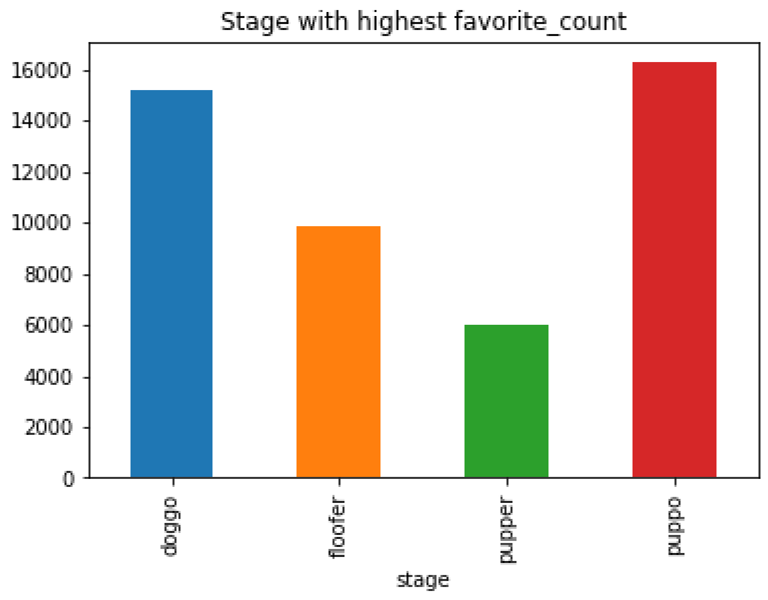


```
In [149]: df_stage.groupby('stage').favorite_count.mean()
```

```
Out[149]: stage
          doggo      15225.525000
          floofer     9865.111111
          pupper      5970.243802
          puppo      16267.758621
          Name: favorite_count, dtype: float64
```

```
In [150]: df_stage.groupby('stage').favorite_count.mean().plot(kind='bar',title='Stage with highest
```

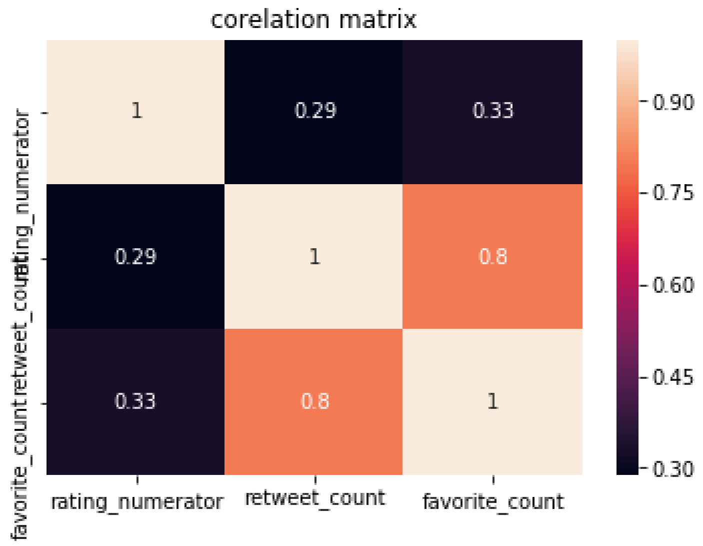
```
Out[150]: <matplotlib.axes._subplots.AxesSubplot at 0x7fc67cc4a048>
```



