# WeRateDogs Project_Wrangling and Analyzing Data from Twitter

*By Shilin Li*

*Date: July 10th, 2018*

# Part I: Data Wrangling

## Introduction

The requirement of current project is to wrangle the tweet archive of **Twitter user @dog_rates (https://twitter.com/dog_rates)** Twitter data, also known as **WeRateDogs (https://en.wikipedia.org/wiki/WeRateDogs)**, to create interesting and trustworthy analyses and visualizations. The Twitter archive is great, but it only contains very basic tweet information. Additional gathering, then assessing and cleaning is required for "Wow!"-worthy analyses and visualizations. In this project, Tweepy is used to query Twitter's API for additional data beyond the data included in the WeRateDogs Twitter archive.

## Gather

Data sources came from three ways shown as below:

- 1 The WeRateDogs Twitter archive is download manually by clicking the following link: **twitter_archive_enhanced.csv (https://d17h27t6h515a5.cloudfront.net/topher/2017/August/59a4e958_twitter-archive-enhanced/twitter-archive-enhanced.csv)**.

- 2 The tweet image predictions, i.e., what breed of dog (or other object, animal, etc.) is present in each tweet according to a neural network. This file (image_predictions.tsv) is hosted on Udacity's servers and should be downloaded programmatically using the Requests library and the following URL: https://d17h27t6h515a5.cloudfront.net/topher/2017/August/599fd2ad_image-predictions/image-predictions.tsv (https://d17h27t6h515a5.cloudfront.net/topher/2017/August/599fd2ad_image-predictions/image-predictions.tsv).

- 3 Each tweet's retweet count and favorite ("like") count at minimum, and any additional data you find interesting. Using the tweet IDs in the WeRateDogs Twitter archive, query the Twitter API for each tweet's JSON data using Python's **Tweepy (http://www.tweepy.org/)** library and store each tweet's entire set of JSON data in a file called tweet_json.txt file. Each tweet's JSON data should be written to its own line. Then read this .txt file line by line into a pandas DataFrame with (at minimum) tweet ID, retweet count, and favorite count.

In [1]:

```
#import major libraries
import pandas as pd
import requests
import os
import tweepy
import json
%matplotlib inline
import matplotlib.pyplot as plt
import seaborn as sns
import time
import numpy as np
from collections import Counter
from IPython.display import Image
import random
import matplotlib
```

```
In [2]:
```

```
# Read the archive data in twitter_archive_enhanced.csv file from local
# print out a few lines to examine file content and structure
archive = pd.read_csv('twitter-archive-enhanced.csv', encoding = 'utf-8')
archive.head()
```

```
Out[2]:
```

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

In [3]:

```
# using Request library to Programmatically download the dog image prediction fi
les
# which is hosted on Udacity server
# https://d17h27t6h515a5.cloudfront.net/topher/2017/August/599fd2ad_image-predic
tions/image-predictions.tsv

# crete storage path and directory
folder_name = 'image_predictions'
if not os.path.exists(folder_name):
    os.makedirs(folder_name)

# download url content by request library
# write into file and save
url = 'https://d17h27t6h515a5.cloudfront.net/topher/2017/August/\
599fd2ad_image-predictions/image-predictions.tsv'
response = requests.get(url)
with open(os.path.join(folder_name, url.split("/")[-1]), mode = 'wb') as file:
        file.write(response.content)

# Read the image prediction file from local
# print out a few lines to examine file content and structure
image = pd.read_csv('image_predictions/image-predictions.tsv', sep = '\t', encod
ing = 'utf-8')
image.head()
```

Out[3]:

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

In [4]:

```
# get API Keys and Tokens for Twitter
# getting tweet JSON data via tweet ID using Tweepy
# Reading and Writing JSON to a File in Python

# https://stackoverflow.com/questions/28384588/twitter-api-get-tweets-with-speci
fic-id
# http://stackabuse.com/reading-and-writing-json-to-a-file-in-python/
# https://www.slickremix.com/docs/how-to-get-api-keys-and-tokens-for-twitter/

consumer_key = 'my consumer_key'
```

```
consumer_secret = 'my consumer_secret'
access_token = 'my access_token'
access_secret = 'my access_secret'

auth = tweepy.OAuthHandler(consumer_key, consumer_secret)
auth.set_access_token(access_token, access_secret)

api = tweepy.API(auth, wait_on_rate_limit = True, # Automatically wait for rate
limits to replenish
                 wait_on_rate_limit_notify = True, # Print a notification when T
weepy is waiting for
#rate limits to replenish
                 parser=tweepy.parsers.JSONParser()) # Parse the result to Json
Object
# https://stackoverflow.com/questions/27900451/convert-tweepy-status-object-into
-json

tweet_ids = list(archive.tweet_id)

tweet_data = {}
error_list = []

# record the start time
start_time = time.time()

# get access to all the tweet content for all the tweet id in archive dataframe(
twit_arc)
for tweet in tweet_ids:
    try:
        tweet_data[str(tweet)] = api.get_status(tweet, tweet_mode='extended')

    # Catch the exceptions of the TweepError
    except:
        print("error of id: " + str(id))
        error_list.append(tweet)

# Calculate the time of excution
end_time = time.time()
print(end_time - start_time)

# write JSON to a File
with open('tweet_json.txt', 'w', encoding = 'utf-8') as outfile:
    json.dump(tweet_data, outfile, indent=4, sort_keys=True, ensure_ascii=False)
```

```
error of id: <built-in function id>
error of id: <built-in function id>
error of id: <built-in function id>
error of id: <built-in function id>
error of id: <built-in function id>
error of id: <built-in function id>
error of id: <built-in function id>
error of id: <built-in function id>
error of id: <built-in function id>
error of id: <built-in function id>
error of id: <built-in function id>
error of id: <built-in function id>

Rate limit reached. Sleeping for: 660

error of id: <built-in function id>

Rate limit reached. Sleeping for: 661

1961.431776046753
```

In [5]:

```
# size of dataframe
print("The number of ids is", len(tweet_data.keys()))
# The number of the errors
print("The number of the errors is", len(error_list))
```

```
The number of ids is 2343
The number of the errors is 13
```

According to the above results:

- Limit of the tweepy API had been reached twice;
- Wait_on_rate_limit automatically wait for rate limits to replenish;
- Wait_on_rate_limit_notify print a notification when Tweepy was waiting;
- The total time was about 1961 seconds (~ 33 min);
- We got 2344 correct tweet_id and 12 errors (we will query those 12 errors later).

In [6]:

```
# Read the tweet data from local
# print out a few lines to examine file content and structure
tweet_df =pd.read_json("tweet_json.txt", orient = 'index')
tweet_df.head()
```

```
Out[6]:
```

| | contributors | coordinates | created_at | display_text_range | er |
|---|---|---|---|---|---|
| **1991-02-08 13:48:08.022790149** | NaN | NaN | 2015-11-15 22:32:08 | [0, 131] | {'hashta []. 'medi [{'displa 'pi... |
| **1991-02-08 16:08:05.002620928** | NaN | NaN | 2015-11-15 23:05:30 | [0, 139] | {'hashta []. 'medi [{'displa 'pi... |
| **1991-02-08 17:16:52.701032449** | NaN | NaN | 2015-11-15 23:21:54 | [0, 130] | {'hashta []. 'medi [{'displa 'pi... |
| **1991-02-08 20:17:06.329800704** | NaN | NaN | 2015-11-16 00:04:52 | [0, 137] | {'hashta []. 'medi [{'displa 'pi... |
| **1991-02-08 21:40:48.165822465** | NaN | NaN | 2015-11-16 00:24:50 | [0, 120] | {'hashta []. 'medi [{'displa 'pi... |

5 rows × 32 columns

## Gather: Summary

By far, the first step of this project is completed. As we know, gathering is key important in the data wrangling process, which largely determine the integrity of the later data analysis. In sum, the data was gathered by the following 3 ways:

- Importing data from an existing file (twitter-archive-enhanced.csv) by pandas;
- Downloading a file according to URL (image-predictions.tsv) by Requests library;
- Querying an API (tweet_json.txt) and geting JSON object of all the tweet_ids using Tweepy.

## Assess

Access is the second step, we will access them visually and programmatically, then recording any quality and tidiness issues found. Those issues will be resolved in the third step, cleaning.

In [7]:

```
# print out the whole archive dataset to assess it visually
archive
```

Out[7]:

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

|  |  |  |  |  |  |
|---|---|---|---|---|---|
|  |  |  |  | +0000 | r... |
| **5** | 891087950875897856 | NaN | NaN | 2017-07-29 00:08:17 +0000 | <a href= r... |
| **6** | 890971913173991426 | NaN | NaN | 2017-07-28 16:27:12 +0000 | <a href= r... |
| **7** | 890729181411237888 | NaN | NaN | 2017-07-28 00:22:40 +0000 | <a href= r... |
| **8** | 890609185150312448 | NaN | NaN | 2017-07-27 16:25:51 +0000 | <a href= r... |
| **9** | 890240255349198849 | NaN | NaN | 2017-07-26 15:59:51 +0000 | <a href= r... |
| **10** | 890006608113172480 | NaN | NaN | 2017-07-26 00:31:25 +0000 | <a href= r... |
| **11** | 889880896479866881 | NaN | NaN | 2017-07-25 16:11:53 +0000 | <a href= r... |
| **12** | 889665388333682689 | NaN | NaN | 2017-07-25 01:55:32 +0000 | <a href= r... |
| **13** | 889638837579907072 | NaN | NaN | 2017-07-25 00:10:02 +0000 | <a href= r... |
| **14** | 889531135344209921 | NaN | NaN | 2017-07-24 17:02:04 | <a href= |

|  |  |  |  |  |  |
|---|---|---|---|---|---|
|  |  |  |  | +0000 | r... |
| **15** | 889278841981685760 | NaN | NaN | 2017-07-24 00:19:32 +0000 | <a href= r... |
| **16** | 888917238123831296 | NaN | NaN | 2017-07-23 00:22:39 +0000 | <a href= r... |
| **17** | 888804989199671297 | NaN | NaN | 2017-07-22 16:56:37 +0000 | <a href= r... |
| **18** | 888554962724278272 | NaN | NaN | 2017-07-22 00:23:06 +0000 | <a href= r... |
| **19** | 888202515573088257 | NaN | NaN | 2017-07-21 01:02:36 +0000 | <a href= r... |
| **20** | 888078434458587136 | NaN | NaN | 2017-07-20 16:49:33 +0000 | <a href= r... |
| **21** | 887705289381826560 | NaN | NaN | 2017-07-19 16:06:48 +0000 | <a href= r... |
| **22** | 887517139158093824 | NaN | NaN | 2017-07-19 03:39:09 +0000 | <a href= r... |
| **23** | 887473957103951883 | NaN | NaN | 2017-07-19 00:47:34 +0000 | <a href= r... |
| **24** | 887343217045368832 | NaN | NaN | 2017-07-18 16:08:03 | <a href= r... |

|  |  |  |  |  |  |
|---|---|---|---|---|---|
|  |  |  |  | +0000 |  |
| **25** | 887101392804085760 | NaN | NaN | 2017-07-18 00:07:08 +0000 | <a href= r... |
| **26** | 886983233522544640 | NaN | NaN | 2017-07-17 16:17:36 +0000 | <a href= r... |
| **27** | 886736880519319552 | NaN | NaN | 2017-07-16 23:58:41 +0000 | <a href= r... |
| **28** | 886680336477933568 | NaN | NaN | 2017-07-16 20:14:00 +0000 | <a href= r... |
| **29** | 886366144734445568 | NaN | NaN | 2017-07-15 23:25:31 +0000 | <a href= r... |
| **...** | ... | ... | ... | ... | ... |
| **2326** | 666411507551481857 | NaN | NaN | 2015-11-17 00:24:19 +0000 | <a href= r... |
| **2327** | 666407126856765440 | NaN | NaN | 2015-11-17 00:06:54 +0000 | <a href= r... |
| **2328** | 666396247373291520 | NaN | NaN | 2015-11-16 23:23:41 +0000 | <a href= r... |
| **2329** | 666373753744588802 | NaN | NaN | 2015-11-16 21:54:18 +0000 | <a href= r... |
| **2330** |  |  |  | 2015-11-16 | <a |

|  |  |  |  |  |  |
|---|---|---|---|---|---|
|  | 666362758909284353 | NaN | NaN | 21:10:36 +0000 | href r... |
| **2331** | 666353288456101888 | NaN | NaN | 2015-11-16 20:32:58 +0000 | <a href r... |
| **2332** | 666345417576210432 | NaN | NaN | 2015-11-16 20:01:42 +0000 | <a href r... |
| **2333** | 666337882303524864 | NaN | NaN | 2015-11-16 19:31:45 +0000 | <a href r... |
| **2334** | 666293911632134144 | NaN | NaN | 2015-11-16 16:37:02 +0000 | <a href r... |
| **2335** | 666287406224695296 | NaN | NaN | 2015-11-16 16:11:11 +0000 | <a href r... |
| **2336** | 666273097616637952 | NaN | NaN | 2015-11-16 15:14:19 +0000 | <a href r... |
| **2337** | 666268910803644416 | NaN | NaN | 2015-11-16 14:57:41 +0000 | <a href r... |
| **2338** | 666104133288665088 | NaN | NaN | 2015-11-16 04:02:55 +0000 | <a href r... |
| **2339** | 666102155909144576 | NaN | NaN | 2015-11-16 03:55:04 +0000 | <a href r... |
| **2340** |  |  |  | 2015-11-16 | <a |

|  |  |  |  |  |  |
|---|---|---|---|---|---|
|  | 666099513787052032 | NaN | NaN | 03:44:34 +0000 | href= r... |
| **2341** | 666094000022159362 | NaN | NaN | 2015-11-16 03:22:39 +0000 | <a href= r... |
| **2342** | 666082916733198337 | NaN | NaN | 2015-11-16 02:38:37 +0000 | <a href= r... |
| **2343** | 666073100786774016 | NaN | NaN | 2015-11-16 01:59:36 +0000 | <a href= r... |
| **2344** | 666071193221509120 | NaN | NaN | 2015-11-16 01:52:02 +0000 | <a href= r... |
| **2345** | 666063827256086533 | NaN | NaN | 2015-11-16 01:22:45 +0000 | <a href= r... |
| **2346** | 666058600524156928 | NaN | NaN | 2015-11-16 01:01:59 +0000 | <a href= r... |
| **2347** | 666057090499244032 | NaN | NaN | 2015-11-16 00:55:59 +0000 | <a href= r... |
| **2348** | 666055525042405380 | NaN | NaN | 2015-11-16 00:49:46 +0000 | <a href= r... |
| **2349** | 666051853826850816 | NaN | NaN | 2015-11-16 00:35:11 +0000 | <a href= r... |
| **2350** | 666050758794694657 | NaN | NaN | 2015-11-16 | <a href= |

|  |  |  |  |  |  |
|---|---|---|---|---|---|
|  |  |  |  | 00:30:50 +0000 | r... |
| **2351** | 666049248165822465 | NaN | NaN | 2015-11-16 00:24:50 +0000 | <a href= r... |
| **2352** | 666044226329800704 | NaN | NaN | 2015-11-16 00:04:52 +0000 | <a href= r... |
| **2353** | 666033412701032449 | NaN | NaN | 2015-11-15 23:21:54 +0000 | <a href= r... |
| **2354** | 666029285002620928 | NaN | NaN | 2015-11-15 23:05:30 +0000 | <a href= r... |
| **2355** | 666020888022790149 | NaN | NaN | 2015-11-15 22:32:08 +0000 | <a href= r... |

