Project: Wrangling and Analyze Data

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

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

Data Gathering

1. Directly download the WeRateDogs Twitter archive data (twitter_archive_enhanced.csv)

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

1. Use the Requests library to download the tweet image prediction (image_predictions.tsv)

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


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

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

1. Use the Tweepy library to query additional data via the Twitter API (tweet_json.txt)

In [ ]:
```
auth = tweepy.OAuth2BearerHandler("MY_BEARER_TOKEN")

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

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

start = time.time()

with open(tweets_text, 'w') as file:
    for id_of_tweet in twitter_archive['tweet_id']:
        try:
            tweet = api.get_status(id_of_tweet, tweet_mode='extended')
```

```
            json.dump(tweet._json, file)
            file.write('\n')
            count_true += 1
        except tweepy.TweepyException as e:
            count_error += 1
            id_error.append(id_of_tweet)
            continue

end = timer()

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

This will take about 20 - 30 minutes

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

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

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

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

Assessing Data

General overview of the 3 datasets

In [7]:
```
twitter_archive.head()
```

Out[7]:

	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/iphon
1	892177421306343426	NaN	NaN	2017-08-01 00:17:27 +0000	href="http://twitter.com/download/iphon
2	891815181378084864	NaN	NaN	2017-07-31 00:18:03 +0000	href="http://twitter.com/download/iphon
3	891689557279858688	NaN	NaN	2017-07-30 15:58:51 +0000	href="http://twitter.com/download/iphon
4	891327558926688256	NaN	NaN	2017-07-	

29 16:00:24 +0000 href="http://twitter.com/download/iphon

In [8]:
```
image_predict.head()
```

Out[8]:

	tweet_id	jpg_url	img_num	p1	p1_co
0	666020888022790149	https://pbs.twimg.com/media/CT4udn0WwAA0aMy.jpg	1	Welsh_springer_spaniel	0.4650
1	666029285002620928	https://pbs.twimg.com/media/CT42GRgUYAA5iDo.jpg	1	redbone	0.5068
2	666033412701032449	https://pbs.twimg.com/media/CT4521TWwAEvMyu.jpg	1	German_shepherd	0.5964
3	666044226329800704	https://pbs.twimg.com/media/CT5Dr8HUEAA-lEu.jpg	1	Rhodesian_ridgeback	0.4081
4	666049248165822465	https://pbs.twimg.com/media/CT5IQmsXIAAKY4A.jpg	1	miniature_pinscher	0.5603

In [9]:
```
tweets_df.head()
```

Out[9]:

	id	retweet_count	favorite_count
0	892420643555336193	6979	33728
1	892177421306343426	5280	29254
2	891815181378084864	3466	21987
3	891689557279858688	7198	36823
4	891327558926688256	7723	35207

Accessing twitter_archive dataframe

In [10]:
```
twitter_archive.columns
```

Out[10]:
```
Index(['tweet_id', 'in_reply_to_status_id', 'in_reply_to_user_id', 'timestamp',
       'source', 'text', 'retweeted_status_id', 'retweeted_status_user_id',
       'retweeted_status_timestamp', 'expanded_urls', 'rating_numerator',
       'rating_denominator', 'name', 'doggo', 'floofer', 'pupper', 'puppo'],
      dtype='object')
```

In [11]:
```
twitter_archive.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
```

In [12]:
```
twitter_archive.describe()
```

Out[12]:

	tweet_id	in_reply_to_status_id	in_reply_to_user_id	retweeted_status_id	retweeted_status_user_id	rating
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 [13]:
```
twitter_archive.rating_numerator.value_counts()
```

Out[13]:
```
12      558
11      464
10      461
13      351
9       158
8       102
7        55
14       54
5        37
6        32
3        19
4        17
2         9
1         9
75        2
15        2
420       2
0         2
80        1
144       1
17        1
26        1
20        1
121       1
143       1
44        1
60        1
45        1
50        1
99        1
204       1
1776      1
165       1
666       1
27        1
182       1
```