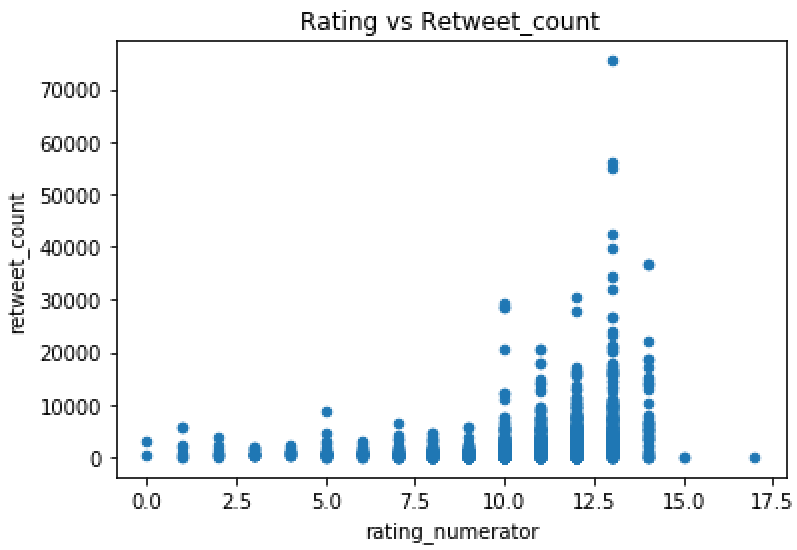
- The stage of doggo and puppo got the highest retweet_count and favorite_count respectively.*

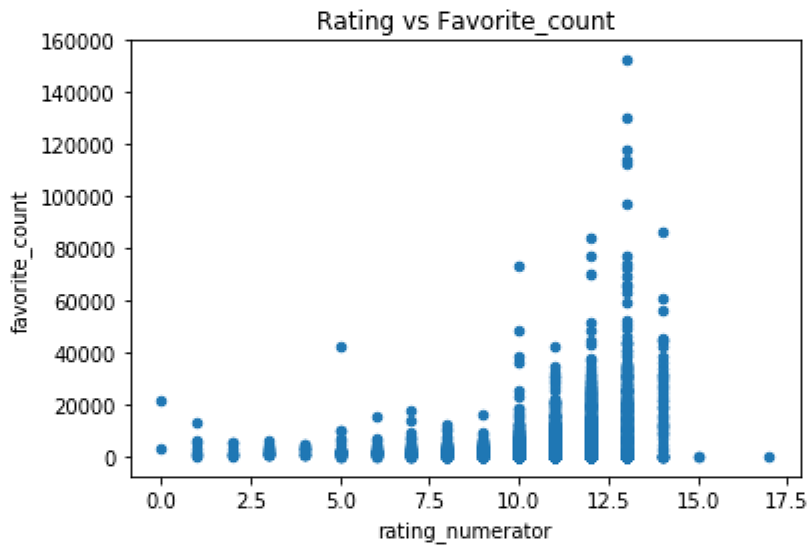
Research Question2:Are ratings of WeRateDogs co related to retweet_count and favorite_count

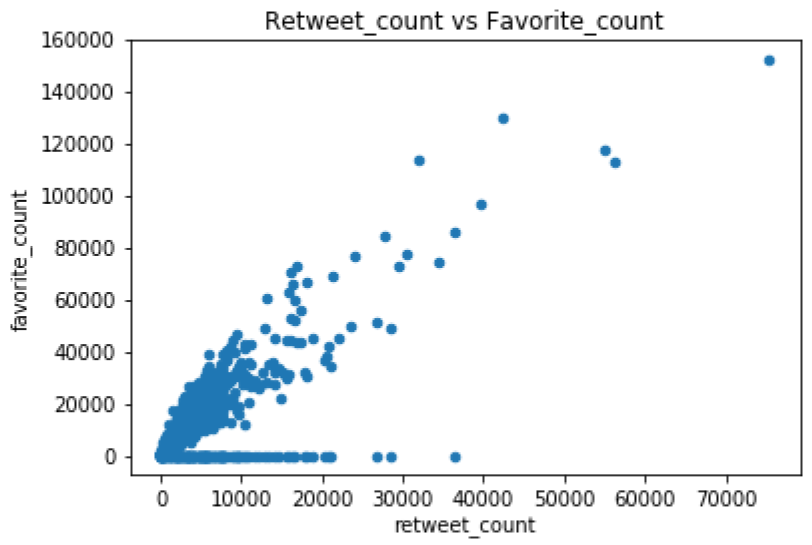
```
In [148]: df_heatmap= pd.DataFrame(df,columns=['rating_numerator','retweet_count','favorite_count'])
          df_heatmap
          ax = sns.heatmap(df_heatmap.corr(),annot=True)
          plt.title('corelation matrix')
          plt.show()
```



In [144]:
```
df.plot(x='rating_numerator',y='retweet_count',kind='scatter',title='Rating vs Retweet_cour
df.plot(x='rating_numerator',y='favorite_count',kind='scatter',title='Rating vs Favorite_cc
df.plot(x='retweet_count',y='favorite_count',kind='scatter',title='Retweet_count vs Favorit
```







*There is no linear relationship between ratings and retweet_count & ratings and favorite_count.But there is strong linear relationship between favorite counts and retweet counts.