In [8]:

```
# assessing the data programmaticaly
archive.info()
archive.describe()
```

```
<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
```

Out[8]:

| | tweet_id | in_reply_to_status_id | in_reply_to_user_id | retweeted_status_id | r |
|---|---|---|---|---|---|
| **count** | 2.356000e+03 | 7.800000e+01 | 7.800000e+01 | 1.810000e+02 | 1 |
| **mean** | 7.427716e+17 | 7.455079e+17 | 2.014171e+16 | 7.720400e+17 | 1 |
| **std** | 6.856705e+16 | 7.582492e+16 | 1.252797e+17 | 6.236928e+16 | 9 |
| **min** | 6.660209e+17 | 6.658147e+17 | 1.185634e+07 | 6.661041e+17 | 7 |
| **25%** | 6.783989e+17 | 6.757419e+17 | 3.086374e+08 | 7.186315e+17 | 4 |
| **50%** | 7.196279e+17 | 7.038708e+17 | 4.196984e+09 | 7.804657e+17 | 4 |
| **75%** | 7.993373e+17 | 8.257804e+17 | 4.196984e+09 | 8.203146e+17 | 4 |
| **max** | 8.924206e+17 | 8.862664e+17 | 8.405479e+17 | 8.874740e+17 | 7 |

In [9]:

```python
# examine the name components programmaticaly
archive['name'].unique()
```

Out[9]:

```
array(['Phineas', 'Tilly', 'Archie', 'Darla', 'Franklin', 'None', 'J
ax',
       'Zoey', 'Cassie', 'Koda', 'Bruno', 'Ted', 'Stuart', 'Oliver',
'Jim',
       'Zeke', 'Ralphus', 'Canela', 'Gerald', 'Jeffrey', 'such', 'Ma
```

```
        'Zeke', 'Ralphus', 'Canela', 'Gerald', 'Jeffrey', 'such', 'Ma
ya',
        'Mingus', 'Derek', 'Roscoe', 'Waffles', 'Jimbo', 'Maisey', 'L
illy',
        'Earl', 'Lola', 'Kevin', 'Yogi', 'Noah', 'Bella', 'Grizzwald'
,
        'Rusty', 'Gus', 'Stanley', 'Alfy', 'Koko', 'Rey', 'Gary', 'a'
,
        'Elliot', 'Louis', 'Jesse', 'Romeo', 'Bailey', 'Duddles', 'Ja
ck',
        'Emmy', 'Steven', 'Beau', 'Snoopy', 'Shadow', 'Terrance', 'Aj
a',
        'Penny', 'Dante', 'Nelly', 'Ginger', 'Benedict', 'Venti', 'Go
ose',
        'Nugget', 'Cash', 'Coco', 'Jed', 'Sebastian', 'Walter', 'Sier
ra',
        'Monkey', 'Harry', 'Kody', 'Lassie', 'Rover', 'Napolean', 'Da
wn',
        'Boomer', 'Cody', 'Rumble', 'Clifford', 'quite', 'Dewey', 'Sc
out',
        'Gizmo', 'Cooper', 'Harold', 'Shikha', 'Jamesy', 'Lili', 'Sam
my',
        'Meatball', 'Paisley', 'Albus', 'Neptune', 'Quinn', 'Belle',
        'Zooey', 'Dave', 'Jersey', 'Hobbes', 'Burt', 'Lorenzo', 'Carl
',
        'Jordy', 'Milky', 'Trooper', 'Winston', 'Sophie', 'Wyatt', 'R
osie',
        'Thor', 'Oscar', 'Luna', 'Callie', 'Cermet', 'George', 'Marle
e',
        'Arya', 'Einstein', 'Alice', 'Rumpole', 'Benny', 'Aspen', 'Ja
rod',
        'Wiggles', 'General', 'Sailor', 'Astrid', 'Iggy', 'Snoop', 'K
yle',
        'Leo', 'Riley', 'Gidget', 'Noosh', 'Odin', 'Jerry', 'Charlie'
,
        'Georgie', 'Rontu', 'Cannon', 'Furzey', 'Daisy', 'Tuck', 'Bar
ney',
        'Vixen', 'Jarvis', 'Mimosa', 'Pickles', 'Bungalo', 'Brady', '
Margo',
        'Sadie', 'Hank', 'Tycho', 'Stephan', 'Indie', 'Winnie', 'Bent
ley',
        'Ken', 'Max', 'Maddie', 'Pipsy', 'Monty', 'Sojourner', 'Odie'
,
        'Arlo', 'Sunny', 'Vincent', 'Lucy', 'Clark', 'Mookie', 'Meera
',
        'Buddy', 'Ava', 'Rory', 'Eli', 'Ash', 'Tucker', 'Tobi', 'Ches
ter',
        'Wilson', 'Sunshine', 'Lipton', 'Gabby', 'Bronte', 'Poppy', '
Rhino',
        'Willow', 'not', 'Orion', 'Eevee', 'Smiley', 'Logan', 'Moreto
n',
        'Klein', 'Miguel', 'Emanuel', 'Kuyu', 'Dutch', 'Pete', 'Scoot
er',
        'Reggie', 'Kyro', 'Samson', 'Loki', 'Mia', 'Malcolm', 'Dexter
'
```

```
',
       'Alfie', 'Fiona', 'one', 'Mutt', 'Bear', 'Doobert', 'Beebop',
       'Alexander', 'Sailer', 'Brutus', 'Kona', 'Boots', 'Ralphie',
'Phil',
       'Cupid', 'Pawnd', 'Pilot', 'Ike', 'Mo', 'Toby', 'Sweet', 'Pab
lo',
       'Nala', 'Balto', 'Crawford', 'Gabe', 'Mattie', 'Jimison',
       'Hercules', 'Duchess', 'Harlso', 'Sampson', 'Sundance', 'Luca
',
       'Flash', 'Finn', 'Peaches', 'Howie', 'Jazzy', 'Anna', 'Bo',
       'Seamus', 'Wafer', 'Chelsea', 'Tom', 'Moose', 'Florence', 'Au
tumn',
       'Dido', 'Eugene', 'Herschel', 'Strudel', 'Tebow', 'Chloe', 'B
etty',
       'Timber', 'Binky', 'Dudley', 'Comet', 'Larry', 'Levi', 'Akumi
',
       'Titan', 'Olivia', 'Alf', 'Oshie', 'Bruce', 'Chubbs', 'Sky',
       'Atlas', 'Eleanor', 'Layla', 'Rocky', 'Baron', 'Tyr', 'Bauer'
,
       'Swagger', 'Brandi', 'Mary', 'Moe', 'Halo', 'Augie', 'Craig',
'Sam',
       'Hunter', 'Pavlov', 'Maximus', 'Wallace', 'Ito', 'Milo', 'Oll
ie',
       'Cali', 'Lennon', 'incredibly', 'Major', 'Duke', 'Reginald',
       'Sansa', 'Shooter', 'Django', 'Diogi', 'Sonny', 'Philbert',
       'Marley', 'Severus', 'Ronnie', 'Anakin', 'Bones', 'Mauve', 'C
hef',
       'Doc', 'Sobe', 'Longfellow', 'Mister', 'Iroh', 'Baloo', 'Stub
ert',
       'Paull', 'Timison', 'Davey', 'Pancake', 'Tyrone', 'Snicku', '
Ruby',
       'Brody', 'Rizzy', 'Mack', 'Butter', 'Nimbus', 'Laika', 'Dobby
',
       'Juno', 'Maude', 'Lily', 'Newt', 'Benji', 'Nida', 'Robin',
       'Monster', 'BeBe', 'Remus', 'Mabel', 'Misty', 'Happy', 'Mosby
',
       'Maggie', 'Leela', 'Ralphy', 'Brownie', 'Meyer', 'Stella', 'm
ad',
       'Frank', 'Tonks', 'Lincoln', 'Oakley', 'Dale', 'Rizzo', 'Arni
e',
       'Pinot', 'Dallas', 'Hero', 'Frankie', 'Stormy', 'Mairi', 'Loo
mis',
       'Godi', 'Kenny', 'Deacon', 'Timmy', 'Harper', 'Chipson', 'Com
bo',
       'Dash', 'Bell', 'Hurley', 'Jay', 'Mya', 'Strider', 'an', 'Wes
ley',
       'Solomon', 'Huck', 'very', 'O', 'Blue', 'Finley', 'Sprinkles'
,
       'Heinrich', 'Shakespeare', 'Fizz', 'Chip', 'Grey', 'Roosevelt
',
       'Gromit', 'Willem', 'Dakota', 'Dixie', 'Al', 'Jackson', 'just
',
       'Carbon', 'DonDon', 'Kirby', 'Lou', 'Nollie', 'Chevy', 'Tito'
```

```
,
       'Louie', 'Rupert', 'Rufus', 'Brudge', 'Shadoe', 'Colby', 'Ang
el',
       'Brat', 'Tove', 'my', 'Aubie', 'Kota', 'Eve', 'Glenn', 'Shelb
y',
       'Sephie', 'Bonaparte', 'Albert', 'Wishes', 'Rose', 'Theo', 'R
occo',
       'Fido', 'Emma', 'Spencer', 'Lilli', 'Boston', 'Brandonald', '
Corey',
       'Leonard', 'Chompsky', 'Beckham', 'Devón', 'Gert', 'Watson',
       'Rubio', 'Keith', 'Dex', 'Carly', 'Ace', 'Tayzie', 'Grizzie',
       'Fred', 'Gilbert', 'Zoe', 'Stewie', 'Calvin', 'Lilah', 'Spank
y',
       'Jameson', 'Piper', 'Atticus', 'Blu', 'Dietrich', 'Divine', '
Tripp',
       'his', 'Cora', 'Huxley', 'Keurig', 'Bookstore', 'Linus', 'Abb
y',
       'Shaggy', 'Shiloh', 'Gustav', 'Arlen', 'Percy', 'Lenox', 'Sug
ar',
       'Harvey', 'Blanket', 'actually', 'Geno', 'Stark', 'Beya', 'Ki
lo',
       'Kayla', 'Maxaroni', 'Doug', 'Edmund', 'Aqua', 'Theodore', 'C
hase',
       'getting', 'Rorie', 'Simba', 'Charles', 'Bayley', 'Axel',
       'Storkson', 'Remy', 'Chadrick', 'Kellogg', 'Buckley', 'Livvie
',
       'Terry', 'Hermione', 'Ralpher', 'Aldrick', 'this', 'unaccepta
ble',
       'Rooney', 'Crystal', 'Ziva', 'Stefan', 'Pupcasso', 'Puff',
       'Flurpson', 'Coleman', 'Enchilada', 'Raymond', 'all', 'Rueben
',
       'Cilantro', 'Karll', 'Sprout', 'Blitz', 'Bloop', 'Lillie',
       'Ashleigh', 'Kreggory', 'Sarge', 'Luther', 'Ivar', 'Jangle',
       'Schnitzel', 'Panda', 'Berkeley', 'Ralphé', 'Charleson', 'Cly
de',
       'Harnold', 'Sid', 'Pippa', 'Otis', 'Carper', 'Bowie',
       'Alexanderson', 'Suki', 'Barclay', 'Skittle', 'Ebby', 'Flávio
',
       'Smokey', 'Link', 'Jennifur', 'Ozzy', 'Bluebert', 'Stephanus'
,
       'Bubbles', 'old', 'Zeus', 'Bertson', 'Nico', 'Michelangelope'
,
       'Siba', 'Calbert', 'Curtis', 'Travis', 'Thumas', 'Kanu', 'Lan
ce',
       'Opie', 'Kane', 'Olive', 'Chuckles', 'Staniel', 'Sora', 'Beem
o',
       'Gunner', 'infuriating', 'Lacy', 'Tater', 'Olaf', 'Cecil', 'V
ince',
       'Karma', 'Billy', 'Walker', 'Rodney', 'Klevin', 'Malikai', 'B
obble',
       'River', 'Jebberson', 'Remington', 'Farfle', 'Jiminus', 'Clar
kus',
       'Finnegus', 'Cupcake', 'Kathmandu', 'Ellie', 'Katie', 'Kara',
       'Adele', 'Zara', 'Ambrose', 'Jimothy', 'Bode', 'Terrenth', 'R
eese',
       'Chesterson', 'Lucia', 'Bisquick', 'Ralphson', 'Socks', 'Ramb
o',
       'Rudy', 'Fiji', 'Rilo', 'Bilbo', 'Coopson', 'Yoda', 'Millie',
       'Chet', 'Crouton', 'Daniel', 'Kaia', 'Murphy', 'Dotsy', 'Eazy
',
       'Coops', 'Fillup', 'Miley', 'Charl', 'Reagan', 'Yukon', 'CeCe
',
       'Cuddles', 'Claude', 'Jessiga', 'Carter', 'Ole', 'Pherb', 'Bl
ipson',
       'Reptar', 'Trevith', 'Berb', 'Bob', 'Colin', 'Brian', 'Olivié
r',
       'Grady', 'Kobe', 'Freddery', 'Bodie', 'Dunkin', 'Wally', 'Tup
awc',
       'Amber', 'Edgar', 'Teddy', 'Kingsley', 'Brockly', 'Richie', '
Molly',
       'Vinscent', 'Cedrick', 'Hazel', 'Lolo', 'Eriq', 'Phred', 'the
',
       'Oddie', 'Maxwell', 'Geoff', 'Covach', 'Durg', 'Fynn', 'Ricky
',
       'Herald', 'Lucky', 'Ferg', 'Trip', 'Clarence', 'Hamrick', 'Br
ad',
       'Pubert', 'Frönq', 'Derby', 'Lizzie', 'Ember', 'Blakely', 'Op
al',
       'Marq', 'Kramer', 'Barry', 'Gordon', 'Baxter', 'Mona', 'Horac
e',
       'Crimson', 'Birf', 'Hammond', 'Lorelei', 'Marty', 'Brooks',
       'Petrick', 'Hubertson', 'Gerbald', 'Oreo', 'Bruiser', 'Perry'
,
       'Bobby', 'Jeph', 'Obi', 'Tino', 'Kulet', 'Sweets', 'Lupe', 'T
iger',
       'Jiminy', 'Griffin', 'Banjo', 'Brandy', 'Lulu', 'Darrel', 'Ta
co',
       'Joey', 'Patrick', 'Kreg', 'Todo', 'Tess', 'Ulysses', 'Toffee
',
       'Apollo', 'Asher', 'Glacier', 'Chuck', 'Champ', 'Ozzie', 'Gri
swold',
       'Cheesy', 'Moofasa', 'Hector', 'Goliath', 'Kawhi', 'by', 'Emm
ie',
       'Penelope', 'Willie', 'Rinna', 'Mike', 'William', 'Dwight', '
Evy',
       'officially', 'Rascal', 'Linda', 'Tug', 'Tango', 'Grizz', 'Je
rome',
       'Crumpet', 'Jessifer', 'Izzy', 'Ralph', 'Sandy', 'Humphrey',
       'Tassy', 'Juckson', 'Chuq', 'Tyrus', 'Karl', 'Godzilla', 'Vin
nie',
       'Kenneth', 'Herm', 'Bert', 'Striker', 'Donny', 'Pepper', 'Ber
nie',
       'Buddah', 'Lenny', 'Arnold', 'Zuzu', 'Mollie', 'Laela', 'Tedd
ers',
       'Superpup', 'Rufio', 'Jeb', 'Rodman', 'Jonah', 'Chesney', 'li
fe',
       'Henry', 'Bobbay', 'Mitch', 'Kaiya', 'Acro', 'Aiden', 'Obie',

'Dot',
       'Shnuggles', 'Kendall', 'Jeffri', 'Steve', 'Mac', 'Fletcher',
       'Kenzie', 'Pumpkin', 'Schnozz', 'Gustaf', 'Cheryl', 'Ed',
       'Leonidas', 'Norman', 'Caryl', 'Scott', 'Taz', 'Darby', 'Jack
ie',
       'light', 'Jazz', 'Franq', 'Pippin', 'Rolf', 'Snickers', 'Ridl
ey',
       'Cal', 'Bradley', 'Bubba', 'Tuco', 'Patch', 'Mojo', 'Batdog',
       'Dylan', 'space', 'Mark', 'JD', 'Alejandro', 'Scruffers', 'Pi
p',
       'Julius', 'Tanner', 'Sparky', 'Anthony', 'Holly', 'Jett', 'Am
y',
       'Sage', 'Andy', 'Mason', 'Trigger', 'Antony', 'Creg', 'Travis
s',
       'Gin', 'Jeffrie', 'Danny', 'Ester', 'Pluto', 'Bloo', 'Edd', '
Willy',
       'Herb', 'Damon', 'Peanut', 'Nigel', 'Butters', 'Sandra', 'Fab
io',
       'Randall', 'Liam', 'Tommy', 'Ben', 'Raphael', 'Julio', 'Andru
',
       'Kloey', 'Shawwn', 'Skye', 'Kollin', 'Ronduh', 'Billl', 'Sayd
ee',
       'Dug', 'Tessa', 'Sully', 'Kirk', 'Ralf', 'Clarq', 'Jaspers',
       'Samsom', 'Harrison', 'Chaz', 'Jeremy', 'Jaycob', 'Lambeau',
       'Ruffles', 'Amélie', 'Bobb', 'Banditt', 'Kevon', 'Winifred',
'Hanz',
       'Churlie', 'Zeek', 'Timofy', 'Maks', 'Jomathan', 'Kallie', 'M
arvin',
       'Spark', 'Gòrdón', 'Jo', 'DayZ', 'Jareld', 'Torque', 'Ron',
       'Skittles', 'Cleopatricia', 'Erik', 'Stu', 'Tedrick', 'Filup'
,
       'Kial', 'Naphaniel', 'Dook', 'Hall', 'Philippe', 'Biden', 'Fw
ed',
       'Genevieve', 'Joshwa', 'Bradlay', 'Clybe', 'Keet', 'Carll',
       'Jockson', 'Josep', 'Lugan', 'Christoper'], dtype=object)
```