```
24         1
960        1
84         1
88         1
Name: rating_numerator, dtype: int64
```

In [14]:
```
twitter_archive.rating_denominator.value_counts()
```

Out[14]:
```
10     2333
11        3
50        3
20        2
80        2
70        1
7         1
15        1
150       1
170       1
0         1
90        1
40        1
130       1
110       1
16        1
120       1
2         1
Name: rating_denominator, dtype: int64
```

In [15]:
```
# Check retweets
twitter_archive[twitter_archive.retweeted_status_id.isna() != True]
```

Out[15]:

	tweet_id	in_reply_to_status_id	in_reply_to_user_id	timestamp	s
19	888202515573088257	NaN	NaN	2017-07-21 01:02:36 +0000	href="http://twitter.com/download/ip
32	886054160059072513	NaN	NaN	2017-07-15 02:45:48 +0000	href="http://twitter.com/download/ip
36	885311592912609280	NaN	NaN	2017-07-13 01:35:06 +0000	href="http://twitter.com/download/ip
68	879130579576475649	NaN	NaN	2017-06-26 00:13:58 +0000	href="http://twitter.com/download/ip
73	878404777348136964	NaN	NaN	2017-06-24 00:09:53 +0000	href="http://twitter.com/download/ip
...	...	...	...	...	
1023	746521445350707200	NaN	NaN	2016-06-25 01:52:36 +0000	href="http://twitter.com/download/ip
1043	743835915802583040	NaN	NaN	2016-06-17 16:01:16 +0000	href="http://twitter.com/download/ip
1242	711998809858043904	NaN	NaN	2016-03-21 19:31:59 +0000	href="http://twitter.com/download/ip

2259	667550904950915073	NaN	NaN	2015-11-20 03:51:52 +0000	<a href="http://twitter rel="nofollow"
2260	667550882905632768	NaN	NaN	2015-11-20 03:51:47 +0000	<a href="http://twitter rel="nofollow"

181 rows × 17 columns

In [16]:
```
# Names are usually in title case i.e first letter capitalised except compound names
twitter_archive[
    (~(twitter_archive.name.str.istitle() == True)) | (twitter_archive.name.str.islower(
```

Out[16]:
```
array(['such', 'a', 'quite', 'not', 'one', 'incredibly', 'BeBe', 'mad',
       'an', 'very', 'just', 'DonDon', 'my', 'his', 'actually', 'getting',
       'this', 'unacceptable', 'all', 'old', 'infuriating', 'CeCe', 'the',
       'by', 'officially', 'life', 'light', 'space', 'JD', 'DayZ'],
      dtype=object)
```

In [17]:
```
# To catch invalid names which which will most probably be less than 4 and do not start
twitter_archive[twitter_archive.name.str.len() < 4].name.unique()
```

Out[17]:
```
array(['Jax', 'Ted', 'Jim', 'Gus', 'Rey', 'a', 'Aja', 'Jed', 'Leo', 'Ken',
       'Max', 'Ava', 'Eli', 'Ash', 'not', 'Mia', 'one', 'Ike', 'Mo', 'Bo',
       'Tom', 'Alf', 'Sky', 'Tyr', 'Moe', 'Sam', 'Ito', 'Doc', 'mad',
       'Jay', 'Mya', 'an', 'O', 'Al', 'Lou', 'my', 'Eve', 'Dex', 'Ace',
       'Zoe', 'Blu', 'his', 'all', 'Sid', 'old', 'Ole', 'Bob', 'the',
       'Obi', 'by', 'Evy', 'Tug', 'Jeb', 'Dot', 'Mac', 'Ed', 'Taz', 'Cal',
       'JD', 'Pip', 'Amy', 'Gin', 'Edd', 'Ben', 'Dug', 'Jo', 'Ron', 'Stu'],
      dtype=object)
```

In [18]:
```
# Display full texts of tweets with such names
with pd.option_context('display.max_colwidth', None):
    display(twitter_archive[twitter_archive.name.isin(
        ['not','one','mad','an','my','his','old','all','the','by']) == True])
```

	tweet_id	in_reply_to_status_id	in_reply_to_user_id	timestamp	
335	832645525019123713	NaN	NaN	2017-02-17 17:38:57 +0000	<a href="http://twitter.com" rel="no
369	828650029636317184	NaN	NaN	2017-02-06 17:02:17 +0000	<a href="http://twitter.com/dov rel="nofollow">Twitter
682	788552643979468800	NaN	NaN	2016-10-19 01:29:35 +0000	<a href="http://twitter.com/dov rel="nofollow">Twitter
759	778396591732486144	NaN	NaN	2016-09-21 00:53:04 +0000	<a href="http://twitter.com/dov rel="nofollow">Twitter
852	765395769549590528	NaN	NaN	2016-08-16 03:52:26 +0000	<a href="http://twitter.com/dov rel="nofollow">Twitter

924	755206590534418437	NaN	NaN	2016-07-19 01:04:16 +0000	<a href="http://twitter.com/dov rel="nofollow">Twitter
988	748977405889503236	NaN	NaN	2016-07-01 20:31:43 +0000	<a href="http://twitter.com/dov rel="nofollow">Twitter
992	748692773788876800	NaN	NaN	2016-07-01 01:40:41 +0000	<a href="http://twitter.com/dov rel="nofollow">Twitter
993	748575535303884801	NaN	NaN	2016-06-30 17:54:50 +0000	<a href="http://twitter.com/dov rel="nofollow">Twitter
1025	746369468511756288	NaN	NaN	2016-06-24 15:48:42 +0000	<a href="http://twitter.com/dov rel="nofollow">Twitter
1095	736392552031657984	NaN	NaN	2016-05-28 03:04:00 +0000	<a href="http://vine.co" rel="nofollov
1138	728035342121635841	NaN	NaN	2016-05-05 01:35:26 +0000	<a href="http://twitter.com/dov rel="nofollow">Twitter
1206	715758151270801409	NaN	NaN	2016-04-01 04:30:16 +0000	<a href="http://vine.co" rel="nofollov
1362	703041949650034688	NaN	NaN	2016-02-26 02:20:37 +0000	<a href="http://twitter.com/dov rel="nofollow">Twitter
1527	690360449368465409	NaN	NaN	2016-01-22 02:28:52 +0000	<a href="http://twitter.com/dov rel="nofollow">Twitter

1603	685943807276412928	NaN	NaN	2016-01-09 21:58:42 +0000	<a href="http://twitter.com/dov rel="nofollow">Twitter
1724	680085611152338944	NaN	NaN	2015-12-24 18:00:19 +0000	href="https://about.twitter.com/produ rel="nofollow">T
1797	677269281705472000	NaN	NaN	2015-12-16 23:29:14 +0000	<a href="http://twitter.com/dov rel="nofollow">Twitter
1815	676613908052996102	NaN	NaN	2015-12-15 04:05:01 +0000	<a href="http://twitter.com/dov rel="nofollow">Twitter
1936	673956914389192708	NaN	NaN	2015-12-07 20:07:04 +0000	<a href="http://twitter.com/dov rel="nofollow">Twitter
2037	671561002136281088	NaN	NaN	2015-12-01 05:26:34 +0000	<a href="http://twitter.com/dov rel="nofollow">Twitter
2204	668636665813057536	NaN	NaN	2015-11-23 03:46:18 +0000	<a href="http://twitter.com/dov rel="nofollow">Twitter
2212	668587383441514497	NaN	NaN	2015-11-23 00:30:28 +0000	<a href="http://vine.co" rel="nofollov
2333	666337882303524864	NaN	NaN	2015-11-16 19:31:45 +0000	<a href="http://twitter.com/dov rel="nofollow">Twitter
2335	666287406224695296	NaN	NaN	2015-11-16 16:11:11 +0000	<a href="http://twitter.com/dov rel="nofollow">Twitter
2345	666063827256086533	NaN	NaN	2015-11-16 01:22:45 +0000	<a href="http://twitter.com/dov rel="nofollow">Twitter

2346	666058600524156928	NaN	NaN	2015-11-16 01:01:59 +0000	<a href="http://twitter.com/dov rel="nofollow">Twitter
2349	666051853826850816	NaN	NaN	2015-11-16 00:35:11 +0000	<a href="http://twitter.com/dov rel="nofollow">Twitter

In [19]:
```
# Display tweet of dogs named O
twitter_archive[twitter_archive.name == 'O']['text']
```

Out[19]:
```
775    This is O'Malley. That is how he sleeps. Doesn...
Name: text, dtype: object
```

In [20]:
```
# Display tweet of dogs named a
with pd.option_context('display.max_colwidth', None):
    display(twitter_archive[twitter_archive.name == 'a'])
```

	tweet_id	in_reply_to_status_id	in_reply_to_user_id	timestamp	s
56	881536004380872706	NaN	NaN	2017-07-02 15:32:16 +0000	href="http://twitter.com/download/ip rel="nofollow">Twitter for iPhon
649	792913359805018113	NaN	NaN	2016-10-31 02:17:31 +0000	href="http://twitter.com/download/ip rel="nofollow">Twitter for iPhon
801	772581559778025472	NaN	NaN	2016-09-04 23:46:12 +0000	href="http://twitter.com/download/ip rel="nofollow">Twitter for iPhon
1002	747885874273214464	NaN	NaN	2016-06-28 20:14:22 +0000	href="http://twitter.com/download/ip rel="nofollow">Twitter for iPhon
1004	747816857231626240	NaN	NaN	2016-06-28 15:40:07 +0000	href="http://twitter.com/download/ip rel="nofollow">Twitter for iPhon
1017	746872823977771008	NaN	NaN	2016-06-26 01:08:52	href="http://twitter.com/download/ip

				+0000	rel="nofollow">Twitter for iPhon
1049	743222593470234624	NaN	NaN	2016-06-15 23:24:09 +0000	href="http://twitter.com/download/ip rel="nofollow">Twitter for iPhon
1193	717537687239008257	NaN	NaN	2016-04-06 02:21:30 +0000	href="http://twitter.com/download/ip rel="nofollow">Twitter for iPhon
1207	715733265223708672	NaN	NaN	2016-04-01 02:51:22 +0000	href="http://twitter.com/download/ip rel="nofollow">Twitter for iPhon
1340	704859558691414016	NaN	NaN	2016-03-02 02:43:09 +0000	href="http://twitter.com/download/ip rel="nofollow">Twitter for iPhon
1351	704054845121142784	NaN	NaN	2016-02-28 21:25:30 +0000	href="http://twitter.com/download/ip rel="nofollow">Twitter for iPhon
1361	703079050210877440	NaN	NaN	2016-02-26 04:48:02 +0000	href="http://twitter.com/download/ip rel="nofollow">Twitter for iPhon
1368	702539513671897089	NaN	NaN	2016-02-24 17:04:07 +0000	href="http://twitter.com/download/ip rel="nofollow">Twitter for iPhon
1382	700864154249383937	NaN	NaN	2016-02-20 02:06:50 +0000	href="http://twitter.com/download/ip rel="nofollow">Twitter for iPhon
1499	692187005137076224	NaN	NaN	2016-01-27 03:26:56 +0000	href="http://twitter.com/download/ip rel="nofollow">Twitter for iPhon
1737	679530280114372609	NaN	NaN	2015-12-23 05:13:38 +0000	href="http://twitter.com/download/ip rel="nofollow">Twitter for iPhon

1785	677644091929329666	NaN	NaN	2015-12-18 00:18:36 +0000	href="http://twitter.com/download/ip rel="nofollow">Twitter for iPhon
1853	675706639471788032	NaN	NaN	2015-12-12 15:59:51 +0000	href="http://twitter.com/download/ip rel="nofollow">Twitter for iPhon
1854	675534494439489536	NaN	NaN	2015-12-12 04:35:48 +0000	href="http://twitter.com/download/ip rel="nofollow">Twitter for iPhon
1877	675109292475830276	NaN	NaN	2015-12-11 00:26:12 +0000	href="http://twitter.com/download/ip rel="nofollow">Twitter for iPhon
1878	675047298674663426	NaN	NaN	2015-12-10 20:19:52 +0000	href="http://twitter.com/download/ip rel="nofollow">Twitter for iPhon
1923	674082852460433408	NaN	NaN	2015-12-08 04:27:30 +0000	href="http://twitter.com/download/ip rel="nofollow">Twitter for iPhon
1941	673715861853720576	NaN	NaN	2015-12-07 04:09:13 +0000	href="http://twitter.com/download/ip rel="nofollow">Twitter for iPhon
1955	673636718965334016	NaN	NaN	2015-12-06 22:54:44 +0000	href="http://twitter.com/download/ip rel="nofollow">Twitter for iPhon
1994	672604026190569472	NaN	NaN	2015-12-04 02:31:10 +0000	href="http://twitter.com/download/ip rel="nofollow">Twitter for iPhon
2034	671743150407421952	NaN	NaN	2015-12-01 17:30:22 +0000	href="http://twitter.com/download/ip rel="nofollow">Twitter for iPhon
2066	671147085991960577	NaN	NaN	2015-11-30 02:01:49	href="http://twitter.com/download/ip

				+0000	rel="nofollow">Twitter for iPhon
2116	670427002554466305	NaN	NaN	2015-11-28 02:20:27 +0000	href="http://twitter.com/download/ip rel="nofollow">Twitter for iPhon
2125	670361874861563904	NaN	NaN	2015-11-27 22:01:40 +0000	href="http://twitter.com/download/ip rel="nofollow">Twitter for iPhon
2128	670303360680108032	NaN	NaN	2015-11-27 18:09:09 +0000	href="http://twitter.com/download/ip rel="nofollow">Twitter for iPhon
2146	669923323644657664	NaN	NaN	2015-11-26 16:59:01 +0000	href="http://twitter.com/download/ip rel="nofollow">Twitter for iPhon
2153	669661792646373376	NaN	NaN	2015-11-25 23:39:47 +0000	href="http://twitter.com/download/ip rel="nofollow">Twitter for iPhon
2161	669564461267722241	NaN	NaN	2015-11-25 17:13:02 +0000	href="http://twitter.com/download/ip rel="nofollow">Twitter for iPhon
2191	668955713004314625	NaN	NaN	2015-11-24 00:54:05 +0000	href="http://twitter.com/download/ip rel="nofollow">Twitter for iPhon
2198	668815180734689280	NaN	NaN	2015-11-23 15:35:39 +0000	href="http://twitter.com/download/ip rel="nofollow">Twitter for iPhon
2211	668614819948453888	NaN	NaN	2015-11-23 02:19:29 +0000	href="http://twitter.com/download/ip rel="nofollow">Twitter for iPhon

2218	668507509523615744	NaN	NaN	2015-11-22 19:13:05 +0000	href="http://twitter.com/download/ip rel="nofollow">Twitter for iPhon
2222	668466899341221888	NaN	NaN	2015-11-22 16:31:42 +0000	href="http://twitter.com/download/ip rel="nofollow">Twitter for iPhon
2235	668171859951755264	NaN	NaN	2015-11-21 20:59:20 +0000	href="http://twitter.com/download/ip rel="nofollow">Twitter for iPhon
2249	667861340749471744	NaN	NaN	2015-11-21 00:25:26 +0000	href="http://twitter.com/download/ip rel="nofollow">Twitter for iPhon
2255	667773195014021121	NaN	NaN	2015-11-20 18:35:10 +0000	<a href="http://twitter rel="nofollow">Twitter Web Clien
2264	667538891197542400	NaN	NaN	2015-11-20 03:04:08 +0000	<a href="http://twitter rel="nofollow">Twitter Web Clien
2273	667470559035432960	NaN	NaN	2015-11-19 22:32:36 +0000	<a href="http://twitter rel="nofollow">Twitter Web Clien
2287	667177989038297088	NaN	NaN	2015-11-19 03:10:02 +0000	href="http://twitter.com/download/ip rel="nofollow">Twitter for iPhon
2304	666983947667116034	NaN	NaN	2015-11-18 14:18:59 +0000	href="http://twitter.com/download/ip rel="nofollow">Twitter for iPhon
2311	666781792255496192	NaN	NaN	2015-11-18 00:55:42	href="http://twitter.com/download/ip

				+0000	rel="nofollow">Twitter for iPhon
2314	666701168228331520	NaN	NaN	2015-11-17 19:35:19 +0000	href="http://twitter.com/download/ip rel="nofollow">Twitter for iPhon
2327	666407126856765440	NaN	NaN	2015-11-17 00:06:54 +0000	href="http://twitter.com/download/ip rel="nofollow">Twitter for iPhon
2334	666293911632134144	NaN	NaN	2015-11-16 16:37:02 +0000	href="http://twitter.com/download/ip rel="nofollow">Twitter for iPhon
2347	666057090499244032	NaN	NaN	2015-11-16 00:55:59 +0000	href="http://twitter.com/download/ip rel="nofollow">Twitter for iPhon
2348	666055525042405380	NaN	NaN	2015-11-16 00:49:46 +0000	href="http://twitter.com/download/ip rel="nofollow">Twitter for iPhon
2350	666050758794694657	NaN	NaN	2015-11-16 00:30:50 +0000	href="http://twitter.com/download/ip rel="nofollow">Twitter for iPhon
2352	666044226329800704	NaN	NaN	2015-11-16 00:04:52 +0000	href="http://twitter.com/download/ip rel="nofollow">Twitter for iPhon
2353	666033412701032449	NaN	NaN	2015-11-15 23:21:54 +0000	href="http://twitter.com/download/ip rel="nofollow">Twitter for iPhon
2354	666029285002620928	NaN	NaN	2015-11-15 23:05:30 +0000	href="http://twitter.com/download/ip rel="nofollow">Twitter for iPhon

Some names that have 'a' as their values sill have the dog's name in the text. They are mostly in the format "This is a *noun that isn't a dog name*."

In [21]:
```
twitter_archive.name.value_counts().head(10)
```

Out[21]:
```
None       745
a           55
Charlie     12
Cooper      11
Lucy        11
Oliver      11
Tucker      10
Penny       10
Lola        10
Winston      9
Name: name, dtype: int64
```

In [22]:
```
# Check duplicated Urls
twitter_archive.expanded_urls.duplicated().sum()
```

Out[22]:
```
137
```

In [23]:
```
# Check duplicated tweets
twitter_archive.tweet_id.duplicated().sum()
```

Out[23]:
```
0
```

In [24]:
```
# Check for null values
twitter_archive.isna().sum()
```

Out[24]:
```
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 [25]:
```
# tweets with no enhanced URL i.e no media file attached
twitter_archive[twitter_archive.expanded_urls.isna()]
```