Research Question 3: What are the top 10 most predicted breed of dog?

In [65]:
```
#use melt function to unpivot columns['p1', 'p2', 'p3'] and store the value in column 'prec
df_image_melt1=pd.melt(df_image, id_vars='tweet_id' ,value_vars=['p1', 'p2', 'p3'], var_nar
```

In [66]:
```
df_image_melt1.shape
```

Out[66]:
```
(5253, 3)
```

In [67]:
```
df_image_melt1.head(5)
```

Out[67]:

	tweet_id	variable	predicted_breed
0	666020888022790149	p1	Welsh_springer_spaniel
1	666029285002620928	p1	redbone
2	666033412701032449	p1	German_shepherd
3	666044226329800704	p1	Rhodesian_ridgeback
4	666049248165822465	p1	miniature_pinscher

In [68]:
```
df_image_melt1.drop('variable',axis=1,inplace=True)
```

In [69]:
```
df_image_melt1.head()
```

Out[69]:

	tweet_id	predicted_breed
0	666020888022790149	Welsh_springer_spaniel
1	666029285002620928	redbone
2	666033412701032449	German_shepherd
3	666044226329800704	Rhodesian_ridgeback
4	666049248165822465	miniature_pinscher

In [70]:
```
#use melt function to unpivot columns['p1_dog','p2_dog','p3_dog'] and store the value in co
df_image_melt2=pd.melt(df_image, id_vars='tweet_id', value_vars=['p1_dog','p2_dog','p3_dog
```

In [71]:
```
df_image_melt2.shape
```

Out[71]:
```
(5253, 3)
```

In [72]:
```
df_image_melt2.head()
```

Out[72]:

	tweet_id	variable	pred_dog
0	666020888022790149	p1_dog	True
1	666029285002620928	p1_dog	True
2	666033412701032449	p1_dog	True
3	666044226329800704	p1_dog	True
4	666049248165822465	p1_dog	True

In [73]:
```
df_image_melt2.drop('variable',axis=1,inplace=True)
```

In [74]:
```
df_image_melt2.head()
```

Out[74]:

	tweet_id	pred_dog
0	666020888022790149	True
1	666029285002620928	True
2	666033412701032449	True
3	666044226329800704	True
4	666049248165822465	True

In [75]:
```
# Merge the both unpivoted dataframe and save into df_image_melt3
df_image_melt3=pd.merge(df_image_melt1,df_image_melt2,right_index=True,left_index=True,on=
```

In [76]:
```
df_image_melt3.shape
```

Out[76]:
```
(5253, 3)
```

In [77]:
```
df_image_melt3.head()
```

Out[77]:

	tweet_id	predicted_breed	pred_dog
0	666020888022790149	Welsh_springer_spaniel	True
1	666029285002620928	redbone	True
2	666033412701032449	German_shepherd	True
3	666044226329800704	Rhodesian_ridgeback	True
4	666049248165822465	miniature_pinscher	True

In [78]:
```
# Filter the data where 'pred_dog'==True and store the data into df_breed
df_breed=df_image_melt3.query('pred_dog==True')
```

In [79]:
```
df_breed.shape[0]
```

Out[79]:
```
4584
```

In [80]:
```
# Merge the dataframe df_breed with df
df_with_breed=pd.merge(df,df_breed,right_index=True,left_index=True,on=['tweet_id'])
```

```
In [81]: df_with_breed.shape
```

```
Out[81]: (2031, 9)
```