In [10]:

```
# randomly check info in 'text' column
random.choice(archive.text.tolist())
```

Out[10]:

```
"This is Crystal. She's a shitty fireman. No sense of urgency. Peopl
e could be dying Crystal. 2/10 just irresponsible https://t.co/rtMtj
Sl9pz"
```

In [27]:

```
# randomly compare the rating numerator and denominator in 'text', 'numerator' a
nd 'denominator' column
random.choice(archive.text.tolist()), random.choice(archive.rating_numerator.tol
ist()), \
random.choice(archive.rating_denominator.tolist())
```

Out[27]:

```
('This is Cheryl AKA Queen Pupper of the Skies. Experienced fighter
pilot. Much skill. True hero. 11/10 https://t.co/i4XJEWwdsp',
 12,
 10)
```

In [28]:

```
image
```

Out[28]:

| | tweet_id | jpg_url | img_ |
|---|---|---|---|
| **0** | 666020888022790149 | https://pbs.twimg.com/media/CT4udn0WwAA0aMy.jpg | 1 |
| **1** | 666029285002620928 | https://pbs.twimg.com/media/CT42GRgUYAA5iDo.jpg | 1 |
| **2** | 666033412701032449 | https://pbs.twimg.com/media/CT4521TWwAEvMyu.jpg | 1 |
| **3** | 666044226329800704 | https://pbs.twimg.com/media/CT5Dr8HUEAA-lEu.jpg | 1 |
| **4** | 666049248165822465 | https://pbs.twimg.com/media/CT5IQmsXIAAKY4A.jpg | 1 |
| **5** | 666050758794694657 | https://pbs.twimg.com/media/CT5Jof1WUAEuVxN.jpg | 1 |
| **6** | 666051853826850816 | https://pbs.twimg.com/media/CT5KoJ1WoAAJash.jpg | 1 |
| **7** | 666055525042405380 | https://pbs.twimg.com/media/CT5N9tpXIAAifs1.jpg | 1 |
| **8** | 666057090499244032 | https://pbs.twimg.com/media/CT5PY90WoAAQGLo.jpg | 1 |
| **9** | 666058600524156928 | https://pbs.twimg.com/media/CT5Qw94XAAA_2dP.jpg | 1 |
| **10** | 666063827256086533 | https://pbs.twimg.com/media/CT5Vg_wXIAAXfnj.jpg | 1 |
| **11** | 666071193221509120 | https://pbs.twimg.com/media/CT5cN_3WEAAlOoZ.jpg | 1 |
| **12** | 666073100786774016 | https://pbs.twimg.com/media/CT5d9DZXAAALcwe.jpg | 1 |
| **13** | 666082916733198337 | https://pbs.twimg.com/media/CT5m4VGWEAAtKc8.jpg | 1 |
| **14** | 666094000022159362 | https://pbs.twimg.com/media/CT5w9gUW4AAsBNN.jpg | 1 |
| **15** | 666099513787052032 | https://pbs.twimg.com/media/CT51-JJUEAA6hV8.jpg | 1 |
| **16** | 666102155909144576 | https://pbs.twimg.com/media/CT54YGiWUAEZnoK.jpg | 1 |
| **17** | 666104133288665088 | https://pbs.twimg.com/media/CT56LSZWoAAlJj2.jpg | 1 |

| | | | |
|---|---|---|---|
| **18** | 666268910803644416 | https://pbs.twimg.com/media/CT8QCd1WEAADXws.jpg | 1 |
| **19** | 666273097616637952 | https://pbs.twimg.com/media/CT8T1mtUwAA3aqm.jpg | 1 |
| **20** | 666287406224695296 | https://pbs.twimg.com/media/CT8g3BpUEAAuFjg.jpg | 1 |
| **21** | 666293911632134144 | https://pbs.twimg.com/media/CT8mx7KW4AEQu8N.jpg | 1 |
| **22** | 666337882303524864 | https://pbs.twimg.com/media/CT9OwFIWEAMuRje.jpg | 1 |
| **23** | 666345417576210432 | https://pbs.twimg.com/media/CT9Vn7PWoAA_ZCM.jpg | 1 |
| **24** | 666353288456101888 | https://pbs.twimg.com/media/CT9cx0tUEAAhNN_.jpg | 1 |
| **25** | 666362758909284353 | https://pbs.twimg.com/media/CT9lXGsUcAAyUFt.jpg | 1 |
| **26** | 666373753744588802 | https://pbs.twimg.com/media/CT9vZEYWUAAlZ05.jpg | 1 |
| **27** | 666396247373291520 | https://pbs.twimg.com/media/CT-D2ZHWIAA3gK1.jpg | 1 |
| **28** | 666407126856765440 | https://pbs.twimg.com/media/CT-NvwmW4AAugGZ.jpg | 1 |
| **29** | 666411507551481857 | https://pbs.twimg.com/media/CT-RugiWIAELEaq.jpg | 1 |
| **...** | ... | ... | ... |
| **2045** | 886366144734445568 | https://pbs.twimg.com/media/DE0BTnQUwAApKEH.jpg | 1 |
| **2046** | 886680336477933568 | https://pbs.twimg.com/media/DE4fEDzWAAAyHMM.jpg | 1 |
| **2047** | 886736880519319552 | https://pbs.twimg.com/media/DE5Se8FXcAAJFx4.jpg | 1 |
| **2048** | 886983233522544640 | https://pbs.twimg.com/media/DE8yicJW0AAAvBJ.jpg | 2 |
| **2049** | 887101392804085760 | https://pbs.twimg.com/media/DE-eAq6UwAA-jaE.jpg | 1 |
| **2050** | 887343217045368832 | https://pbs.twimg.com/ext_tw_video_thumb/88734... | 1 |
| **2051** | 887473957103951883 | https://pbs.twimg.com/media/DFDw2tyUQAAAFke.jpg | 2 |
| **2052** | 887517139158093824 | https://pbs.twimg.com/ext_tw_video_thumb/88751... | 1 |
| **2053** | 887705289381826560 | https://pbs.twimg.com/media/DFHDQBbXgAEqY7t.jpg | 1 |
| **2054** | 888078434458587136 | https://pbs.twimg.com/media/DFMWn56WsAAkA7B.jpg | 1 |
| **2055** | 888202515573088257 | https://pbs.twimg.com/media/DFDw2tyUQAAAFke.jpg | 2 |
| **2056** | 888554962724278272 | https://pbs.twimg.com/media/DFTH_O-UQAACu20.jpg | 3 |
| **2057** | 888804989199671297 | https://pbs.twimg.com/media/DFWra-3VYAA2piG.jpg | 1 |
| **2058** | 888917238123831296 | https://pbs.twimg.com/media/DFYRgsOUQAARGhO.jpg | 1 |
| **2059** | 889278841981685760 | https://pbs.twimg.com/ext_tw_video_thumb/88927... | 1 |
| **2060** | 889531135344209921 | https://pbs.twimg.com/media/DFg_2PVW0AEHN3p.jpg | 1 |
| **2061** | 889638837579907072 | https://pbs.twimg.com/media/DFihzFfXsAYGDPR.jpg | 1 |
| **2062** | 889665388333682689 | https://pbs.twimg.com/media/DFi579UWsAAatzw.jpg | 1 |

|  |  |  |  |
|---|---|---|---|
| **2063** | 889880896479866881 | https://pbs.twimg.com/media/DFl99B1WsAITKsg.jpg | 1 |
| **2064** | 890006608113172480 | https://pbs.twimg.com/media/DFnwSY4WAAAMliS.jpg | 1 |
| **2065** | 890240255349198849 | https://pbs.twimg.com/media/DFrEyVuW0AAO3t9.jpg | 1 |
| **2066** | 890609185150312448 | https://pbs.twimg.com/media/DFwUU__XcAEpyXI.jpg | 1 |
| **2067** | 890729181411237888 | https://pbs.twimg.com/media/DFyBahAVwAAhUTd.jpg | 2 |
| **2068** | 890971913173991426 | https://pbs.twimg.com/media/DF1eOmZXUAALUcq.jpg | 1 |
| **2069** | 891087950875897856 | https://pbs.twimg.com/media/DF3HwyEWsAABqE6.jpg | 1 |
| **2070** | 891327558926688256 | https://pbs.twimg.com/media/DF6hr6BUMAAzZgT.jpg | 2 |
| **2071** | 891689557279858688 | https://pbs.twimg.com/media/DF_q7IAWsAEuuN8.jpg | 1 |
| **2072** | 891815181378084864 | https://pbs.twimg.com/media/DGBdLU1WsAANxJ9.jpg | 1 |
| **2073** | 892177421306343426 | https://pbs.twimg.com/media/DGGmoV4XsAAUL6n.jpg | 1 |
| **2074** | 892420643555336193 | https://pbs.twimg.com/media/DGKD1-bXoAAIAUK.jpg | 1 |

2075 rows × 12 columns

In [29]:

```
image.info()
image.jpg_url.value_counts()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2075 entries, 0 to 2074
Data columns (total 12 columns):
tweet_id    2075 non-null int64
jpg_url     2075 non-null object
img_num     2075 non-null int64
p1          2075 non-null object
p1_conf     2075 non-null float64
p1_dog      2075 non-null bool
p2          2075 non-null object
p2_conf     2075 non-null float64
p2_dog      2075 non-null bool
p3          2075 non-null object
p3_conf     2075 non-null float64
p3_dog      2075 non-null bool
dtypes: bool(3), float64(3), int64(2), object(4)
memory usage: 152.1+ KB
```