Out[25]:

	tweet_id	in_reply_to_status_id	in_reply_to_user_id	timestamp	s
30	886267009285017600	8.862664e+17	2.281182e+09	2017-07-15 16:51:35 +0000	href="http://twitter.com/download/ip
55	881633300179243008	8.816070e+17	4.738443e+07	2017-07-02 21:58:53 +0000	href="http://twitter.com/download/ip
64	879674319642796034	8.795538e+17	3.105441e+09	2017-06-27 12:14:36	href="http://twitter.com/download/ip

				+0000	
113	870726314365509632	8.707262e+17	1.648776e+07	2017-06-02 19:38:25 +0000	href="http://twitter.com/download/ip
148	863427515083354112	8.634256e+17	7.759620e+07	2017-05-13 16:15:35 +0000	href="http://twitter.com/download/ip
179	857214891891077121	8.571567e+17	1.806710e+08	2017-04-26 12:48:51 +0000	href="http://twitter.com/download/ip
185	856330835276025856	NaN	NaN	2017-04-24 02:15:55 +0000	href="http://twitter.com/download/ip
186	856288084350160898	8.562860e+17	2.792810e+08	2017-04-23 23:26:03 +0000	href="http://twitter.com/download/ip
188	855862651834028034	8.558616e+17	1.943518e+08	2017-04-22 19:15:32 +0000	href="http://twitter.com/download/ip
189	855860136149123072	8.558585e+17	1.361572e+07	2017-04-22 19:05:32 +0000	href="http://twitter.com/download/ip
218	850333567704068097	8.503288e+17	2.195506e+07	2017-04-07 13:04:55 +0000	href="http://twitter.com/download/ip
228	848213670039564288	8.482121e+17	4.196984e+09	2017-04-01 16:41:12 +0000	href="http://twitter.com/download/ip
234	847617282490613760	8.476062e+17	4.196984e+09	2017-03-31 01:11:22 +0000	href="http://twitter.com/download/ip
274	840698636975636481	8.406983e+17	8.405479e+17	2017-03-11 22:59:09 +0000	href="http://twitter.com/download/ip
290	838150277551247360	8.381455e+17	2.195506e+07	2017-03-04 22:12:52 +0000	href="http://twitter.com/download/ip
291	838085839343206401	8.380855e+17	2.894131e+09	2017-03-04 17:56:49 +0000	href="http://twitter.com/download/ip
313	835246439529840640	8.352460e+17	2.625958e+07	2017-02-24 21:54:03 +0000	href="http://twitter.com/download/ip
342	832088576586297345	8.320875e+17	3.058208e+07	2017-02-16 04:45:50 +0000	href="http://twitter.com/download/ip
346	831926988323639298	8.319030e+17	2.068372e+07	2017-02-15 18:03:45 +0000	href="http://twitter.com/download/ip
375	828361771580813312	NaN	NaN	2017-02-05 21:56:51 +0000	<a href="http://twitter rel="nofollow"

387	826598799820865537	8.265984e+17	4.196984e+09	2017-02-01 01:11:25 +0000	href="http://twitter.com/download/ip
409	823333489516937216	8.233264e+17	1.582854e+09	2017-01-23 00:56:15 +0000	href="http://twitter.com/download/ip
427	821153421864615936	8.211526e+17	1.132119e+08	2017-01-17 00:33:26 +0000	href="http://twitter.com/download/ip
498	813130366689148928	8.131273e+17	4.196984e+09	2016-12-25 21:12:41 +0000	href="http://twitter.com/download/ip
513	811647686436880384	8.116272e+17	4.196984e+09	2016-12-21 19:01:02 +0000	href="http://twitter.com/download/ip
570	801854953262350336	8.018543e+17	1.185634e+07	2016-11-24 18:28:13 +0000	href="http://twitter.com/download/ip
576	800859414831898624	8.008580e+17	2.918590e+08	2016-11-22 00:32:18 +0000	href="http://twitter.com/download/ip
611	797165961484890113	7.971238e+17	2.916630e+07	2016-11-11 19:55:50 +0000	href="http://twitter.com/download/ip
701	786051337297522688	7.727430e+17	7.305050e+17	2016-10-12 03:50:17 +0000	href="http://twitter.com/download/ip
707	785515384317313025	NaN	NaN	2016-10-10 16:20:36 +0000	href="http://twitter.com/download/ip
843	766714921925144576	7.667118e+17	4.196984e+09	2016-08-19 19:14:16 +0000	href="http://twitter.com/download/ip
857	763956972077010945	7.638652e+17	1.584641e+07	2016-08-12 04:35:10 +0000	href="http://twitter.com/download/ip
967	750381685133418496	7.501805e+17	4.717297e+09	2016-07-05 17:31:49 +0000	href="http://twitter.com/download/ip
1005	747651430853525504	7.476487e+17	4.196984e+09	2016-06-28 04:42:46 +0000	href="http://twitter.com/download/ip
1080	738891149612572673	7.384119e+17	3.589728e+08	2016-06-04 00:32:32 +0000	href="http://twitter.com/download/ip
1295	707983188426153984	7.079801e+17	2.319108e+09	2016-03-10 17:35:20 +0000	href="http://twitter.com/download/ip
1345	704491224099647488	7.044857e+17	2.878549e+07	2016-03-01 02:19:31 +0000	href="http://twitter.com/download/ip
1445	696518437233913856	NaN	NaN	2016-02-	

				08 02:18:30 +0000	href="http://twitter.com/download/ip
1446	696490539101908992	6.964887e+17	4.196984e+09	2016-02-08 00:27:39 +0000	href="http://twitter.com/download/ip
1474	693644216740769793	6.936422e+17	4.196984e+09	2016-01-31 03:57:23 +0000	href="http://twitter.com/download/ip
1479	693582294167244802	6.935722e+17	1.198989e+09	2016-01-30 23:51:19 +0000	href="http://twitter.com/download/ip
1497	692423280028966913	6.924173e+17	4.196984e+09	2016-01-27 19:05:49 +0000	href="http://twitter.com/download/ip
1523	690607260360429569	6.903413e+17	4.670367e+08	2016-01-22 18:49:36 +0000	href="http://twitter.com/download/ip
1598	686035780142297088	6.860340e+17	4.196984e+09	2016-01-10 04:04:10 +0000	href="http://twitter.com/download/ip
1605	685681090388975616	6.855479e+17	4.196984e+09	2016-01-09 04:34:45 +0000	href="http://twitter.com/download/ip
1618	684969860808454144	6.849598e+17	4.196984e+09	2016-01-07 05:28:35 +0000	href="http://twitter.com/download/ip
1663	682808988178739200	6.827884e+17	4.196984e+09	2016-01-01 06:22:03 +0000	href="http://twitter.com/download/ip
1689	681340665377193984	6.813394e+17	4.196984e+09	2015-12-28 05:07:27 +0000	href="http://twitter.com/download/ip
1774	678023323247357953	6.780211e+17	4.196984e+09	2015-12-19 01:25:31 +0000	href="http://twitter.com/download/ip
1819	676590572941893632	6.765883e+17	4.196984e+09	2015-12-15 02:32:17 +0000	href="http://twitter.com/download/ip
1844	675849018447167488	6.758457e+17	4.196984e+09	2015-12-13 01:25:37 +0000	href="http://twitter.com/download/ip
1895	674742531037511680	6.747400e+17	4.196984e+09	2015-12-10 00:08:50 +0000	href="http://twitter.com/download/ip
1905	674606911342424069	6.744689e+17	4.196984e+09	2015-12-09 15:09:55 +0000	href="http://twitter.com/download/ip
1914	674330906434379776	6.658147e+17	1.637468e+07	2015-12-08 20:53:11 +0000	href="http://twitter.com/download/ip
1940	673716320723169284	6.737159e+17	4.196984e+09	2015-12-07 04:11:02 +0000	href="http://twitter.com/download/ip