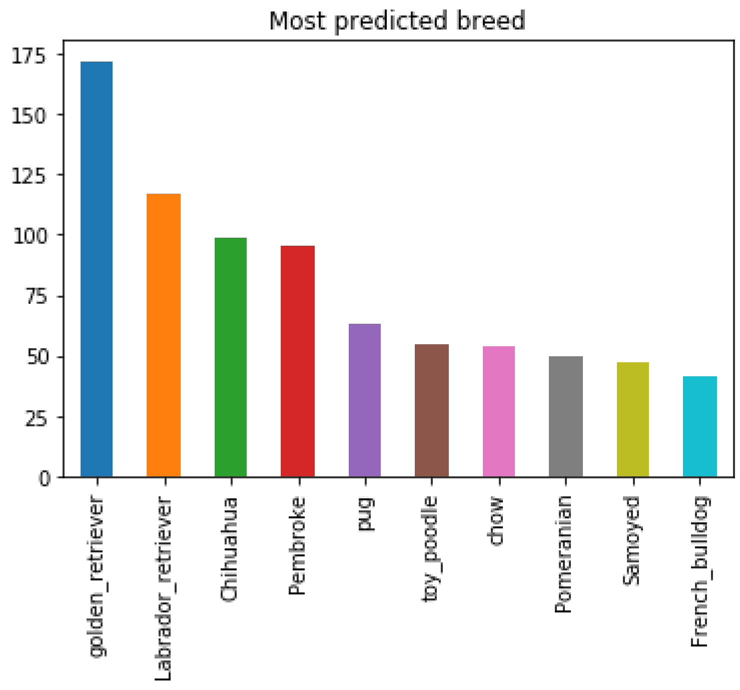
```
In [82]: df_with_breed.retweet_count.describe()
```

```
Out[82]: count     2014.000000
         mean      2607.972691
         std       4366.281145
         min          2.000000
         25%        538.250000
         50%       1246.000000
         75%       3072.750000
         max      75421.000000
         Name: retweet_count, dtype: float64
```

```
In [83]: df_with_breed.predicted_breed.value_counts().iloc[:10]
```

```
Out[83]: golden_retriever      172
         Labrador_retriever    117
         Chihuahua              99
         Pembroke               95
         pug                    63
         toy_poodle             55
         chow                   54
         Pomeranian             50
         Samoyed                47
         French_bulldog         41
         Name: predicted_breed, dtype: int64
```

```
In [103]: df_with_breed.predicted_breed.value_counts().iloc[:10].plot(kind='bar',title='Most predicte
```



As we can see golden_retriever,Labrador_retriever,Chihuahua,Pembroke,pug,toy_poodle,chow,Pomeranian,Samoyed,French_b are the top 10 most predicted breed

Research Question 4: What are the top three breeds of dogs got the highest ratings?

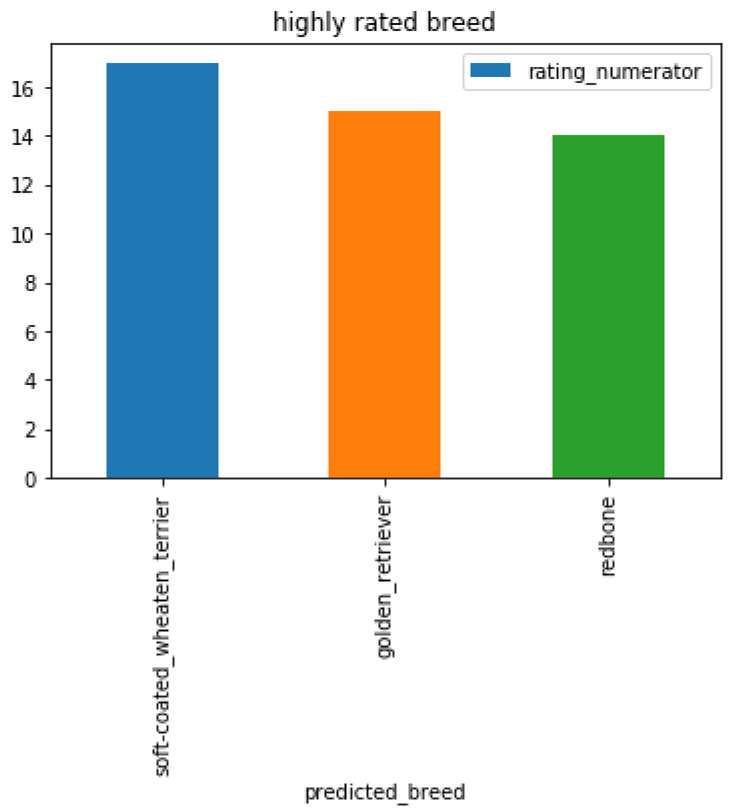
In [85]:
```
df_with_breed.sort_values(by='rating_numerator',ascending=False).loc[:,['predicted_breed',
```