Out[29]:

```
https://pbs.twimg.com/media/CtKHLuCWYAA2TTs.jpg
2
https://pbs.twimg.com/media/Cbs3DOAXIAAp3Bd.jpg
2
https://pbs.twimg.com/media/CtVAvX-WIAAcGTf.jpg
2
```

```
https://pbs.twimg.com/media/CcG07BYW0AEfYC9.jpg
2
https://pbs.twimg.com/media/CZhn-QAWwAASQan.jpg
2
https://pbs.twimg.com/media/C12whDoVEAALRxa.jpg
2
https://pbs.twimg.com/media/Cveg1-NXgAASaaT.jpg
2
https://pbs.twimg.com/media/CvyVxQRWEAAdSZS.jpg
2
https://pbs.twimg.com/media/CWyD2HGUYAQ1Xa7.jpg
2
https://pbs.twimg.com/media/C12x-JTVIAAzdfl.jpg
2
https://pbs.twimg.com/media/CUN4Or5UAAAa5K4.jpg
2
https://pbs.twimg.com/media/CwS4aqZXUAAe3IO.jpg
2
https://pbs.twimg.com/media/Cwx99rpW8AMk_Ie.jpg
2
https://pbs.twimg.com/media/DA7iHL5U0AA1OQo.jpg
2
https://pbs.twimg.com/media/CsrjryzWgAAZY00.jpg
2
https://pbs.twimg.com/media/DFDw2tyUQAAAFke.jpg
2
https://pbs.twimg.com/media/CVMOlMiWwAA4Yxl.jpg
2
https://pbs.twimg.com/media/Ck2d7tJWUAEPTL3.jpg
2
https://pbs.twimg.com/media/CW88XN4WsAAlo8r.jpg
2
https://pbs.twimg.com/ext_tw_video_thumb/815965888126062592/pu/img/J
leSw4wRhgKDWQj5.jpg    2
https://pbs.twimg.com/media/C4KHj-nWQAA3poV.jpg
2
https://pbs.twimg.com/media/C2oRbOuWEAAbVSl.jpg
2
https://pbs.twimg.com/media/CeRoBaxWEAABi0X.jpg
2
https://pbs.twimg.com/media/CrXhIqBW8AA6Bse.jpg
2
https://pbs.twimg.com/media/CvoBPWRWgAA4het.jpg
2
https://pbs.twimg.com/media/Cp6db4-XYAAMmqL.jpg
2
https://pbs.twimg.com/media/CwiuEJmW8AAZnit.jpg
2
https://pbs.twimg.com/media/CxqsX-8XUAAEvjD.jpg
2
https://pbs.twimg.com/media/CWza7kpWcAAdYLc.jpg
2
https://pbs.twimg.com/ext_tw_video_thumb/675354114423808004/pu/img/q
L1R_nGLqa6lmkOx.jpg    2
```

```
..
https://pbs.twimg.com/media/CehIzzZWQAEyHH5.jpg
1
https://pbs.twimg.com/media/CZNexghWAAAYnT-.jpg
1
https://pbs.twimg.com/media/C57sMJwXMAASBSx.jpg
1
https://pbs.twimg.com/media/CUeBiqgXAAARLbj.jpg
1
https://pbs.twimg.com/media/CZDRTAPUoAEaqxF.jpg
1
https://pbs.twimg.com/media/C3xq1ZeWEAEuzw3.jpg
1
https://pbs.twimg.com/media/CubGchjXEAA6gpw.jpg
1
https://pbs.twimg.com/media/CU7seitWwAArlVy.jpg
1
https://pbs.twimg.com/media/Crsgi9dWEAApQd8.jpg
1
https://pbs.twimg.com/media/CisqdVcXEAE3iW7.jpg
1
https://pbs.twimg.com/media/CWt-MNIWEAAUC9S.jpg
1
https://pbs.twimg.com/media/CYP62A6WkAAOnL4.jpg
1
https://pbs.twimg.com/media/CmPkGhFXEAABOln.jpg
1
https://pbs.twimg.com/media/CY9xf1dUAAE4XLc.jpg
1
https://pbs.twimg.com/media/CgwuWCeW4AAsgbD.jpg
1
https://pbs.twimg.com/media/Cell8ikWIAACCJ-.jpg
1
https://pbs.twimg.com/media/C09p5dJWIAE5qKL.jpg
1
https://pbs.twimg.com/media/CXKxkseW8AAjAMY.jpg
1
https://pbs.twimg.com/media/CWJmzNsWUAE706Z.jpg
1
https://pbs.twimg.com/media/CWMyl9EWUAAnZJ0.jpg
1
https://pbs.twimg.com/media/CWEWClfW4AAnqhG.jpg
1
https://pbs.twimg.com/media/DAElHfmUMAEH9lB.jpg
1
https://pbs.twimg.com/media/CrCh5RgW8AAXW4U.jpg
1
https://pbs.twimg.com/media/CkTvJTdXAAAEfbT.jpg
1
https://pbs.twimg.com/ext_tw_video_thumb/887517108413886465/pu/img/W
anJKwssZj4VJvL9.jpg    1
https://pbs.twimg.com/media/CWsGnyMVEAAM1Y1.jpg
1
```

```
1

https://pbs.twimg.com/media/CWUA1GFW4AAowiq.jpg
1
https://pbs.twimg.com/media/CUT9PuQWwAABQv7.jpg
1
https://pbs.twimg.com/media/Co5lf-KW8AAIwJw.jpg
1
https://pbs.twimg.com/media/CXKuiyHUEAAMAGa.jpg
1
Name: jpg_url, Length: 2009, dtype: int64
```

In [30]:

```
tweet_df
```

Out[30]:

| | contributors | coordinates | created_at | display_text_range | er |
|---|---|---|---|---|---|
| **1991-02-08 13:48:08.022790149** | NaN | NaN | 2015-11-15 22:32:08 | [0, 131] | {'hashta [], 'medi [{'displa 'pi... |
| **1991-02-08 16:08:05.002620928** | NaN | NaN | 2015-11-15 23:05:30 | [0, 139] | {'hashta [], 'medi [{'displa 'pi... |
| **1991-02-08 17:16:52.701032449** | NaN | NaN | 2015-11-15 23:21:54 | [0, 130] | {'hashta [], 'medi [{'displa 'pi... |
| **1991-02-08 20:17:06.329800704** | NaN | NaN | 2015-11-16 00:04:52 | [0, 137] | {'hashta [], 'medi [{'displa 'pi... |
| **1991-02-08 21:40:48.165822465** | NaN | NaN | 2015-11-16 00:24:50 | [0, 120] | {'hashta [], 'medi [{'displa 'pi... |
| **1991-02-08 22:05:58.794694657** | NaN | NaN | 2015-11-16 00:30:50 | [0, 140] | {'hashta [], 'medi [{'displa 'pi... |
| **1991-02-08 22:24:13.826850816** | | | 2015-11- | | {'hashta [], 'medi |

| | | | | | |
|---|---|---|---|---|---|
| | NaN | NaN | 16 00:35:11 | [0, 138] | [{'displa 'pi... |
| **1991-02-08 23:25:25.042405380** | NaN | NaN | 2015-11-16 00:49:46 | [0, 140] | {'hashta [], 'medi [{'displa 'pi... |
| **1991-02-08 23:51:30.499244032** | NaN | NaN | 2015-11-16 00:55:59 | [0, 124] | {'hashta [], 'medi [{'displa 'pi... |
| **1991-02-09 00:16:40.524156928** | NaN | NaN | 2015-11-16 01:01:59 | [0, 135] | {'hashta [], 'medi [{'displa 'pi... |
| **1991-02-09 01:43:47.256086533** | NaN | NaN | 2015-11-16 01:22:45 | [0, 107] | {'hashta [], 'medi [{'displa 'pi... |
| **1991-02-09 03:46:33.221509120** | NaN | NaN | 2015-11-16 01:52:02 | [0, 137] | {'hashta [], 'medi [{'displa 'pi... |
| **1991-02-09 04:18:20.786774016** | NaN | NaN | 2015-11-16 01:59:36 | [0, 137] | {'hashta [], 'medi [{'displa 'pi... |
| **1991-02-09 07:01:56.733198337** | NaN | NaN | 2015-11-16 02:38:37 | [0, 125] | {'hashta [], 'medi [{'displa 'pi... |
| **1991-02-09 10:06:40.022159362** | NaN | NaN | 2015-11-16 03:22:39 | [0, 132] | {'hashta [], 'medi [{'displa 'pi... |
| **1991-02-09 11:38:33.787052032** | NaN | NaN | 2015-11-16 03:44:34 | [0, 140] | {'hashta [], 'medi [{'displa 'pi... |
| **1991-02-09 12:22:35.909144576** | | | 2015-11-16 | | {'hashta [], 'medi |

| | | | | | |
|---|---|---|---|---|---|
| | NaN | NaN | 03:55:04 | [0, 128] | [{'displa 'pi... |
| **1991-02-09 12:55:33.288665088** | NaN | NaN | 2015-11-16 04:02:55 | [0, 134] | {'hashta [], 'medi [{'displa 'pi... |
| **1991-02-11 10:41:50.803644416** | NaN | NaN | 2015-11-16 14:57:41 | [0, 82] | {'hashta [], 'medi [{'displa 'pi... |
| **1991-02-11 11:51:37.616637952** | NaN | NaN | 2015-11-16 15:14:19 | [0, 46] | {'hashta [], 'medi [{'displa 'pi... |
| **1991-02-11 15:50:06.224695296** | NaN | NaN | 2015-11-16 16:11:11 | [0, 136] | {'hashta [], 'medi [{'displa 'pi... |
| **1991-02-11 17:38:31.632134144** | NaN | NaN | 2015-11-16 16:37:02 | [0, 138] | {'hashta [], 'medi [{'displa 'pi... |
| **1991-02-12 05:51:22.303524864** | NaN | NaN | 2015-11-16 19:31:45 | [0, 139] | {'hashta [], 'medi [{'displa 'pi... |
| **1991-02-12 07:56:57.576210432** | NaN | NaN | 2015-11-16 20:01:42 | [0, 112] | {'hashta [], 'medi [{'displa 'pi... |
| **1991-02-12 10:08:08.456101888** | NaN | NaN | 2015-11-16 20:32:58 | [0, 135] | {'hashta [], 'medi [{'displa 'pi... |
| **1991-02-12 12:45:58.909284353** | NaN | NaN | 2015-11-16 21:10:36 | [0, 140] | {'hashta [], 'medi [{'displa 'pi... |
| **1991-02-12 15:49:13.744588802** | | | 2015-11- | | {'hashta [], 'medi |

| | | | | | |
|---|---|---|---|---|---|
| | NaN | NaN | 16 21:54:18 | [0, 81] | [{'displa 'pi... |
| **1991-02-12 22:04:07.373291520** | NaN | NaN | 2015-11-16 23:23:41 | [0, 137] | {'hashta [], 'medi [{'displa 'pi... |
| **1991-02-13 01:05:26.856765440** | NaN | NaN | 2015-11-17 00:06:54 | [0, 139] | {'hashta [], 'medi [{'displa 'pi... |
| **1991-02-13 02:18:27.551481857** | NaN | NaN | 2015-11-17 00:24:19 | [0, 140] | {'hashta [], 'medi [{'displa 'pi... |
| **...** | ... | ... | ... | ... | ... |
| **1998-01-31 17:16:49.285017600** | NaN | NaN | 2017-07-15 16:51:35 | [27, 105] | {'hashta [], 'symb [], 'urls': 'u... |
| **1998-02-01 20:49:04.734445568** | NaN | NaN | 2017-07-15 23:25:31 | [0, 131] | {'hashta [], 'medi [{'displa 'pi... |
| **1998-02-05 12:05:36.477933568** | NaN | NaN | 2017-07-16 20:14:00 | [0, 71] | {'hashta [], 'medi [{'displa 'pi... |
| **1998-02-06 03:48:00.519319552** | NaN | NaN | 2017-07-16 23:58:41 | [0, 121] | {'hashta [], 'medi [{'displa 'pi... |
| **1998-02-09 00:13:53.522544640** | NaN | NaN | 2017-07-17 16:17:36 | [0, 101] | {'hashta [], 'medi [{'displa 'pi... |
| **1998-02-10 09:03:12.804085760** | NaN | NaN | 2017-07-18 00:07:08 | [0, 129] | {'hashta [], 'medi [{'displa 'pi... |
| **1998-02-13** | | | | | {'hashta |

| | | | | | |
|---|---|---|---|---|---|
| **04:13:37.045368832** | NaN | NaN | 2017-07-18 16:08:03 | [0, 88] | [], 'medi [{'displa 'pi... |
| **1998-02-14 16:32:37.103951883** | NaN | NaN | 2017-07-19 00:47:34 | [0, 99] | {'hashta [], 'medi [{'displa 'pi... |
| **1998-02-15 04:32:19.158093824** | NaN | NaN | 2017-07-19 03:39:09 | [0, 108] | {'hashta [], 'medi [{'displa 'pi... |
| **1998-02-17 08:48:09.381826560** | NaN | NaN | 2017-07-19 16:06:48 | [0, 127] | {'hashta [], 'medi [{'displa 'pi... |
| **1998-02-21 16:27:14.458587136** | NaN | NaN | 2017-07-20 16:49:33 | [0, 127] | {'hashta [], 'medi [{'displa 'pi... |
| **1998-02-27 04:49:22.724278272** | NaN | NaN | 2017-07-22 00:23:06 | [0, 87] | {'hashta [], 'medi [{'displa 'pi... |
| **1998-03-02 02:16:29.199671297** | NaN | NaN | 2017-07-22 16:56:37 | [0, 128] | {'hashta [], 'medi [{'displa 'pi... |
| **1998-03-03 09:27:18.123831296** | NaN | NaN | 2017-07-23 00:22:39 | [0, 86] | {'hashta [], 'medi [{'displa 'pi... |
| **1998-03-07 13:54:01.981685760** | NaN | NaN | 2017-07-24 00:19:32 | [0, 138] | {'hashta [{'indice [129, 13 'text': ... |
| **1998-03-10 11:58:55.344209921** | NaN | NaN | 2017-07-24 17:02:04 | [0, 118] | {'hashta [{'indice [109, 11 'text': ... |
| **1998-03-11** | | | | | {'hashta |