2038	671550332464455680	6.715449e+17	4.196984e+09	2015-12-01 04:44:10 +0000	href="http://twitter.com/download/ip
2149	669684865554620416	6.693544e+17	4.196984e+09	2015-11-26 01:11:28 +0000	href="http://twitter.com/download/ip
2189	668967877119254528	6.689207e+17	2.143566e+07	2015-11-24 01:42:25 +0000	href="http://twitter.com/download/ip
2298	667070482143944705	6.670655e+17	4.196984e+09	2015-11-18 20:02:51 +0000	href="http://twitter.com/download/ip

In [26]:
```
twitter_archive.in_reply_to_status_id[twitter_archive.expanded_urls.isna()].shape
```

Out[26]:
```
(59,)
```

In [27]:
```
# Check unique tweet sources
twitter_archive.source.unique()
```

Out[27]:
```
array(['<a href="http://twitter.com/download/iphone" rel="nofollow">Twitter for iPhone</
a>',
       '<a href="http://twitter.com" rel="nofollow">Twitter Web Client</a>',
       '<a href="http://vine.co" rel="nofollow">Vine - Make a Scene</a>',
       '<a href="https://about.twitter.com/products/tweetdeck" rel="nofollow">TweetDeck
</a>'],
      dtype=object)
```

Accessing image_predict dataframe

In [28]:
```
image_predict.head(5)
```

Out[28]:

	tweet_id	jpg_url	img_num	p1	p1_co
0	666020888022790149	https://pbs.twimg.com/media/CT4udn0WwAA0aMy.jpg	1	Welsh_springer_spaniel	0.4650
1	666029285002620928	https://pbs.twimg.com/media/CT42GRgUYAA5iDo.jpg	1	redbone	0.5068
2	666033412701032449	https://pbs.twimg.com/media/CT4521TWwAEvMyu.jpg	1	German_shepherd	0.5964
3	666044226329800704	https://pbs.twimg.com/media/CT5Dr8HUEAA-lEu.jpg	1	Rhodesian_ridgeback	0.4081
4	666049248165822465	https://pbs.twimg.com/media/CT5IQmsXIAAKY4A.jpg	1	miniature_pinscher	0.5603

In [29]:
```
image_predict.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
```

In [30]:
```
image_predict.img_num.value_counts()
```

Out[30]:
```
1    1780
2     198
3      66
4      31
Name: img_num, dtype: int64
```

In [31]:
```
image_predict.describe()
```

Out[31]:

	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
75%	7.932034e+17	1.000000	0.843855	1.955655e-01	9.180755e-02
max	8.924206e+17	4.000000	1.000000	4.880140e-01	2.734190e-01

In [32]:
```
image_predict.tweet_id.duplicated().sum()
```

Out[32]:
```
0
```

Accessing tweets_df dataframe

In [33]:
```
tweets_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2327 entries, 0 to 2326
Data columns (total 3 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   id              2327 non-null   int64
 1   retweet_count   2327 non-null   int64
 2   favorite_count  2327 non-null   int64
dtypes: int64(3)
memory usage: 54.7 KB
```

In [34]:
```
tweets_df.describe()
```

Out[34]:

	id	retweet_count	favorite_count
count	2.327000e+03	2327.000000	2327.000000
mean	7.417930e+17	2461.259132	7033.580576
std	6.820795e+16	4168.498178	10929.707459
min	6.660209e+17	1.000000	0.000000

25%	6.781394e+17	492.500000	1221.000000
50%	7.178418e+17	1147.000000	3041.000000
75%	7.986547e+17	2847.000000	8574.500000
max	8.924206e+17	70427.000000	144401.000000

In [35]:
```
tweets_df.isna().sum()
```

Out[35]:
```
id                0
retweet_count     0
favorite_count    0
dtype: int64
```

Quality issues

twitter_archive dataframe

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

image_predict dataframe

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

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

Tidiness issues

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

Cleaning Data

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

Issue #1:

Erroneous datatype for timestamp, in_reply_to_status_id, in_reply_to_user_id columns and also that of tweet_id.

Define:

Use pythons .astype() function to change the datatypes

Code

In [37]:
```
twitter_archive_clean.timestamp = pd.to_datetime(twitter_archive_clean.timestamp)
twitter_archive_clean.in_reply_to_status_id = twitter_archive_clean.in_reply_to_status_i
twitter_archive_clean.in_reply_to_user_id = twitter_archive_clean.in_reply_to_user_id.as
twitter_archive_clean.tweet_id = twitter_archive_clean.tweet_id.astype('str')
```

Test

In [38]:
```
twitter_archive_clean.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   object
 1   in_reply_to_status_id       2356 non-null   object
 2   in_reply_to_user_id         2356 non-null   object
 3   timestamp                   2356 non-null   datetime64[ns, UTC]
 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: datetime64[ns, UTC](1), float64(2), int64(2), object(12)
memory usage: 313.0+ KB
```

Issue #2:

tweets_archive: 181 Retweets (none original tweets) with unneccesary reply features columns present

Define

- Drop rows where retweeted_status_id is not null.
- Drop retweeted_status_id, retweeted_status_user_id, retweeted_status_timestamp, in_reply_to_user_id, in_reply_to_status_id columns

Code

In [39]:
```
# Drop non-null enteries for retweet status
twitter_archive_clean.drop(
    twitter_archive_clean[twitter_archive_clean.retweeted_status_id.isna() != True].inde

# Drop the unnecessary columns
twitter_archive_clean.drop(
    ['retweeted_status_id', 'retweeted_status_user_id', 'retweeted_status_timestamp', 'i
     'in_reply_to_status_id'],axis = 1, inplace=True)
```

Test

In [40]:
```
twitter_archive_clean.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2175 entries, 0 to 2355
Data columns (total 12 columns):
 #   Column              Non-Null Count  Dtype              
---  ------              --------------  -----              
 0   tweet_id            2175 non-null   object             
 1   timestamp           2175 non-null   datetime64[ns, UTC]
 2   source              2175 non-null   object             
 3   text                2175 non-null   object             
 4   expanded_urls       2117 non-null   object             
 5   rating_numerator    2175 non-null   int64              
 6   rating_denominator  2175 non-null   int64              
 7   name                2175 non-null   object             
 8   doggo               2175 non-null   object             
 9   floofer             2175 non-null   object             
 10  pupper              2175 non-null   object             
 11  puppo               2175 non-null   object             
dtypes: datetime64[ns, UTC](1), int64(2), object(9)
memory usage: 220.9+ KB
```

Issue #3:

twitter_archive: expanded_urls has duplicate values.

Define

Drop duplicate URLs in the dataframe

Code

In [41]:
```
twitter_archive_clean.drop_duplicates('expanded_urls', inplace=True)
```

Test

In [42]:
```
twitter_archive_clean['expanded_urls'].duplicated().sum()
```

Out[42]:
```
0
```

Issue #4:

Some tweets do not have images attached; no value for expanded_urls.

Define

Drop null values with expanded values as subset

Code

In [43]:
```
twitter_archive_clean.dropna(subset = 'expanded_urls', inplace=True)
```

Test

In [44]:
```
twitter_archive_clean.expanded_urls.isna().any()
```

Out[44]:
```
False
```

Issue #5:

Invalid Dog names like 'a', 'none', 'by', 'very' and so on.

Define

Replace invalid Dog names with Nan.

Code

In [45]:
```
twitter_archive_clean.name.replace(
    to_replace=['not','one','mad','an','my','his','old','all','the','by', 'None', 'a'],
```

Test

In [46]:
```
twitter_archive_clean.name.value_counts()
```

Out[46]:
```
Lucy          11
Charlie       11
Cooper        10
Oliver        10
Tucker         9
              ..
Bonaparte      1
Wishes         1
Rose           1
Theo           1
Christoper     1
Name: name, Length: 944, dtype: int64
```

Issue #6:

Twitter_archive: Incomplete dog name; 'O'.

Define

Manually in put the correct name; **O'Malley**

Code

In [47]:
```
twitter_archive_clean.name.replace(to_replace= 'O', value = 'O\'Malley', inplace =True)
```

Test

In [48]:
```
twitter_archive_clean[twitter_archive_clean.name== 'O\'Malley']
```

Out[48]:

	tweet_id	timestamp	source	text	
775	776201521193218049	2016-09-14 23:30:38+00:00	<a href="http://twitter.com/download/iphone" r...	This is O'Malley. That is how he sleeps. Doesn...	https://twitter.com/dog_

Issue #7:

twitter_archive: Source column has unnecesary characters.

Define

Use RegEx to extract the clean tweet source.

Code

In [49]:
```
twitter_archive_clean['tweet_source']=twitter_archive_clean['source'].str.extract('(?:.*
twitter_archive_clean.tweet_source.replace(to_replace= 'twitter', value = 'twitter web c
twitter_archive_clean = twitter_archive_clean.drop('source', axis=1)
```

Test

In [50]:
```
twitter_archive_clean.tweet_source.value_counts()
```

Out[50]:
```
iphone                1985
vine                    90
twitter web client      30
tweetdeck               11
Name: tweet_source, dtype: int64
```

Issue #8:

image_predict: No column for Dog breed in image_predicted dataframe.