Out[85]:

	predicted_breed	rating_numerator
55	soft-coated_wheaten_terrier	17
283	golden_retriever	15
212	redbone	14

In [147]:
```
df_with_breed.sort_values(by='rating_numerator',ascending=False).loc[:,['predicted_breed',
```

Out[147]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fc67469ff98>
```



As we can see soft-coated_wheaten_terrier,golden_retriever,redbone are the top 3 breed of dogs got highest ratings.

Research Question 5: Whats are the top three breeds of dogs got the highest average retweet counts ?

In [87]:
```
df_breed_retweet=df_with_breed.groupby('predicted_breed').retweet_count.mean()
```

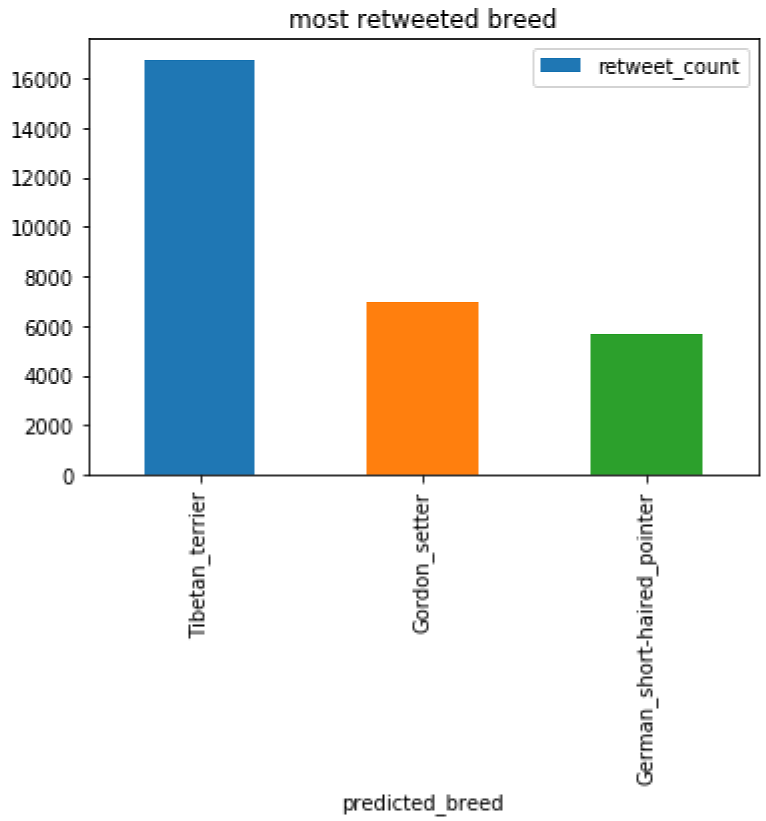
In [88]:
```
df_breed_retweet=df_breed_retweet.reset_index()
```

In [89]:
```
df_breed_retweet.sort_values(by='retweet_count',ascending=False).loc[:,('predicted_breed',
```

Out[89]:

	predicted_breed	retweet_count
66	Tibetan_terrier	16770.4
26	Gordon_setter	6979.0
25	German_short-haired_pointer	5666.9

In [104]:
```
df_breed_retweet.sort_values(by='retweet_count',ascending=False).loc[:,('predicted_breed',
```



As we can see Tibetan_terrier,Gordon_setter,German_short-haired_pointer are the top 3 breed of dogs got highest average retweet_count.

/