| | | | | | |
|---|---|---|---|---|---|
| **17:53:57.579907072** | NaN | NaN | 2017-07-25 00:10:02 | [0, 91] | [], 'medi [{'displa 'pi... |
| **1998-03-12 01:16:28.333682689** | NaN | NaN | 2017-07-25 01:55:32 | [0, 106] | {'hashta [], 'medi [{'displa 'pi... |
| **1998-03-14 13:08:16.479866881** | NaN | NaN | 2017-07-25 16:11:53 | [0, 107] | {'hashta [], 'medi [{'displa 'pi... |
| **1998-03-16 00:03:28.113172480** | NaN | NaN | 2017-07-26 00:31:25 | [0, 130] | {'hashta [{'indice [121, 13 'text': ... |
| **1998-03-18 16:57:35.349198849** | NaN | NaN | 2017-07-26 15:59:51 | [0, 133] | {'hashta [], 'medi [{'displa 'pi... |
| **1998-03-22 23:26:25.150312448** | NaN | NaN | 2017-07-27 16:25:51 | [0, 122] | {'hashta [{'indice [113, 12 'text': ... |
| **1998-03-24 08:46:21.411237888** | NaN | NaN | 2017-07-28 00:22:40 | [0, 118] | {'hashta [], 'medi [{'displa 'pi... |
| **1998-03-27 04:11:53.173991426** | NaN | NaN | 2017-07-28 16:27:12 | [0, 140] | {'hashta [], 'medi [{'displa 'pi... |
| **1998-03-28 12:25:50.875897856** | NaN | NaN | 2017-07-29 00:08:17 | [0, 138] | {'hashta [{'indice [129, 13 'text': ... |
| **1998-03-31 06:59:18.926688256** | NaN | NaN | 2017-07-29 16:00:24 | [0, 138] | {'hashta [{'indice [129, 13 'text': ... |
| **1998-04-04** | | | 2017-07- | | {'hashta |

| | | | | | |
|---|---|---|---|---|---|
| **11:32:37.279858688** | NaN | NaN | 30 15:58:51 | [0, 79] | [], 'medi [{'displa 'pi... |
| **1998-04-05 22:26:21.378084864** | NaN | NaN | 2017-07-31 00:18:03 | [0, 121] | {'hashta [], 'medi [{'displa 'pi... |
| **1998-04-10 03:03:41.306343426** | NaN | NaN | 2017-08-01 00:17:27 | [0, 138] | {'hashta [], 'medi [{'displa 'pi... |
| **1998-04-12 22:37:23.555336193** | NaN | NaN | 2017-08-01 16:23:56 | [0, 85] | {'hashta [], 'medi [{'displa 'pi... |

2343 rows × 32 columns

In [31]:

```
tweet_df.info()
tweet_df.describe()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 2343 entries, 1991-02-08 13:48:08.022790149 to 1998-0
4-12 22:37:23.555336193
Data columns (total 32 columns):
contributors                     0 non-null float64
coordinates                      0 non-null float64
created_at                       2343 non-null datetime64[ns]
display_text_range               2343 non-null object
entities                         2343 non-null object
extended_entities                2068 non-null object
favorite_count                   2343 non-null int64
favorited                        2343 non-null int64
full_text                        2343 non-null object
geo                              0 non-null float64
id                               2343 non-null int64
id_str                           2343 non-null int64
in_reply_to_screen_name          78 non-null object
in_reply_to_status_id            78 non-null float64
in_reply_to_status_id_str        78 non-null float64
in_reply_to_user_id              78 non-null float64
in_reply_to_user_id_str          78 non-null float64
is_quote_status                  2343 non-null int64
lang                             2343 non-null object
place                            1 non-null object
possibly_sensitive               2206 non-null float64
possibly_sensitive_appealable    2206 non-null float64
quoted_status                    24 non-null object
quoted_status_id                 26 non-null float64
quoted_status_id_str             26 non-null float64
quoted_status_permalink          26 non-null object
retweet_count                    2343 non-null int64
retweeted                        2343 non-null int64
retweeted_status                 169 non-null object
source                           2343 non-null object
truncated                        2343 non-null int64
user                             2343 non-null object
dtypes: datetime64[ns](1), float64(11), int64(8), object(12)
memory usage: 604.1+ KB
```

```
Out[31]:
```

| | contributors | coordinates | favorite_count | favorited | geo | id | |
|---|---|---|---|---|---|---|---|
| **count** | 0.0 | 0.0 | 2343.000000 | 2343.0 | 0.0 | 2.343000e+03 | 2.3430 |
| **mean** | NaN | NaN | 8045.846778 | 0.0 | NaN | 7.422769e+17 | 7.4227 |
| **std** | NaN | NaN | 12170.706476 | 0.0 | NaN | 6.836264e+16 | 6.8362 |
| **min** | NaN | NaN | 0.000000 | 0.0 | NaN | 6.660209e+17 | 6.6602 |
| **25%** | NaN | NaN | 1401.500000 | 0.0 | NaN | 6.783607e+17 | 6.7836 |
| **50%** | NaN | NaN | 3523.000000 | 0.0 | NaN | 7.186315e+17 | 7.1863 |
| **75%** | NaN | NaN | 9924.000000 | 0.0 | NaN | 7.986999e+17 | 7.9869 |
| **max** | NaN | NaN | 142677.000000 | 0.0 | NaN | 8.924206e+17 | 8.9242 |

```
In [32]:
```

```
list(tweet_df)
```

Out[32]:

```
['contributors',
 'coordinates',
 'created_at',
 'display_text_range',
 'entities',
 'extended_entities',
 'favorite_count',
 'favorited',
 'full_text',
 'geo',
 'id',
 'id_str',
 'in_reply_to_screen_name',
 'in_reply_to_status_id',
 'in_reply_to_status_id_str',
 'in_reply_to_user_id',
 'in_reply_to_user_id_str',
 'is_quote_status',
 'lang',
 'place',
 'possibly_sensitive',
 'possibly_sensitive_appealable',
 'quoted_status',
 'quoted_status_id',
 'quoted_status_id_str',
 'quoted_status_permalink',
 'retweet_count',
 'retweeted',
 'retweeted_status',
 'source',
 'truncated',
 'user']
```

In [33]:

```
# randomly visualize the info in user column of tweet_df table
random.choice(tweet_df.user.tolist())
```

Out[33]:

```
{'contributors_enabled': False,
 'created_at': 'Sun Nov 15 21:41:29 +0000 2015',
 'default_profile': False,
 'default_profile_image': False,
 'description': 'Your Only Source for Pawfessional Dog Ratings STORE
: @ShopWeRateDogs | IG, FB & SC: WeRateDogs | MOBILE APP: @GoodDogsG
ame Business: dogratingtwitter@gmail.com',
 'entities': {'description': {'urls': []},
  'url': {'urls': [{'display_url': 'weratedogs.com',
     'expanded_url': 'http://weratedogs.com',
     'indices': [0, 23],
     'url': 'https://t.co/N7sNNHSfPg'}]}},
     'url': 'https://t.co/N7sNNHSfPq'}]}},
 'favourites_count': 135451,
 'follow_request_sent': False,
 'followers_count': 7070900,
 'following': False,
 'friends_count': 9,
 'geo_enabled': True,
 'has_extended_profile': True,
 'id': 4196983835,
 'id_str': '4196983835',
 'is_translation_enabled': False,
 'is_translator': False,
 'lang': 'en',
 'listed_count': 4718,
 'location': '𝓶𝓮𝓻𝓬𝓱 ↴      DM YOUR DOGS',
 'name': 'WeRateDogs™',
 'notifications': False,
 'profile_background_color': '000000',
 'profile_background_image_url': 'http://abs.twimg.com/images/themes
/theme1/bg.png',
 'profile_background_image_url_https': 'https://abs.twimg.com/images
/themes/theme1/bg.png',
 'profile_background_tile': False,
 'profile_banner_url': 'https://pbs.twimg.com/profile_banners/419698
3835/1525830435',
 'profile_image_url': 'http://pbs.twimg.com/profile_images/948761950
363664385/Fpr2Oz35_normal.jpg',
 'profile_image_url_https': 'https://pbs.twimg.com/profile_images/94
8761950363664385/Fpr2Oz35_normal.jpg',
 'profile_link_color': 'F5ABB5',
 'profile_sidebar_border_color': '000000',
 'profile_sidebar_fill_color': '000000',
 'profile_text_color': '000000',
 'profile_use_background_image': False,
 'protected': False,
 'screen_name': 'dog_rates',
 'statuses_count': 7352,
 'time_zone': None,
 'translator_type': 'none',
 'url': 'https://t.co/N7sNNHSfPq',
 'utc_offset': None,
 'verified': True}
```

In [34]:

```
all_columns = pd.Series(list(archive) + list(image) + list(tweet_df))
all_columns[all_columns.duplicated()]
```

Out[34]:

```
17                 tweet_id
42    in_reply_to_status_id
44      in_reply_to_user_id
58                   source
dtype: object
```

**Quality**

***`archive`* table**

- We only want original ratings (no retweets) that have images;
- Timestamp column should be in datetime format;
- Name column consists of many invalid values i.e 'just, 'None', 'a', 'an', 'all';
- The NA value in name colunm is not in accurate data format;
- Extract gender info from 'source' column and convert the values of 'None' into programmable NA values;
- Parse the datetime column into separate columns;
- Extract the the rating numerator from 'text' column beucase some values in 'rating_numerator' column is wrong;
- Convert the data type in both 'rating_numerator' and 'rating_denominator' columns as float;
- Convert 'None' in 'stage' column to programmable NA values.

***`image`* table**

- Different tweet_ids have the same jpg_url;
- Simplify the table by keeping only one prediction, according to the odds priority order is as p1 > p2 > p3;
- Convert data type in 'tweet_id' column from a integer to string.

***`tweet_df`* table**

- Extract followers_count and favourites_count values from 'user' column;
- Rename the 'id' column to "tweet_id" to match the other 2 tables;
- Reset the chaotic index by sequential order;
- Convert data type in 'id' column from integer to string.

### Tidiness

#### *`archive` table*

- Melt dog stage column into a single column;
- Values in tweet_id column need to be converted from integer to string because there would not be any mathmatic operations on them;
- Keep only the necessary columns for analysis, 'tweet_id', 'time_stamp', 'rating_numerator', 'rating_denominator', 'name', 'date', 'time', 'stage'.

#### *`image` table*

- Keep only the necessary columns, such as 'tweet_id', 'jpg_url', 'img_num', 'predictions', 'odds'.

#### *`tweet_df` table*

- Remove the columns we don't need

#### *'all the 3 tables'*

- Consolidate the 3 tables.

## Cleaning data

This is the third step of data wrangling, according to the points denoted above quality and tidiness issues will be fixed.

In [92]:

```
# backup the dataset
archive_clean = archive.copy()
image_clean = image.copy()
tweet_df_clean = tweet_df.copy()
```

### `archive_clean` table

#### *Define*

Keep only original ratings (no retweets) that have images by removing rows of which the values in 'retweeted_status_id' column is not null.

#### *Code*

In [93]:

```
# remove rows of which the values in 'retweeted_status_id' column is not null
archive_clean.drop(archive_clean[archive_clean.retweeted_status_id == \
                                 archive_clean.retweeted_status_id].index, inpla
ce=True)
```

***Test***

In [94]:

```
# randomly print out 5 rows for visualization
# the retweets should be removed, thus the column contains only NA values
archive_clean.query('retweeted_status_id != retweeted_status_id').sample(5)
```

```
Out[94]:
```

| | tweet_id | in_reply_to_status_id | in_reply_to_user_id | timestamp | |
|---|---|---|---|---|---|
| **401** | 824663926340194305 | NaN | NaN | 2017-01-26 17:02:56 +0000 | <a href r... |
| **2143** | 669970042633789440 | NaN | NaN | 2015-11-26 20:04:40 +0000 | <a href r... |
| **972** | 750086836815486976 | NaN | NaN | 2016-07-04 22:00:12 +0000 | <a href |
| **1098** | 736010884653420544 | NaN | NaN | 2016-05-27 01:47:23 +0000 | <a href r... |
| **2333** | 666337882303524864 | NaN | NaN | 2015-11-16 19:31:45 +0000 | <a href r... |

***Define***

Convert values in timestamp column to datetime format.

***Code***

In [95]:

```
archive_clean.timestamp = pd.to_datetime(archive_clean.timestamp)
```

***Test***

In [96]:

```
archive_clean.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2175 entries, 0 to 2355
Data columns (total 17 columns):
tweet_id                      2175 non-null int64
in_reply_to_status_id         78 non-null float64
in_reply_to_user_id           78 non-null float64
timestamp                     2175 non-null datetime64[ns]
source                        2175 non-null object
text                          2175 non-null object
retweeted_status_id           0 non-null float64
retweeted_status_user_id      0 non-null float64
retweeted_status_timestamp    0 non-null object
expanded_urls                 2117 non-null object
rating_numerator              2175 non-null int64
rating_denominator            2175 non-null int64
name                          2175 non-null object
doggo                         2175 non-null object
floofer                       2175 non-null object
pupper                        2175 non-null object
puppo                         2175 non-null object
dtypes: datetime64[ns](1), float64(4), int64(3), object(9)
memory usage: 305.9+ KB
```

***Define***

Correct invalid values in name column.