Define

Use For loop to loop through the algorithm's different predictions for the image in the tweet and append the breed and correlating algoritm confidence to new **breed** and confidence columns

Code

In [51]:
```
# Create new list, breed
breed = []

# Convert breed prediction columns to lists
```

```
p1_list = image_predict_clean.p1.tolist()
p2_list = image_predict_clean.p2.tolist()
p3_list = image_predict_clean.p3.tolist()

#  loop through dog / not a dog prediction and append appropriate dog breed to breed lis
for i, j in enumerate(image_predict_clean.p1_dog):
    if j == True:
        breed.append(p1_list[i])
    elif image_predict.p2_dog[i] == True:
            breed.append(p2_list[i])
    elif image_predict.p3_dog[i] == True:
            breed.append(p3_list[i])
    else:
        breed.append(np.nan)
```

In [52]:
```
# For degree of confidence column
confidence = []

# Convert confidence percent columns to lists
p1_conf_list = image_predict_clean.p2_conf.tolist()
p2_conf_list = image_predict_clean.p2_conf.tolist()
p3_conf_list = image_predict_clean.p3_conf.tolist()


# Loop through breed and confidence percent lists to assign correct percentage to confid
for i,j in enumerate(breed):
    if breed[i] == p1_list[i]:
        confidence.append(p1_conf_list[i])
    elif breed[i] == p2_list[i]:
        confidence.append(p2_conf_list[i])
    elif breed[i] == p3_list[i]:
        confidence.append(p3_conf_list[i])
    else:
        confidence.append(np.nan)
```

In [53]:
```
# Add breed list to dataframe
image_predict_clean = image_predict_clean.assign(breed=breed)
image_predict_clean.breed = image_predict_clean.breed.str.replace('_', ' ').str.capitali

# Add confidence list to dataframe
image_predict_clean = image_predict_clean.assign(confidence=confidence)

# Drop the algorithm's prediction columns
image_predict_clean = image_predict_clean.drop([
    'p1','p1_conf','p1_dog','p2','p2_conf','p2_dog','p3','p3_conf','p3_dog'], axis=1)
```

Test

In [54]:
```
image_predict_clean.head(10)
```

Out[54]:

	tweet_id	jpg_url	img_num	breed	confidence
0	666020888022790149	https://pbs.twimg.com/media/CT4udn0WwAA0aMy.jpg	1	Welsh springer spaniel	0.156665
1	666029285002620928	https://pbs.twimg.com/media/CT42GRgUYAA5iDo.jpg	1	Redbone	0.074192
2	666033412701032449	https://pbs.twimg.com/media/CT4521TWwAEvMyu.jpg	1	German shepherd	0.138584
3	666044226329800704	https://pbs.twimg.com/media/CT5Dr8HUEAA-lEu.jpg	1	Rhodesian ridgeback	0.360687
4	666049248165822465	https://pbs.twimg.com/media/CT5IQmsXIAAKY4A.jpg	1	Miniature	0.243682

				pinscher	
5	666050758794694657	https://pbs.twimg.com/media/CT5Jof1WUAEuVxN.jpg	1	Bernese mountain dog	0.263788
6	666051853826850816	https://pbs.twimg.com/media/CT5KoJ1WoAAJash.jpg	1	NaN	NaN
7	666055525042405380	https://pbs.twimg.com/media/CT5N9tpXIAAifs1.jpg	1	Chow	0.058279
8	666057090499244032	https://pbs.twimg.com/media/CT5PY90WoAAQGLo.jpg	1	Golden retriever	0.007959
9	666058600524156928	https://pbs.twimg.com/media/CT5Qw94XAAA_2dP.jpg	1	Miniature poodle	0.192305

Issue #9:

twitter_archive: The doggo, floofer, pupper and puppo columns violate the first rule of tidiness: that each variable forms a column

Define

Melt the variables into one column; dog_stage

Code

In [55]:
```
# Replace 'None' values with Nan
twitter_archive_clean.floofer = twitter_archive_clean.floofer.replace('None', np.nan)
twitter_archive_clean.pupper = twitter_archive_clean.pupper.replace('None', np.nan)
twitter_archive_clean.puppo = twitter_archive_clean.puppo.replace('None', np.nan)
twitter_archive_clean.doggo = twitter_archive_clean.doggo.replace('None', np.nan)
```

In [56]:
```
# Melt the dataframe to create new dog_stage column
twitter_archive_clean = twitter_archive_clean.melt(id_vars=['tweet_id', 'timestamp', 'tw
        'expanded_urls', 'rating_numerator', 'rating_denominator', 'name'],
        var_name = 'stage', value_vars = ['doggo', 'floofer', 'pupper', 'puppo'], value_
```

In [57]:
```
# Move Nan dog_stages down the dataframe and drop duplicates, kepping first occurences b
twitter_archive_clean.sort_values(by = 'dog_stage', inplace=True)
twitter_archive_clean = twitter_archive_clean.drop_duplicates(subset = 'tweet_id')

# Drop 'stage' column
twitter_archive_clean = twitter_archive_clean.drop('stage', axis = 1)

# Set dog_stage dataype to category
twitter_archive_clean.dog_stage = twitter_archive_clean.dog_stage.astype('category')
```

In [58]:
```
# Sort dataframe by index
twitter_archive_clean = twitter_archive_clean.sort_index()
```

Test

In [59]:
```
twitter_archive_clean['dog_stage'].value_counts()
```

Out[59]:
```
pupper     222
doggo       84
puppo       23
floofer      9
Name: dog_stage, dtype: int64
```

Issue #10:

Wrong rating of 75/10

Define

Manually correct the rating_numerator value.

Code

In [60]:
```
twitter_archive_clean.rating_numerator.replace(to_replace = 75, value = 9.75, inplace =
```

Test

In [61]:
```
twitter_archive_clean.rating_numerator.value_counts()
```

Out[61]:
```
12.00      489
10.00      436
11.00      417
13.00      294
9.00       154
8.00        98
7.00        52
14.00       40
5.00        34
6.00        32
3.00        19
4.00        16
2.00         9
1.00         5
0.00         2
60.00        1
88.00        1
420.00       1
144.00       1
26.00        1
121.00       1
143.00       1
44.00        1
1776.00      1
45.00        1
80.00        1
99.00        1
50.00        1
204.00       1
165.00       1
9.75         1
24.00        1
84.00        1
27.00        1
Name: rating_numerator, dtype: int64
```

Issue #11:

rating_numerator and rating_denominator should be merged into one column.

Define

- Create new column, rating, by dividing rating_numerator with rating_denominator then rounding up.
- Create new column, raw_rating, by adding rating_numerator and rating_denominator separated by '/'

Code

In [62]:
```
# Create new column, rating
twitter_archive_clean['rating'] = (
    twitter_archive_clean.rating_numerator / twitter_archive_clean.rating_denominator).r
```

In [63]:
```
# Create new column, raw_rating
twitter_archive_clean['raw_rating'] = twitter_archive_clean['rating_numerator'].astype(i
twitter_archive_clean = twitter_archive_clean.drop(['rating_numerator', 'rating_denomina
```

Test

In [64]:
```
twitter_archive_clean['rating'].value_counts()
```

Out[64]:
```
1      2026
0        85
3         3
178       1
42        1
Name: rating, dtype: int64
```

In [65]:
```
twitter_archive_clean.raw_rating.value_counts()
```

Out[65]:
```
12/10      489
10/10      436
11/10      417
13/10      294
9/10       154
8/10        98
7/10        51
14/10       40
5/10        34
6/10        32
3/10        19
4/10        15
2/10         9
1/10         4
0/10         2
1/2          1
88/80        1
144/120      1
44/40        1
420/10       1
26/10        1
7/11         1
121/110      1
143/130      1
204/170      1
60/50        1
45/50        1
80/80        1
99/90        1
50/50        1
4/20         1
9/11         1
1776/10      1
165/150      1
24/7         1
84/70        1
27/10        1
Name: raw_rating, dtype: int64
```

Issue #12:

image_predict: Not all enteries for P1, P2 and P3 are dogs.

Define

Drop rows that have Nan values for breed

Code

In [66]:
```
image_predict_clean = image_predict_clean.dropna()
```

Test

In [67]:
```
image_predict_clean.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1751 entries, 0 to 2073
Data columns (total 5 columns):
 #   Column      Non-Null Count  Dtype  
---  ------      --------------  -----  
 0   tweet_id    1751 non-null   int64  
 1   jpg_url     1751 non-null   object 
 2   img_num     1751 non-null   int64  
 3   breed       1751 non-null   object 
 4   confidence  1751 non-null   float64
dtypes: float64(1), int64(2), object(2)
memory usage: 82.1+ KB
```

Issue #13:

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

Define

Convert the datatype to String using .astype()

Code

In [68]:
```
# Convert datatype to string
image_predict_clean.tweet_id = image_predict_clean.tweet_id.astype('str')
tweets_df_clean['tweet_id'] = tweets_df_clean.id.astype('str')

# Drop 'id' column
tweets_df_clean = tweets_df_clean.drop('id', axis= 1)
```

Test

In [69]:
```
print(type(twitter_archive_clean.tweet_id[0]))
print(type(image_predict_clean.tweet_id[0]))
print(type(tweets_df_clean.tweet_id[0]))
```

```
<class 'str'>
<class 'str'>
<class 'str'>
```

Issue #14:

twitter_archive dataframe should be merged with tweets_df and image_predict dataframe, with unnecessary columns dropped.