***Code***

In [97]:

```
# find the error names again as some of them might be taken away alone with the
action of removing retweets
archive_clean.name.unique()
```

Out[97]:

```
array(['Phineas', 'Tilly', 'Archie', 'Darla', 'Franklin', 'None', 'J
ax',
       'Zoey', 'Cassie', 'Koda', 'Bruno', 'Ted', 'Stuart', 'Oliver',
'Jim',
 'Jim',
       'Zeke', 'Ralphus', 'Gerald', 'Jeffrey', 'such', 'Canela', 'Ma
ya',
       'Mingus', 'Derek', 'Roscoe', 'Waffles', 'Jimbo', 'Maisey', 'E
arl',
       'Lola', 'Kevin', 'Yogi', 'Noah', 'Bella', 'Grizzwald', 'Rusty
',
       'Gus', 'Stanley', 'Alfy', 'Koko', 'Rey', 'Gary', 'a', 'Elliot
',
       'Louis', 'Jesse', 'Romeo', 'Bailey', 'Duddles', 'Jack', 'Stev
en',
       'Beau', 'Snoopy', 'Shadow', 'Emmy', 'Aja', 'Penny', 'Dante',
       'Nelly', 'Ginger', 'Benedict', 'Venti', 'Goose', 'Nugget', 'C
ash',
       'Jed', 'Sebastian', 'Sierra', 'Monkey', 'Harry', 'Kody', 'Las
sie',
       'Rover', 'Napolean', 'Boomer', 'Cody', 'Rumble', 'Clifford',
       'Dewey', 'Scout', 'Gizmo', 'Walter', 'Cooper', 'Harold', 'Shi
kha',
       'Lili', 'Jamesy', 'Coco', 'Sammy', 'Meatball', 'Paisley', 'Al
bus',
       'Neptune', 'Belle', 'Quinn', 'Zooey', 'Dave', 'Jersey', 'Hobb
es',
       'Burt', 'Lorenzo', 'Carl', 'Jordy', 'Milky', 'Trooper', 'quit
e',
       'Sophie', 'Wyatt', 'Rosie', 'Thor', 'Oscar', 'Callie', 'Cerme
t',
       'Marlee', 'Arya', 'Einstein', 'Alice', 'Rumpole', 'Benny', 'A
spen',
       'Jarod', 'Wiggles', 'General', 'Sailor', 'Iggy', 'Snoop', 'Ky
le',
       'Leo', 'Riley', 'Noosh', 'Odin', 'Jerry', 'Georgie', 'Rontu',
       'Cannon', 'Furzey', 'Daisy', 'Tuck', 'Barney', 'Vixen', 'Jarv
is',
       'Mimosa', 'Pickles', 'Brady', 'Luna', 'Charlie', 'Margo', 'Sa
die',
       'Hank', 'Tycho', 'Indie', 'Winnie', 'George', 'Bentley', 'Max
',
       'Dawn', 'Maddie', 'Monty', 'Sojourner', 'Winston', 'Odie', 'A
rlo',
       'Vincent', 'Lucy', 'Clark', 'Mookie', 'Meera', 'Ava', 'Eli',
'Ash',
       'Tucker', 'Tobi', 'Chester', 'Wilson', 'Sunshine', 'Lipton',
       'Bronte', 'Poppy', 'Gidget', 'Rhino', 'Willow', 'not', 'Orion
',
       'Eevee', 'Smiley', 'Miguel', 'Emanuel', 'Kuyu', 'Dutch', 'Pet
e',
       'Scooter', 'Reggie', 'Lilly', 'Samson', 'Mia', 'Astrid', 'Mal
colm',
       'Dexter', 'Alfie', 'Fiona', 'one', 'Mutt', 'Bear', 'Doobert',
       'Beebop', 'Alexander', 'Sailer', 'Brutus', 'Kona', 'Boots',
       'Ralphie', 'Loki', 'Cupid', 'Pawnd', 'Pilot', 'Ike', 'Mo', 'T
oby',
       'Sweet', 'Pablo', 'Nala', 'Crawford', 'Gabe', 'Jimison', 'Duc
```

```
hess',
       'Harlso', 'Sundance', 'Luca', 'Flash', 'Sunny', 'Howie', 'Jaz
zy',
       'Anna', 'Finn', 'Bo', 'Wafer', 'Tom', 'Florence', 'Autumn', '
Buddy',
       'Dido', 'Eugene', 'Ken', 'Strudel', 'Tebow', 'Chloe', 'Timber
',
       'Binky', 'Moose', 'Dudley', 'Comet', 'Akumi', 'Titan', 'Olivi
a',
       'Alf', 'Oshie', 'Chubbs', 'Sky', 'Atlas', 'Eleanor', 'Layla',
       'Rocky', 'Baron', 'Tyr', 'Bauer', 'Swagger', 'Brandi', 'Mary'
,
       'Moe', 'Halo', 'Augie', 'Craig', 'Sam', 'Hunter', 'Pavlov', '
Phil',
       'Kyro', 'Wallace', 'Ito', 'Seamus', 'Ollie', 'Stephan', 'Lenn
on',
       'incredibly', 'Major', 'Duke', 'Sansa', 'Shooter', 'Django',
       'Diogi', 'Sonny', 'Marley', 'Severus', 'Ronnie', 'Milo', 'Bon
es',
       'Mauve', 'Chef', 'Doc', 'Peaches', 'Sobe', 'Longfellow', 'Mis
ter',
       'Iroh', 'Pancake', 'Snicku', 'Ruby', 'Brody', 'Mack', 'Nimbus
',
       'Laika', 'Maximus', 'Dobby', 'Moreton', 'Juno', 'Maude', 'Lil
y',
       'Newt', 'Benji', 'Nida', 'Robin', 'Monster', 'BeBe', 'Remus',
       'Levi', 'Mabel', 'Misty', 'Betty', 'Mosby', 'Maggie', 'Bruce'
,
       'Happy', 'Ralphy', 'Brownie', 'Rizzy', 'Stella', 'Butter', 'F
rank',
       'Tonks', 'Lincoln', 'Rory', 'Logan', 'Dale', 'Rizzo', 'Arnie'
,
       'Mattie', 'Pinot', 'Dallas', 'Hero', 'Frankie', 'Stormy',
       'Reginald', 'Balto', 'Mairi', 'Loomis', 'Godi', 'Cali', 'Deac
on',
       'Timmy', 'Sampson', 'Chipson', 'Combo', 'Oakley', 'Dash',
       'Hercules', 'Jay', 'Mya', 'Strider', 'Wesley', 'Solomon', 'Hu
ck',
       'O', 'Blue', 'Anakin', 'Finley', 'Sprinkles', 'Heinrich',
       'Shakespeare', 'Chelsea', 'Bungalo', 'Chip', 'Grey', 'Rooseve
lt',
       'Willem', 'Davey', 'Dakota', 'Fizz', 'Dixie', 'very', 'Al',
       'Jackson', 'Carbon', 'Klein', 'DonDon', 'Kirby', 'Lou', 'Chev
y',
       'Tito', 'Philbert', 'Louie', 'Rupert', 'Rufus', 'Brudge', 'Sh
adoe',
       'Angel', 'Brat', 'Tove', 'my', 'Gromit', 'Aubie', 'Kota', 'Le
ela',
       'Glenn', 'Shelby', 'Sephie', 'Bonaparte', 'Albert', 'Wishes',
       'Rose', 'Theo', 'Rocco', 'Fido', 'Emma', 'Spencer', 'Lilli',
       'Boston', 'Brandonald', 'Corey', 'Leonard', 'Beckham', 'Devón
',
       'Gert', 'Watson', 'Keith', 'Dex', 'Ace', 'Tayzie', 'Grizzie',
        'Fred', 'Gilbert', 'Meyer', 'Zoe', 'Stewie', 'Calvin', 'Lilah
',
        'Spanky', 'Jameson', 'Piper', 'Atticus', 'Blu', 'Dietrich',
        'Divine', 'Tripp', 'his', 'Cora', 'Huxley', 'Keurig', 'Bookst
ore',
        'Linus', 'Abby', 'Shiloh', 'an', 'Gustav', 'Arlen', 'Percy',
        'Lenox', 'Sugar', 'Harvey', 'Blanket', 'actually', 'Geno', 'S
tark',
        'Beya', 'Kilo', 'Kayla', 'Maxaroni', 'Bell', 'Doug', 'Edmund'
,
        'Aqua', 'Theodore', 'just', 'Baloo', 'Chase', 'getting', 'Nol
lie',
        'Rorie', 'Simba', 'Charles', 'Bayley', 'Axel', 'Storkson', 'R
emy',
        'Chadrick', 'mad', 'Kellogg', 'Buckley', 'Livvie', 'Terry',
        'Hermione', 'Ralpher', 'Aldrick', 'Larry', 'this', 'unaccepta
ble',
        'Rooney', 'Crystal', 'Ziva', 'Stefan', 'Pupcasso', 'Puff',
        'Flurpson', 'Coleman', 'Enchilada', 'Raymond', 'all', 'Rueben
',
        'Cilantro', 'Karll', 'Sprout', 'Blitz', 'Bloop', 'Colby', 'Li
llie',
        'Ashleigh', 'Kreggory', 'Sarge', 'Luther', 'Ivar', 'Jangle',
        'Schnitzel', 'Panda', 'Berkeley', 'Ralphé', 'Charleson', 'Cly
de',
        'Harnold', 'Sid', 'Pippa', 'Otis', 'Carper', 'Bowie',
        'Alexanderson', 'Suki', 'Barclay', 'Skittle', 'Ebby', 'Flávio
',
        'Smokey', 'Link', 'Jennifur', 'Ozzy', 'Bluebert', 'Stephanus'
,
        'Bubbles', 'old', 'Zeus', 'Bertson', 'Nico', 'Michelangelope'
,
        'Siba', 'Calbert', 'Curtis', 'Travis', 'Thumas', 'Kanu', 'Lan
ce',
        'Opie', 'Stubert', 'Kane', 'Olive', 'Chuckles', 'Staniel', 'S
ora',
        'Beemo', 'Gunner', 'infuriating', 'Lacy', 'Tater', 'Olaf', 'C
ecil',
        'Vince', 'Karma', 'Billy', 'Walker', 'Rodney', 'Klevin', 'Mal
ikai',
        'Bobble', 'River', 'Jebberson', 'Remington', 'Farfle', 'Jimin
us',
        'Harper', 'Clarkus', 'Finnegus', 'Cupcake', 'Kathmandu', 'Ell
ie',
        'Katie', 'Kara', 'Adele', 'Zara', 'Ambrose', 'Jimothy', 'Bode
',
        'Terrenth', 'Reese', 'Chesterson', 'Lucia', 'Bisquick', 'Ralp
hson',
        'Socks', 'Rambo', 'Rudy', 'Fiji', 'Rilo', 'Bilbo', 'Coopson',
        'Yoda', 'Millie', 'Chet', 'Crouton', 'Daniel', 'Kaia', 'Murph
y',
        'Dotsy', 'Eazy', 'Coops', 'Fillup', 'Miley', 'Charl', 'Reagan
',
        'Yukon', 'CeCe', 'Cuddles', 'Claude', 'Jessiga', 'Carter', 'O
le',
        'Pherb', 'Blipson', 'Reptar', 'Trevith', 'Berb', 'Bob', 'Coli
n',
        'Brian', 'Oliviér', 'Grady', 'Kobe', 'Freddery', 'Bodie', 'Du
nkin',
        'Wally', 'Tupawc', 'Amber', 'Herschel', 'Edgar', 'Teddy',
        'Kingsley', 'Brockly', 'Richie', 'Molly', 'Vinscent', 'Cedric
k',
        'Hazel', 'Lolo', 'Eriq', 'Phred', 'the', 'Oddie', 'Maxwell',
        'Geoff', 'Covach', 'Durg', 'Fynn', 'Ricky', 'Herald', 'Lucky'
,
        'Ferg', 'Trip', 'Clarence', 'Hamrick', 'Brad', 'Pubert', 'Frö
nq',
        'Derby', 'Lizzie', 'Ember', 'Blakely', 'Opal', 'Marq', 'Krame
r',
        'Barry', 'Tyrone', 'Gordon', 'Baxter', 'Mona', 'Horace', 'Cri
mson',
        'Birf', 'Hammond', 'Lorelei', 'Marty', 'Brooks', 'Petrick',
        'Hubertson', 'Gerbald', 'Oreo', 'Bruiser', 'Perry', 'Bobby',
'Jeph',
        'Obi', 'Tino', 'Kulet', 'Sweets', 'Lupe', 'Tiger', 'Jiminy',
        'Griffin', 'Banjo', 'Brandy', 'Lulu', 'Darrel', 'Taco', 'Joey
',
        'Patrick', 'Kreg', 'Todo', 'Tess', 'Ulysses', 'Toffee', 'Apol
lo',
        'Carly', 'Asher', 'Glacier', 'Chuck', 'Champ', 'Ozzie', 'Gris
wold',
        'Cheesy', 'Moofasa', 'Hector', 'Goliath', 'Kawhi', 'by', 'Emm
ie',
        'Penelope', 'Willie', 'Rinna', 'Mike', 'William', 'Dwight', '
Evy',
        'Hurley', 'Rubio', 'officially', 'Chompsky', 'Rascal', 'Linda
',
        'Tug', 'Tango', 'Grizz', 'Jerome', 'Crumpet', 'Jessifer', 'Iz
zy',
        'Ralph', 'Sandy', 'Humphrey', 'Tassy', 'Juckson', 'Chuq', 'Ty
rus',
        'Karl', 'Godzilla', 'Vinnie', 'Kenneth', 'Herm', 'Bert', 'Str
iker',
        'Donny', 'Pepper', 'Bernie', 'Buddah', 'Lenny', 'Arnold', 'Zu
zu',
        'Mollie', 'Laela', 'Tedders', 'Superpup', 'Rufio', 'Jeb', 'Ro
dman',
        'Jonah', 'Chesney', 'life', 'Kenny', 'Henry', 'Bobbay', 'Mitc
h',
        'Kaiya', 'Acro', 'Aiden', 'Obie', 'Dot', 'Shnuggles', 'Kendal
l',
        'Jeffri', 'Steve', 'Eve', 'Mac', 'Fletcher', 'Kenzie', 'Pumpk
in',
        'Schnozz', 'Gustaf', 'Cheryl', 'Ed', 'Leonidas', 'Norman', 'C
aryl',
        'Scott', 'Taz', 'Darby', 'Jackie', 'light', 'Jazz', 'Franq',
       'Pippin', 'Rolf', 'Snickers', 'Ridley', 'Cal', 'Bradley', 'Bu
bba',
       'Tuco', 'Patch', 'Mojo', 'Batdog', 'Dylan', 'space', 'Mark',
'JD',
       'Alejandro', 'Scruffers', 'Pip', 'Julius', 'Tanner', 'Sparky'
,
       'Anthony', 'Holly', 'Jett', 'Amy', 'Sage', 'Andy', 'Mason',
       'Trigger', 'Antony', 'Creg', 'Traviss', 'Gin', 'Jeffrie', 'Da
nny',
       'Ester', 'Pluto', 'Bloo', 'Edd', 'Paull', 'Willy', 'Herb', 'D
amon',
       'Peanut', 'Nigel', 'Butters', 'Sandra', 'Fabio', 'Randall', '
Liam',
       'Tommy', 'Ben', 'Raphael', 'Julio', 'Andru', 'Kloey', 'Shawwn
',
       'Skye', 'Kollin', 'Ronduh', 'Billl', 'Saydee', 'Dug', 'Tessa'
,
       'Sully', 'Kirk', 'Ralf', 'Clarq', 'Jaspers', 'Samsom', 'Terra
nce',
       'Harrison', 'Chaz', 'Jeremy', 'Jaycob', 'Lambeau', 'Ruffles',
       'Amélie', 'Bobb', 'Banditt', 'Kevon', 'Winifred', 'Hanz', 'Ch
urlie',
       'Zeek', 'Timofy', 'Maks', 'Jomathan', 'Kallie', 'Marvin', 'Sp
ark',
       'Gòrdón', 'Jo', 'DayZ', 'Jareld', 'Torque', 'Ron', 'Skittles'
,
       'Cleopatricia', 'Erik', 'Stu', 'Tedrick', 'Shaggy', 'Filup',
'Kial',
       'Naphaniel', 'Dook', 'Hall', 'Philippe', 'Biden', 'Fwed',
       'Genevieve', 'Joshwa', 'Timison', 'Bradlay', 'Pipsy', 'Clybe'
,
       'Keet', 'Carll', 'Jockson', 'Josep', 'Lugan', 'Christoper'],
dtype=object)
```

In [98]:

```python
for name in archive_clean.name:
    if name.islower():
        archive_clean.name.replace(name, 'None', inplace = True)
archive_clean.name = archive_clean.name.replace('O', 'None')
```

***Test***

In [99]:

```python
archive_clean.name.unique()
```

Out[99]:

```
array(['Phineas', 'Tilly', 'Archie', 'Darla', 'Franklin', 'None', 'J
ax',
       'Zoey', 'Cassie', 'Koda', 'Bruno', 'Ted', 'Stuart', 'Oliver',
'Jim',
       'Zeke', 'Ralphus', 'Gerald', 'Jeffrey', 'Canela', 'Maya', 'Mi
ngus',
       'Derek', 'Roscoe', 'Waffles', 'Jimbo', 'Maisey', 'Earl', 'Lol
a',
       'Kevin', 'Yogi', 'Noah', 'Bella', 'Grizzwald', 'Rusty', 'Gus'
,
       'Stanley', 'Alfy', 'Koko', 'Rey', 'Gary', 'Elliot', 'Louis',
       'Jesse', 'Romeo', 'Bailey', 'Duddles', 'Jack', 'Steven', 'Bea
u',
       'Snoopy', 'Shadow', 'Emmy', 'Aja', 'Penny', 'Dante', 'Nelly',
       'Ginger', 'Benedict', 'Venti', 'Goose', 'Nugget', 'Cash', 'Je
d',
       'Sebastian', 'Sierra', 'Monkey', 'Harry', 'Kody', 'Lassie', '
Rover',
       'Napolean', 'Boomer', 'Cody', 'Rumble', 'Clifford', 'Dewey',
       'Scout', 'Gizmo', 'Walter', 'Cooper', 'Harold', 'Shikha', 'Li
li',
       'Jamesy', 'Coco', 'Sammy', 'Meatball', 'Paisley', 'Albus',
       'Neptune', 'Belle', 'Quinn', 'Zooey', 'Dave', 'Jersey', 'Hobb
es',
       'Burt', 'Lorenzo', 'Carl', 'Jordy', 'Milky', 'Trooper', 'Soph
ie',
       'Wyatt', 'Rosie', 'Thor', 'Oscar', 'Callie', 'Cermet', 'Marle
e',
       'Arya', 'Einstein', 'Alice', 'Rumpole', 'Benny', 'Aspen', 'Ja
rod',
       'Wiggles', 'General', 'Sailor', 'Iggy', 'Snoop', 'Kyle', 'Leo
',
       'Riley', 'Noosh', 'Odin', 'Jerry', 'Georgie', 'Rontu', 'Canno
n',
       'Furzey', 'Daisy', 'Tuck', 'Barney', 'Vixen', 'Jarvis', 'Mimo
sa',
       'Pickles', 'Brady', 'Luna', 'Charlie', 'Margo', 'Sadie', 'Han
k',
       'Tycho', 'Indie', 'Winnie', 'George', 'Bentley', 'Max', 'Dawn
',
       'Maddie', 'Monty', 'Sojourner', 'Winston', 'Odie', 'Arlo',
       'Vincent', 'Lucy', 'Clark', 'Mookie', 'Meera', 'Ava', 'Eli',
'Ash',
       'Tucker', 'Tobi', 'Chester', 'Wilson', 'Sunshine', 'Lipton',
       'Bronte', 'Poppy', 'Gidget', 'Rhino', 'Willow', 'Orion', 'Eev
ee',
       'Smiley', 'Miguel', 'Emanuel', 'Kuyu', 'Dutch', 'Pete', 'Scoo
ter',
       'Reggie', 'Lilly', 'Samson', 'Mia', 'Astrid', 'Malcolm', 'Dex
ter',
       'Alfie', 'Fiona', 'Mutt', 'Bear', 'Doobert', 'Beebop', 'Alexa
nder',
       'Sailer', 'Brutus', 'Kona', 'Boots', 'Ralphie', 'Loki', 'Cupi
d',
       'Pawnd', 'Pilot', 'Ike', 'Mo', 'Toby', 'Sweet', 'Pablo', 'Nal
a',
       'Crawford', 'Gabe', 'Jimison', 'Duchess', 'Harlso', 'Sundance
```

```
',
       'Luca', 'Flash', 'Sunny', 'Howie', 'Jazzy', 'Anna', 'Finn', '
Bo',
       'Wafer', 'Tom', 'Florence', 'Autumn', 'Buddy', 'Dido', 'Eugen
e',
       'Ken', 'Strudel', 'Tebow', 'Chloe', 'Timber', 'Binky', 'Moose
',
       'Dudley', 'Comet', 'Akumi', 'Titan', 'Olivia', 'Alf', 'Oshie'
,
       'Chubbs', 'Sky', 'Atlas', 'Eleanor', 'Layla', 'Rocky', 'Baron
',
       'Tyr', 'Bauer', 'Swagger', 'Brandi', 'Mary', 'Moe', 'Halo', '
Augie',
       'Craig', 'Sam', 'Hunter', 'Pavlov', 'Phil', 'Kyro', 'Wallace'
,
       'Ito', 'Seamus', 'Ollie', 'Stephan', 'Lennon', 'Major', 'Duke
',
       'Sansa', 'Shooter', 'Django', 'Diogi', 'Sonny', 'Marley', 'Se
verus',
       'Ronnie', 'Milo', 'Bones', 'Mauve', 'Chef', 'Doc', 'Peaches',
       'Sobe', 'Longfellow', 'Mister', 'Iroh', 'Pancake', 'Snicku',
'Ruby',
       'Brody', 'Mack', 'Nimbus', 'Laika', 'Maximus', 'Dobby', 'More
ton',
       'Juno', 'Maude', 'Lily', 'Newt', 'Benji', 'Nida', 'Robin',
       'Monster', 'BeBe', 'Remus', 'Levi', 'Mabel', 'Misty', 'Betty'
,
       'Mosby', 'Maggie', 'Bruce', 'Happy', 'Ralphy', 'Brownie', 'Ri
zzy',
       'Stella', 'Butter', 'Frank', 'Tonks', 'Lincoln', 'Rory', 'Log
an',
       'Dale', 'Rizzo', 'Arnie', 'Mattie', 'Pinot', 'Dallas', 'Hero'
,
       'Frankie', 'Stormy', 'Reginald', 'Balto', 'Mairi', 'Loomis',
'Godi',
       'Cali', 'Deacon', 'Timmy', 'Sampson', 'Chipson', 'Combo', 'Oa
kley',
       'Dash', 'Hercules', 'Jay', 'Mya', 'Strider', 'Wesley', 'Solom
on',
       'Huck', 'Blue', 'Anakin', 'Finley', 'Sprinkles', 'Heinrich',
       'Shakespeare', 'Chelsea', 'Bungalo', 'Chip', 'Grey', 'Rooseve
lt',
       'Willem', 'Davey', 'Dakota', 'Fizz', 'Dixie', 'Al', 'Jackson'
,
       'Carbon', 'Klein', 'DonDon', 'Kirby', 'Lou', 'Chevy', 'Tito',
       'Philbert', 'Louie', 'Rupert', 'Rufus', 'Brudge', 'Shadoe', '
Angel',
       'Brat', 'Tove', 'Gromit', 'Aubie', 'Kota', 'Leela', 'Glenn',
       'Shelby', 'Sephie', 'Bonaparte', 'Albert', 'Wishes', 'Rose',
'Theo',
       'Rocco', 'Fido', 'Emma', 'Spencer', 'Lilli', 'Boston', 'Brand
onald',
       'Corey', 'Leonard', 'Beckham', 'Devón', 'Gert', 'Watson', 'Ke
```

```
ith',
       'Dex', 'Ace', 'Tayzie', 'Grizzie', 'Fred', 'Gilbert', 'Meyer'
,
       'Zoe', 'Stewie', 'Calvin', 'Lilah', 'Spanky', 'Jameson', 'Pip
er',
       'Atticus', 'Blu', 'Dietrich', 'Divine', 'Tripp', 'Cora', 'Hux
ley',
       'Keurig', 'Bookstore', 'Linus', 'Abby', 'Shiloh', 'Gustav', '
Arlen',
       'Percy', 'Lenox', 'Sugar', 'Harvey', 'Blanket', 'Geno', 'Star
k',
       'Beya', 'Kilo', 'Kayla', 'Maxaroni', 'Bell', 'Doug', 'Edmund'
,
       'Aqua', 'Theodore', 'Baloo', 'Chase', 'Nollie', 'Rorie', 'Sim
ba',
       'Charles', 'Bayley', 'Axel', 'Storkson', 'Remy', 'Chadrick',
       'Kellogg', 'Buckley', 'Livvie', 'Terry', 'Hermione', 'Ralpher
',
       'Aldrick', 'Larry', 'Rooney', 'Crystal', 'Ziva', 'Stefan',
       'Pupcasso', 'Puff', 'Flurpson', 'Coleman', 'Enchilada', 'Raym
ond',
       'Rueben', 'Cilantro', 'Karll', 'Sprout', 'Blitz', 'Bloop', 'C
olby',
       'Lillie', 'Ashleigh', 'Kreggory', 'Sarge', 'Luther', 'Ivar',
       'Jangle', 'Schnitzel', 'Panda', 'Berkeley', 'Ralphé', 'Charle
son',
       'Clyde', 'Harnold', 'Sid', 'Pippa', 'Otis', 'Carper', 'Bowie'
,
       'Alexanderson', 'Suki', 'Barclay', 'Skittle', 'Ebby', 'Flávio
',
       'Smokey', 'Link', 'Jennifur', 'Ozzy', 'Bluebert', 'Stephanus'
,
       'Bubbles', 'Zeus', 'Bertson', 'Nico', 'Michelangelope', 'Siba
',
       'Calbert', 'Curtis', 'Travis', 'Thumas', 'Kanu', 'Lance', 'Op
ie',
       'Stubert', 'Kane', 'Olive', 'Chuckles', 'Staniel', 'Sora', 'B
eemo',
       'Gunner', 'Lacy', 'Tater', 'Olaf', 'Cecil', 'Vince', 'Karma',
       'Billy', 'Walker', 'Rodney', 'Klevin', 'Malikai', 'Bobble', '
River',
       'Jebberson', 'Remington', 'Farfle', 'Jiminus', 'Harper', 'Cla
rkus',
       'Finnegus', 'Cupcake', 'Kathmandu', 'Ellie', 'Katie', 'Kara',
       'Adele', 'Zara', 'Ambrose', 'Jimothy', 'Bode', 'Terrenth', 'R
eese',
       'Chesterson', 'Lucia', 'Bisquick', 'Ralphson', 'Socks', 'Ramb
o',
       'Rudy', 'Fiji', 'Rilo', 'Bilbo', 'Coopson', 'Yoda', 'Millie',
       'Chet', 'Crouton', 'Daniel', 'Kaia', 'Murphy', 'Dotsy', 'Eazy
',
       'Coops', 'Fillup', 'Miley', 'Charl', 'Reagan', 'Yukon', 'CeCe
',
        'Cuddles', 'Claude', 'Jessiga', 'Carter', 'Ole', 'Pherb', 'Bl

ipson',
        'Reptar', 'Trevith', 'Berb', 'Bob', 'Colin', 'Brian', 'Olivié
r',
        'Grady', 'Kobe', 'Freddery', 'Bodie', 'Dunkin', 'Wally', 'Tup
awc',
        'Amber', 'Herschel', 'Edgar', 'Teddy', 'Kingsley', 'Brockly',
        'Richie', 'Molly', 'Vinscent', 'Cedrick', 'Hazel', 'Lolo', 'E
riq',
        'Phred', 'Oddie', 'Maxwell', 'Geoff', 'Covach', 'Durg', 'Fynn
',
        'Ricky', 'Herald', 'Lucky', 'Ferg', 'Trip', 'Clarence', 'Hamr
ick',
        'Brad', 'Pubert', 'Frönq', 'Derby', 'Lizzie', 'Ember', 'Blake
ly',
        'Opal', 'Marq', 'Kramer', 'Barry', 'Tyrone', 'Gordon', 'Baxte
r',
        'Mona', 'Horace', 'Crimson', 'Birf', 'Hammond', 'Lorelei', 'M
arty',
        'Brooks', 'Petrick', 'Hubertson', 'Gerbald', 'Oreo', 'Bruiser
',
        'Perry', 'Bobby', 'Jeph', 'Obi', 'Tino', 'Kulet', 'Sweets', '
Lupe',
        'Tiger', 'Jiminy', 'Griffin', 'Banjo', 'Brandy', 'Lulu', 'Dar
rel',
        'Taco', 'Joey', 'Patrick', 'Kreg', 'Todo', 'Tess', 'Ulysses',
        'Toffee', 'Apollo', 'Carly', 'Asher', 'Glacier', 'Chuck', 'Ch
amp',
        'Ozzie', 'Griswold', 'Cheesy', 'Moofasa', 'Hector', 'Goliath'
,
        'Kawhi', 'Emmie', 'Penelope', 'Willie', 'Rinna', 'Mike', 'Wil
liam',
        'Dwight', 'Evy', 'Hurley', 'Rubio', 'Chompsky', 'Rascal', 'Li
nda',
        'Tug', 'Tango', 'Grizz', 'Jerome', 'Crumpet', 'Jessifer', 'Iz
zy',
        'Ralph', 'Sandy', 'Humphrey', 'Tassy', 'Juckson', 'Chuq', 'Ty
rus',
        'Karl', 'Godzilla', 'Vinnie', 'Kenneth', 'Herm', 'Bert', 'Str
iker',
        'Donny', 'Pepper', 'Bernie', 'Buddah', 'Lenny', 'Arnold', 'Zu
zu',
        'Mollie', 'Laela', 'Tedders', 'Superpup', 'Rufio', 'Jeb', 'Ro
dman',
        'Jonah', 'Chesney', 'Kenny', 'Henry', 'Bobbay', 'Mitch', 'Kai
ya',
        'Acro', 'Aiden', 'Obie', 'Dot', 'Shnuggles', 'Kendall', 'Jeff
ri',
        'Steve', 'Eve', 'Mac', 'Fletcher', 'Kenzie', 'Pumpkin', 'Schn
ozz',
        'Gustaf', 'Cheryl', 'Ed', 'Leonidas', 'Norman', 'Caryl', 'Sco
tt',
        'Taz', 'Darby', 'Jackie', 'Jazz', 'Franq', 'Pippin', 'Rolf',
       'Snickers', 'Ridley', 'Cal', 'Bradley', 'Bubba', 'Tuco', 'Pat
ch',
       'Mojo', 'Batdog', 'Dylan', 'Mark', 'JD', 'Alejandro', 'Scruff
ers',
       'Pip', 'Julius', 'Tanner', 'Sparky', 'Anthony', 'Holly', 'Jet
t',
       'Amy', 'Sage', 'Andy', 'Mason', 'Trigger', 'Antony', 'Creg',
       'Traviss', 'Gin', 'Jeffrie', 'Danny', 'Ester', 'Pluto', 'Bloo
',
       'Edd', 'Paull', 'Willy', 'Herb', 'Damon', 'Peanut', 'Nigel',
       'Butters', 'Sandra', 'Fabio', 'Randall', 'Liam', 'Tommy', 'Be
n',
       'Raphael', 'Julio', 'Andru', 'Kloey', 'Shawwn', 'Skye', 'Koll
in',
       'Ronduh', 'Billl', 'Saydee', 'Dug', 'Tessa', 'Sully', 'Kirk',
       'Ralf', 'Clarq', 'Jaspers', 'Samsom', 'Terrance', 'Harrison',
       'Chaz', 'Jeremy', 'Jaycob', 'Lambeau', 'Ruffles', 'Amélie', '
Bobb',
       'Banditt', 'Kevon', 'Winifred', 'Hanz', 'Churlie', 'Zeek', 'T
imofy',
       'Maks', 'Jomathan', 'Kallie', 'Marvin', 'Spark', 'Gòrdón', 'J
o',
       'DayZ', 'Jareld', 'Torque', 'Ron', 'Skittles', 'Cleopatricia'
,
       'Erik', 'Stu', 'Tedrick', 'Shaggy', 'Filup', 'Kial', 'Naphani
el',
       'Dook', 'Hall', 'Philippe', 'Biden', 'Fwed', 'Genevieve', 'Jo
shwa',
       'Timison', 'Bradlay', 'Pipsy', 'Clybe', 'Keet', 'Carll', 'Joc
kson',
       'Josep', 'Lugan', 'Christoper'], dtype=object)
```