Define

Merge 3 dataframes on tweet_id with Inner Join

Code

In [70]:
```
# Merge all 3 dataframes
twitter_archive_cleaned = twitter_archive_clean.merge(
    tweets_df_clean, on = 'tweet_id').merge(image_predict_clean, on ='tweet_id')

# Drop duplicates
twitter_archive_cleaned = twitter_archive_cleaned.drop_duplicates()
```

Test

In [71]:
```
twitter_archive_cleaned.head()
```

Out[71]:

	tweet_id	timestamp	tweet_source	text	expanded_urls
0	892177421306343426	2017-08-01 00:17:27+00:00	iphone	This is Tilly. She's just checking pup on you....	https://twitter.com/dog_rates/status/892177421...
1	891815181378084864	2017-07-31 00:18:03+00:00	iphone	This is Archie. He is a rare Norwegian Pouncin...	https://twitter.com/dog_rates/status/891815181...
2	891689557279858688	2017-07-30 15:58:51+00:00	iphone	This is Darla. She commenced a snooze mid meal...	https://twitter.com/dog_rates/status/891689557...
3	891327558926688256	2017-07-29 16:00:24+00:00	iphone	This is Franklin. He would like you to stop ca...	https://twitter.com/dog_rates/status/891327558...
4	891087950875897856	2017-07-29 00:08:17+00:00	iphone	Here we have a majestic great white breaching ...	https://twitter.com/dog_rates/status/891087950...

In [72]:
```
twitter_archive_cleaned.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1678 entries, 0 to 1677
Data columns (total 15 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   tweet_id        1678 non-null   object
 1   timestamp       1678 non-null   datetime64[ns, UTC]
 2   tweet_source    1678 non-null   object
 3   text            1678 non-null   object
 4   expanded_urls   1678 non-null   object
 5   name            1197 non-null   object
```

```
 6   dog_stage        259 non-null    category
 7   rating           1678 non-null   int32
 8   raw_rating       1678 non-null   object
 9   retweet_count    1678 non-null   int64
 10  favorite_count   1678 non-null   int64
 11  jpg_url          1678 non-null   object
 12  img_num          1678 non-null   int64
 13  breed            1678 non-null   object
 14  confidence       1678 non-null   float64
dtypes: category(1), datetime64[ns, UTC](1), float64(1), int32(1), int64(3), object(8)
memory usage: 191.9+ KB
```

Storing Data

Save gathered, assessed, and cleaned master dataset to a CSV file named "twitter_archive_master.csv".

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

Analyzing and Visualizing Data

In this section, analyze and visualize your wrangled data. You must produce at least **three (3) insights and one (1) visualization.**

In [74]:
```
# Load dataframe for analysis
df = pd.read_csv('twitter_archive_master.csv')
```

Checking the data info

In [75]:
```
df.head(10)
```

Out[75]:

	tweet_id	timestamp	tweet_source	text	expanded_urls
0	892177421306343426	2017-08-01 00:17:27+00:00	iphone	This is Tilly. She's just checking pup on you....	https://twitter.com/dog_rates/status/892177421...
1	891815181378084864	2017-07-31 00:18:03+00:00	iphone	This is Archie. He is a rare Norwegian Pouncin...	https://twitter.com/dog_rates/status/891815181...
2	891689557279858688	2017-07-30 15:58:51+00:00	iphone	This is Darla. She commenced a snooze mid meal...	https://twitter.com/dog_rates/status/891689557...
3	891327558926688256	2017-07-29 16:00:24+00:00	iphone	This is Franklin. He would like you to stop ca...	https://twitter.com/dog_rates/status/891327558...
4	891087950875897856	2017-07-29 00:08:17+00:00	iphone	Here we have a majestic	https://twitter.com/dog_rates/status/891087950...

				great white breaching ...	
5	890971913173991426	2017-07-28 16:27:12+00:00	iphone	Meet Jax. He enjoys ice cream so much he gets ...	https://gofundme.com/ydvmve-surgery-for-jax,ht...
6	890729181411237888	2017-07-28 00:22:40+00:00	iphone	When you watch your owner call another dog a g...	https://twitter.com/dog_rates/status/890729181...
7	890609185150312448	2017-07-27 16:25:51+00:00	iphone	This is Zoey. She doesn't want to be one of th...	https://twitter.com/dog_rates/status/890609185...
8	890240255349198849	2017-07-26 15:59:51+00:00	iphone	This is Cassie. She is a college pup. Studying...	https://twitter.com/dog_rates/status/890240255...
9	890006608113172480	2017-07-26 00:31:25+00:00	iphone	This is Koda. He is a South Australian decksha...	https://twitter.com/dog_rates/status/890006608...

In [76]:
```
df.tail(10)
```

Out[76]:

	tweet_id	timestamp	tweet_source	text	expanded_urls
1668	802239329049477120	2016-11-25 19:55:35+00:00	iphone	This is Loki. He'll do your taxes for you. Can...	https://twitter.com/dog_rates/status/802239329...
1669	793195938047070209	2016-10-31 21:00:23+00:00	iphone	Say hello to Lily. She's pupset that her costu...	https://twitter.com/dog_rates/status/793195938...
1670	790946055508652032	2016-10-25 16:00:09+00:00	iphone	This is Betty. She's assisting with the dishes...	https://twitter.com/dog_rates/status/790946055...
1671	787717603741622272	2016-10-16 18:11:26+00:00	iphone	This is Tonks. She is a service puppo. Can hea...	https://twitter.com/dog_rates/status/787717603...
1672	756275833623502848	2016-07-21 23:53:04+00:00	iphone	When ur older siblings	https://twitter.com/dog_rates/status/756275833...

				get to play in the deep...		
1673	752519690950500352	2016-07-11 15:07:30+00:00	iphone	Hopefully this puppo on a swing will help get ...	https://twitter.com/dog_rates/status/752519690...	
1674	751132876104687617	2016-07-07 19:16:47+00:00	iphone	This is Cooper. He's just so damn happy. 10/10...	https://twitter.com/dog_rates/status/751132876...	
1675	744995568523612160	2016-06-20 20:49:19+00:00	iphone	This is Abby. She got her face stuck in a glas...	https://twitter.com/dog_rates/status/744995568...	
1676	743253157753532416	2016-06-16 01:25:36+00:00	iphone	This is Kilo. He cannot reach the snackum. Nif...	https://twitter.com/dog_rates/status/743253157...	
1677	738537504001953792	2016-06-03 01:07:16+00:00	iphone	This is Bayley. She fell asleep trying to esca...	https://twitter.com/dog_rates/status/738537504...	

In [77]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1678 entries, 0 to 1677
Data columns (total 15 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   tweet_id        1678 non-null   int64  
 1   timestamp       1678 non-null   object 
 2   tweet_source    1678 non-null   object 
 3   text            1678 non-null   object 
 4   expanded_urls   1678 non-null   object 
 5   name            1197 non-null   object 
 6   dog_stage       259 non-null    object 
 7   rating          1678 non-null   int64  
 8   raw_rating      1678 non-null   object 
 9   retweet_count   1678 non-null   int64  
 10  favorite_count  1678 non-null   int64  
 11  jpg_url         1678 non-null   object 
 12  img_num         1678 non-null   int64  
 13  breed           1678 non-null   object 
 14  confidence      1678 non-null   float64
dtypes: float64(1), int64(5), object(9)
memory usage: 196.8+ KB
```

In [78]:
```
# Convert datatype to string
df.tweet_id = df.tweet_id.astype('str')
```

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

Out[79]:

	rating	retweet_count	favorite_count	img_num	confidence
count	1678.000000	1678.000000	1678.000000	1678.000000	1678.000000
mean	0.985697	2274.331943	7976.435042	1.216329	0.135266
std	0.157607	4140.556771	11755.949188	0.577078	0.101238
min	0.000000	11.000000	66.000000	1.000000	0.000010
25%	1.000000	512.500000	1797.000000	1.000000	0.052987
50%	1.000000	1126.500000	3660.000000	1.000000	0.118710
75%	1.000000	2580.500000	9852.500000	1.000000	0.197506
max	3.000000	70427.000000	144401.000000	4.000000	0.467678

Insights:

1. What are the common dog names?
2. What is the most common dog stage?
3. What is the most liked tweet?
4. What is the average likes for dog stages?
5. Dog(s) with the highest rating.
6. What is the most used tweet source?

In [80]:
```
# to duplicte the dataframe before working
df_clean = df.copy()
```

In [81]:
```
# Convert datatypes appropriately
df_clean.tweet_id = df_clean.tweet_id.astype('str')
df_clean.dog_stage = df_clean.dog_stage.astype('category')
```

In [82]:
```
# Sets the style for the visuals
sns.set_theme(style='darkgrid')
```

---

1. What are the common dog names?

In [83]:
```
df_clean.name.value_counts().head(13)
```

Out[83]:
```
Cooper     10
Oliver      9
Charlie     9
Lucy        9
Tucker      9
Penny       8
Daisy       7
Sadie       7
```

```
Winston     7
Koda        6
Toby        6
Jax         6
Lola        6
Name: name, dtype: int64
```

These names all appear more than 5 times as names of dogs.

2. What is the most common dog stage??

In [84]:
```
df_clean.dog_stage.value_counts()
```

Out[84]:
```
pupper     168
doggo       63
puppo       21
floofer      7
Name: dog_stage, dtype: int64
```

The most common dog stage is **pupper**

3. What is the most liked tweet?

In [85]:
```
# assign tweet(s) with highest likes to max_likes
max_likes = df_clean.favorite_count.max()

df_clean.query('favorite_count == {}'.format(max_likes))
```

Out[85]:

	tweet_id	timestamp	tweet_source	text	expanded_urls	na
621	744234799360020481	2016-06-18 18:26:18+00:00	iphone	Here's a doggo realizing you can stand in a po...	https://twitter.com/dog_rates/status/744234799...	N

In [86]:
```
# Return text of tweet with highest likes
df_clean.query('favorite_count == {}'.format(max_likes)).text.item()
```

Out[86]:
```
"Here's a doggo realizing you can stand in a pool. 13/10 enlightened af (vid by Tina Con
rad) https://t.co/7wE9LTEXC4"
```

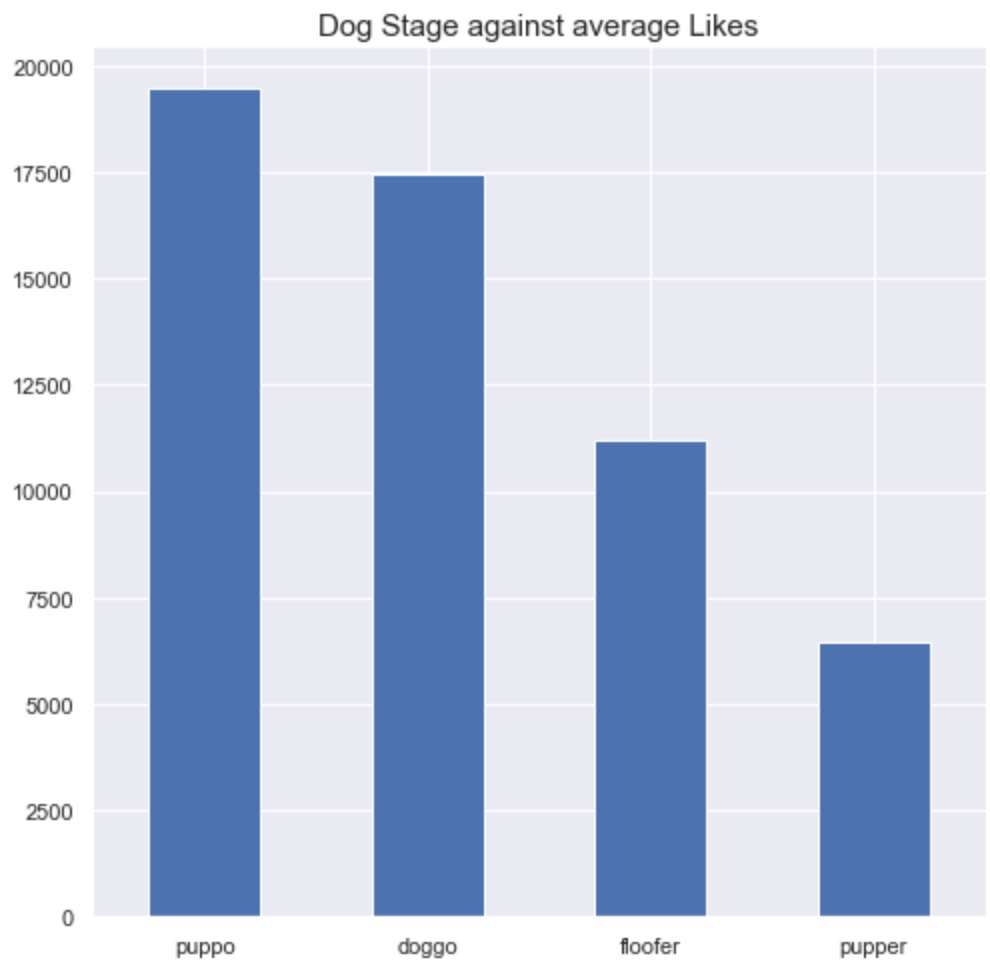
In [87]:
```
# Get image of dog(s) in tweet
url = df_clean.query('favorite_count == {}'.format(max_likes)).jpg_url.item()
r = requests.get(url)

Image.open(BytesIO(r.content))
```

Out[87]:

4. What is the average likes for dog stages?

In [88]:
```
df_clean.groupby('dog_stage')['favorite_count'].mean().sort_values(ascending=False).plot
plt.title('Dog Stage against average Likes', fontsize=15)
plt.xlabel('');
```



On average, Puppos get more tweet likes.

5. Dog(s) with the highest rating?

In [89]:
```
# Assign tweets with highest calculated ratings to max_rating
max_rating = df_clean.rating.max()

df_clean.query('rating == {}'.format(max_rating))
```

Out[89]:

	tweet_id	timestamp	tweet_source	text	expanded_urls
312	810984652412424192	2016-12-19 23:06:23+00:00	iphone	Meet Sam. She smiles 24/7 & secretly aspir...	https://www.gofundme.com/sams-smile,https://tw...
1045	680494726643068929	2015-12-25 21:06:00+00:00	iphone	Here we have uncovered an entire battalion of ...	https://twitter.com/dog_rates/status/680494726...
1515	778027034220126208	2016-09-20 00:24:34+00:00	iphone	This is Sophie. She's a Jubilant Bush Pupper. ...	https://twitter.com/dog_rates/status/778027034...

In [90]:
```
# URL of such tweets
url = df_clean.query('rating == {}'.format(max_rating)).jpg_url.to_list()

r1 = requests.get(url[0])
r2 = requests.get(url[1])
r3 = requests.get(url[2])

# Open their pictures
Image.open(BytesIO(r1.content))
```

Out[90]:

In [91]:
```
Image.open(BytesIO(r2.content))
```

Out[91]:

In [92]:
```
Image.open(BytesIO(r3.content))
```

Out[92]:

6. What is the most used tweet source?

In [93]:
```
df_clean.tweet_source.value_counts()
```

Out[93]:
```
iphone                1648
twitter web client      22
tweetdeck                8
Name: tweet_source, dtype: int64
```

7. Most liked dog breeds

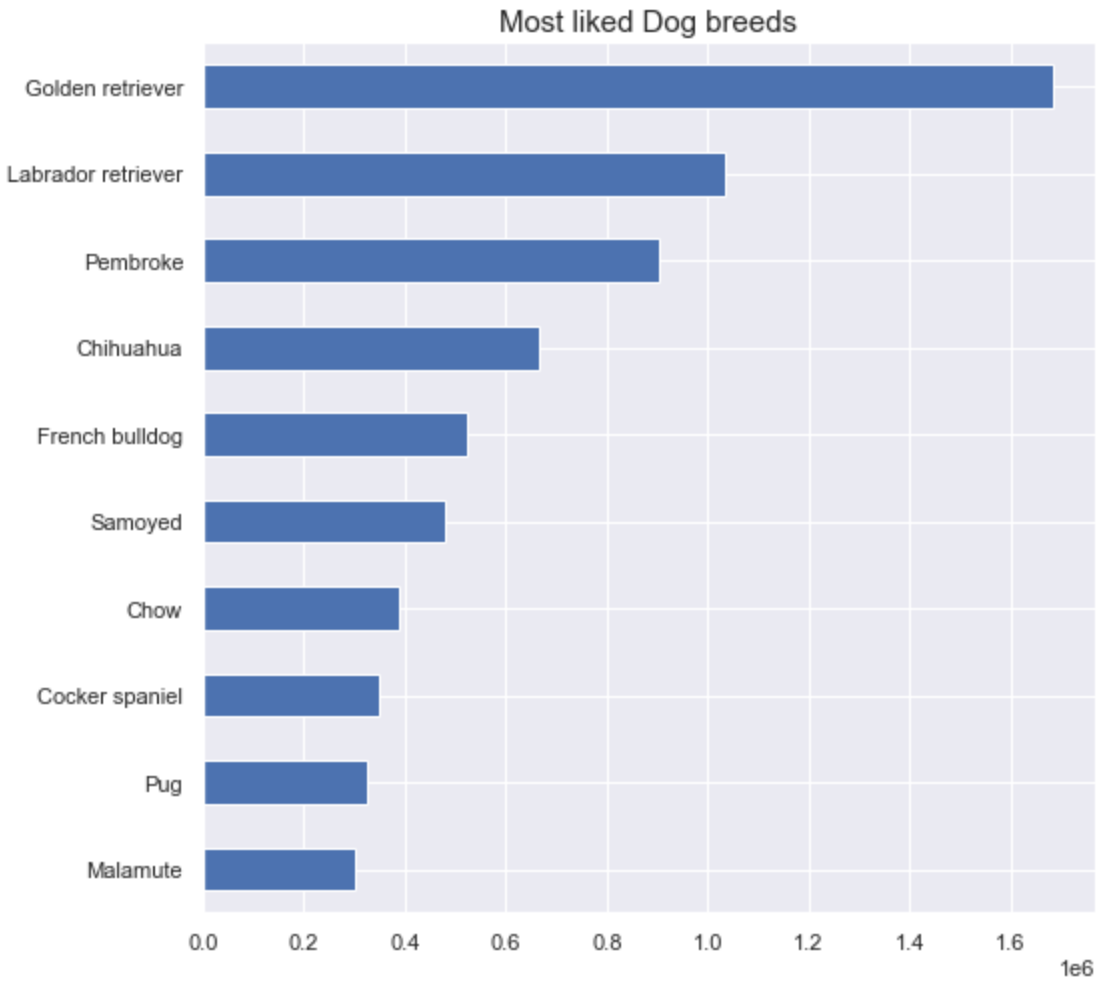
In [94]:
```
df_clean.groupby('breed').sum().sort_values(by = 'favorite_count', ascending = False).he
```

Out[94]:

breed	rating	retweet_count	favorite_count	img_num	confidence
Golden retriever	159	475396	1683384	201	17.470215
Labrador retriever	105	312301	1036514	124	14.235984
Pembroke	93	235959	903248	119	13.365032
Chihuahua	88	210051	667314	112	10.091322
French bulldog	30	131691	524721	35	2.900188
Samoyed	41	155418	480653	48	4.386557
Chow	48	106755	388434	62	5.521922
Cocker spaniel	30	118294	351164	37	4.286072
Pug	61	94035	324427	77	5.288428
Malamute	33	88161	303844	40	6.056324

In [95]:
```
df_clean.groupby('breed').sum().favorite_count.sort_values(ascending = False).head(10).s
plt.title('Most liked Dog breeds', fontsize=15)
```

```
plt.xlabel('');
plt.ylabel('');
```



Visualization

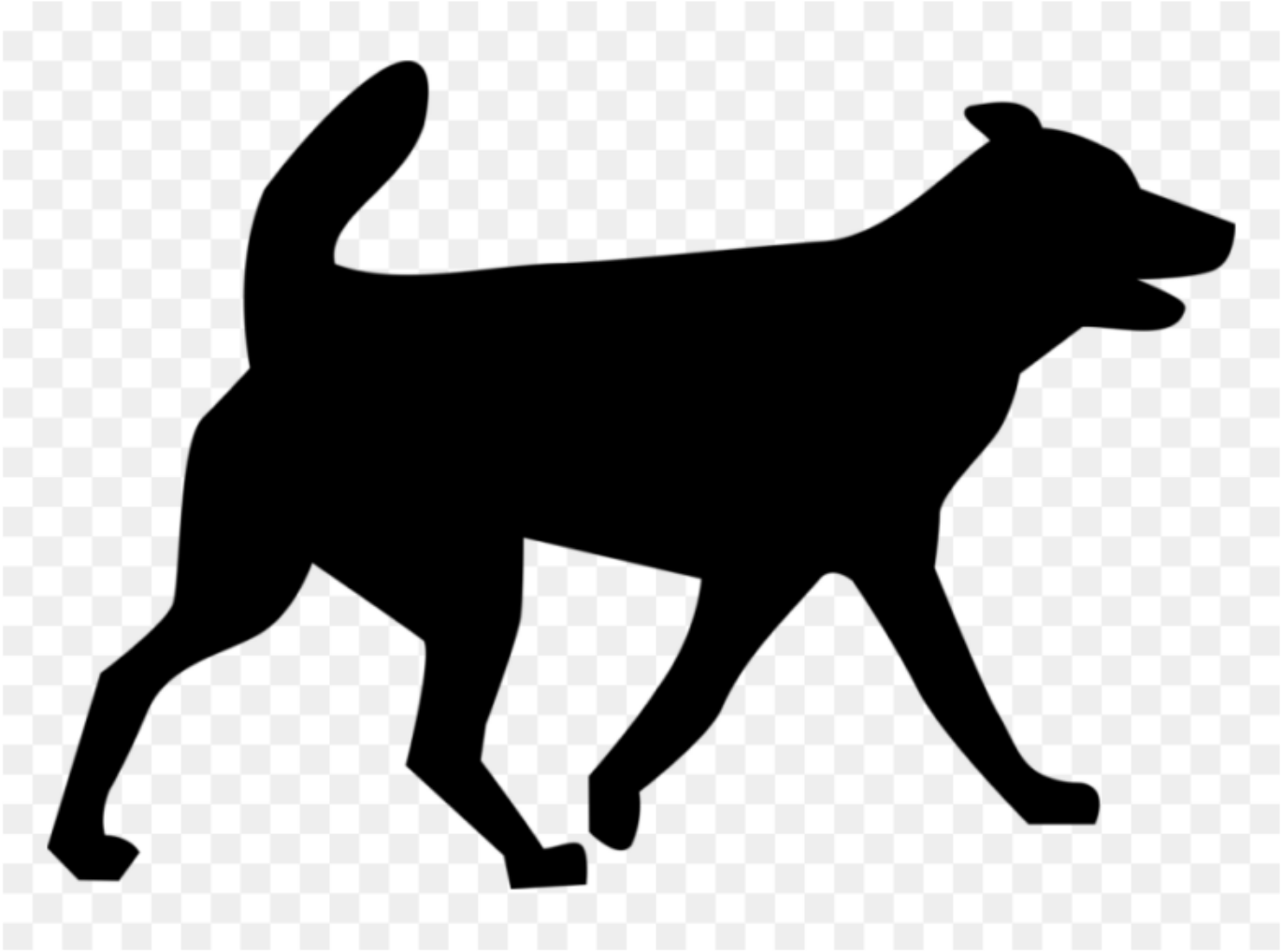
In [96]:
```
# Import dog silhoutte
url = 'https://www.dlf.pt/dfpng/middlepng/57-578964_silhouette-transparent-background-do
r = requests.get(url)
folder_name = 'C:/Users/Zion/Documents/Udacity_Wrangling'

# Download image for wordcloud
i = Image.open(BytesIO(r.content))
i.save(folder_name + "/" + 'dog_clipart' + '.' + 'png')

# Load image for wordcloud
image = np.array(Image.open('dog_clipart.png'))

fig = plt.figure() # Instantiate the figure object
fig.set_figwidth(14) # set width
fig.set_figheight(18) # set height

plt.imshow(image, cmap=plt.cm.gray, interpolation='bilinear') # Display data as an image
plt.axis('off') # Remove axis
plt.show() # Display image
```



In [97]:
```
# Create function to generate the blue colour for the Word CLoud

def blue_color_func(word, font_size, position, orientation, random_state=None,**kwargs):
    return "hsl(210, 100%%, %d%%)" % random.randint(50, 70)
```

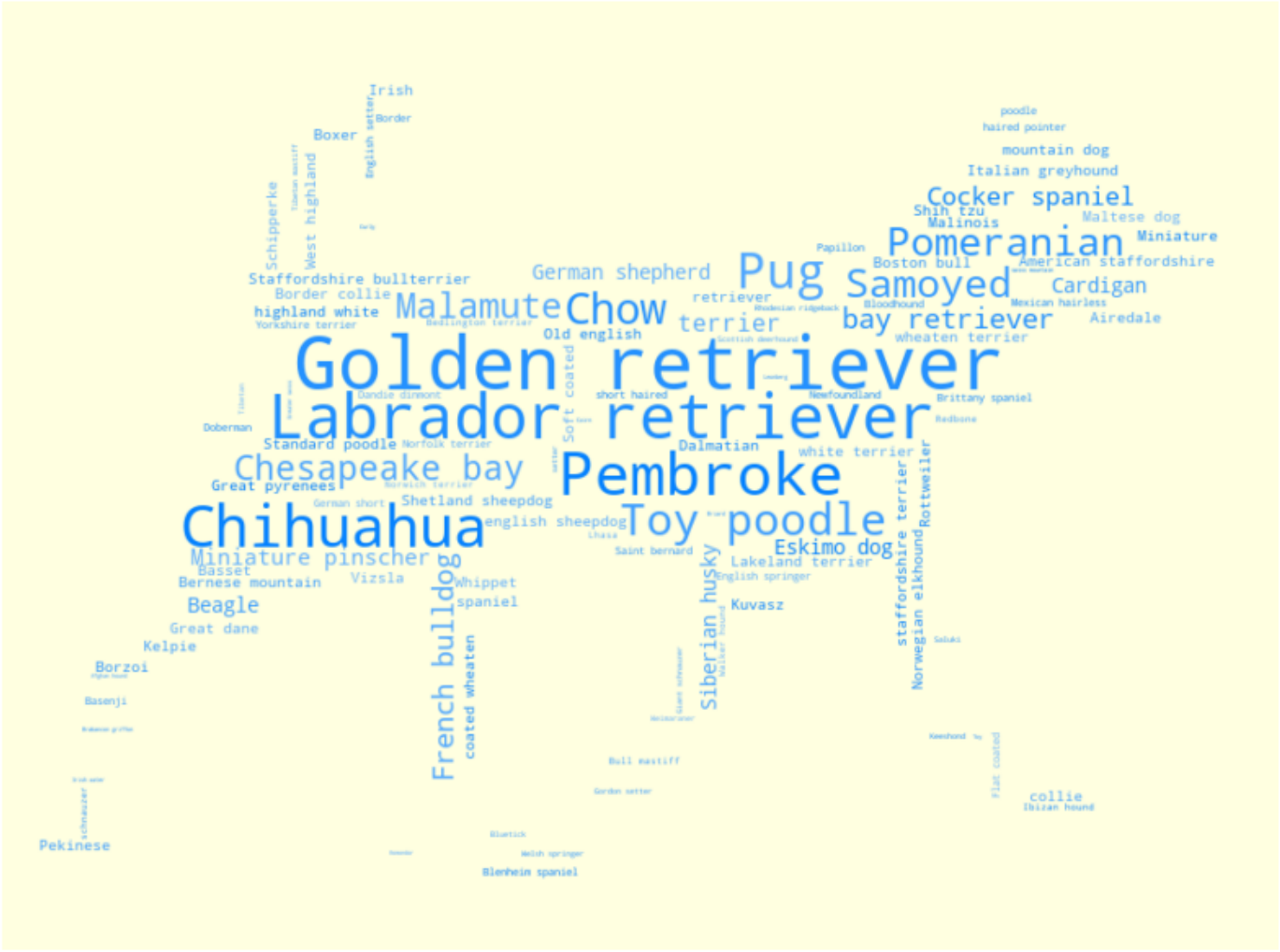
In [98]:
```
# Extract all breed into one long string separated by space
breeds_long_string = df_clean['breed'].replace(" ", "_").tolist()
breeds_long_string = " ".join(breeds_long_string)
```

In [99]:
```
# Instantiate the Twitter word cloud object
wc = WordCloud(mode='RGBA',background_color='lightyellow', max_words=1500, mask=image)

# generate the word cloud
wc.generate(breeds_long_string)

# display the word cloud
fig = plt.figure()
fig.set_figwidth(14)  # set width
fig.set_figheight(18)  # set height

plt.imshow(wc.recolor(color_func=blue_color_func, random_state=3),
           interpolation="bilinear")
plt.axis('off')
plt.show()
```



In [100…

```
# Save to a png file
wc.to_file("breed_wordcloud.png")
```

Out[100]:

```
<wordcloud.wordcloud.WordCloud at 0x18c3a33eca0>
```