***Define***

Convert NA value in name column to accurate data type.

***Code***

In [100]:

```
archive_clean.name = archive_clean.name.apply(lambda x: x if x != 'None' else np
.nan)
```

***Test***

In [101]:

```
archive_clean.name.unique()
```

Out[101]:

```
Out[101]:

array(['Phineas', 'Tilly', 'Archie', 'Darla', 'Franklin', nan, 'Jax'
,
       'Zoey', 'Cassie', 'Koda', 'Bruno', 'Ted', 'Stuart', 'Oliver',
'Jim',
       'Zeke', 'Ralphus', 'Gerald', 'Jeffrey', 'Canela', 'Maya', 'Mi
ngus',
       'Derek', 'Roscoe', 'Waffles', 'Jimbo', 'Maisey', 'Earl', 'Lol
a',
       'Kevin', 'Yogi', 'Noah', 'Bella', 'Grizzwald', 'Rusty', 'Gus'
,
       'Stanley', 'Alfy', 'Koko', 'Rey', 'Gary', 'Elliot', 'Louis',
       'Jesse', 'Romeo', 'Bailey', 'Duddles', 'Jack', 'Steven', 'Bea
u',
       'Snoopy', 'Shadow', 'Emmy', 'Aja', 'Penny', 'Dante', 'Nelly',
       'Ginger', 'Benedict', 'Venti', 'Goose', 'Nugget', 'Cash', 'Je
d',
       'Sebastian', 'Sierra', 'Monkey', 'Harry', 'Kody', 'Lassie', '
Rover',
       'Napolean', 'Boomer', 'Cody', 'Rumble', 'Clifford', 'Dewey',
       'Scout', 'Gizmo', 'Walter', 'Cooper', 'Harold', 'Shikha', 'Li
li',
       'Jamesy', 'Coco', 'Sammy', 'Meatball', 'Paisley', 'Albus',
       'Neptune', 'Belle', 'Quinn', 'Zooey', 'Dave', 'Jersey', 'Hobb
es',
       'Burt', 'Lorenzo', 'Carl', 'Jordy', 'Milky', 'Trooper', 'Soph
ie',
       'Wyatt', 'Rosie', 'Thor', 'Oscar', 'Callie', 'Cermet', 'Marle
e',
       'Arya', 'Einstein', 'Alice', 'Rumpole', 'Benny', 'Aspen', 'Ja
rod',
       'Wiggles', 'General', 'Sailor', 'Iggy', 'Snoop', 'Kyle', 'Leo
',
       'Riley', 'Noosh', 'Odin', 'Jerry', 'Georgie', 'Rontu', 'Canno
n',
       'Furzey', 'Daisy', 'Tuck', 'Barney', 'Vixen', 'Jarvis', 'Mimo
sa',
       'Pickles', 'Brady', 'Luna', 'Charlie', 'Margo', 'Sadie', 'Han
k',
       'Tycho', 'Indie', 'Winnie', 'George', 'Bentley', 'Max', 'Dawn
',
       'Maddie', 'Monty', 'Sojourner', 'Winston', 'Odie', 'Arlo',
       'Vincent', 'Lucy', 'Clark', 'Mookie', 'Meera', 'Ava', 'Eli',
'Ash',
       'Tucker', 'Tobi', 'Chester', 'Wilson', 'Sunshine', 'Lipton',
       'Bronte', 'Poppy', 'Gidget', 'Rhino', 'Willow', 'Orion', 'Eev
ee',
       'Smiley', 'Miguel', 'Emanuel', 'Kuyu', 'Dutch', 'Pete', 'Scoo
ter',
       'Reggie', 'Lilly', 'Samson', 'Mia', 'Astrid', 'Malcolm', 'Dex
ter',
       'Alfie', 'Fiona', 'Mutt', 'Bear', 'Doobert', 'Beebop', 'Alexa
nder',
       'Sailer', 'Brutus', 'Kona', 'Boots', 'Ralphie', 'Loki', 'Cupi
```

```
d',
        'Pawnd', 'Pilot', 'Ike', 'Mo', 'Toby', 'Sweet', 'Pablo', 'Nal
a',
        'Crawford', 'Gabe', 'Jimison', 'Duchess', 'Harlso', 'Sundance
',
        'Luca', 'Flash', 'Sunny', 'Howie', 'Jazzy', 'Anna', 'Finn', '
Bo',
        'Wafer', 'Tom', 'Florence', 'Autumn', 'Buddy', 'Dido', 'Eugen
e',
        'Ken', 'Strudel', 'Tebow', 'Chloe', 'Timber', 'Binky', 'Moose
',
        'Dudley', 'Comet', 'Akumi', 'Titan', 'Olivia', 'Alf', 'Oshie'
,
        'Chubbs', 'Sky', 'Atlas', 'Eleanor', 'Layla', 'Rocky', 'Baron
',
        'Tyr', 'Bauer', 'Swagger', 'Brandi', 'Mary', 'Moe', 'Halo', '
Augie',
        'Craig', 'Sam', 'Hunter', 'Pavlov', 'Phil', 'Kyro', 'Wallace'
,
        'Ito', 'Seamus', 'Ollie', 'Stephan', 'Lennon', 'Major', 'Duke
',
        'Sansa', 'Shooter', 'Django', 'Diogi', 'Sonny', 'Marley', 'Se
verus',
        'Ronnie', 'Milo', 'Bones', 'Mauve', 'Chef', 'Doc', 'Peaches',
        'Sobe', 'Longfellow', 'Mister', 'Iroh', 'Pancake', 'Snicku',
'Ruby',
        'Brody', 'Mack', 'Nimbus', 'Laika', 'Maximus', 'Dobby', 'More
ton',
        'Juno', 'Maude', 'Lily', 'Newt', 'Benji', 'Nida', 'Robin',
        'Monster', 'BeBe', 'Remus', 'Levi', 'Mabel', 'Misty', 'Betty'
,
        'Mosby', 'Maggie', 'Bruce', 'Happy', 'Ralphy', 'Brownie', 'Ri
zzy',
        'Stella', 'Butter', 'Frank', 'Tonks', 'Lincoln', 'Rory', 'Log
an',
        'Dale', 'Rizzo', 'Arnie', 'Mattie', 'Pinot', 'Dallas', 'Hero'
,
        'Frankie', 'Stormy', 'Reginald', 'Balto', 'Mairi', 'Loomis',
'Godi',
        'Cali', 'Deacon', 'Timmy', 'Sampson', 'Chipson', 'Combo', 'Oa
kley',
        'Dash', 'Hercules', 'Jay', 'Mya', 'Strider', 'Wesley', 'Solom
on',
        'Huck', 'Blue', 'Anakin', 'Finley', 'Sprinkles', 'Heinrich',
        'Shakespeare', 'Chelsea', 'Bungalo', 'Chip', 'Grey', 'Rooseve
lt',
        'Willem', 'Davey', 'Dakota', 'Fizz', 'Dixie', 'Al', 'Jackson'
,
        'Carbon', 'Klein', 'DonDon', 'Kirby', 'Lou', 'Chevy', 'Tito',
        'Philbert', 'Louie', 'Rupert', 'Rufus', 'Brudge', 'Shadoe', '
Angel',
        'Brat', 'Tove', 'Gromit', 'Aubie', 'Kota', 'Leela', 'Glenn',
        'Shelby', 'Sephie', 'Bonaparte', 'Albert', 'Wishes', 'Rose',
'Theo',

       'Rocco', 'Fido', 'Emma', 'Spencer', 'Lilli', 'Boston', 'Brand
onald',
       'Corey', 'Leonard', 'Beckham', 'Devón', 'Gert', 'Watson', 'Ke
ith',
       'Dex', 'Ace', 'Tayzie', 'Grizzie', 'Fred', 'Gilbert', 'Meyer'
,
       'Zoe', 'Stewie', 'Calvin', 'Lilah', 'Spanky', 'Jameson', 'Pip
er',
       'Atticus', 'Blu', 'Dietrich', 'Divine', 'Tripp', 'Cora', 'Hux
ley',
       'Keurig', 'Bookstore', 'Linus', 'Abby', 'Shiloh', 'Gustav', '
Arlen',
       'Percy', 'Lenox', 'Sugar', 'Harvey', 'Blanket', 'Geno', 'Star
k',
       'Beya', 'Kilo', 'Kayla', 'Maxaroni', 'Bell', 'Doug', 'Edmund'
,
       'Aqua', 'Theodore', 'Baloo', 'Chase', 'Nollie', 'Rorie', 'Sim
ba',
       'Charles', 'Bayley', 'Axel', 'Storkson', 'Remy', 'Chadrick',
       'Kellogg', 'Buckley', 'Livvie', 'Terry', 'Hermione', 'Ralpher
',
       'Aldrick', 'Larry', 'Rooney', 'Crystal', 'Ziva', 'Stefan',
       'Pupcasso', 'Puff', 'Flurpson', 'Coleman', 'Enchilada', 'Raym
ond',
       'Rueben', 'Cilantro', 'Karll', 'Sprout', 'Blitz', 'Bloop', 'C
olby',
       'Lillie', 'Ashleigh', 'Kreggory', 'Sarge', 'Luther', 'Ivar',
       'Jangle', 'Schnitzel', 'Panda', 'Berkeley', 'Ralphé', 'Charle
son',
       'Clyde', 'Harnold', 'Sid', 'Pippa', 'Otis', 'Carper', 'Bowie'
,
       'Alexanderson', 'Suki', 'Barclay', 'Skittle', 'Ebby', 'Flávio
',
       'Smokey', 'Link', 'Jennifur', 'Ozzy', 'Bluebert', 'Stephanus'
,
       'Bubbles', 'Zeus', 'Bertson', 'Nico', 'Michelangelope', 'Siba
',
       'Calbert', 'Curtis', 'Travis', 'Thumas', 'Kanu', 'Lance', 'Op
ie',
       'Stubert', 'Kane', 'Olive', 'Chuckles', 'Staniel', 'Sora', 'B
eemo',
       'Gunner', 'Lacy', 'Tater', 'Olaf', 'Cecil', 'Vince', 'Karma',
       'Billy', 'Walker', 'Rodney', 'Klevin', 'Malikai', 'Bobble', '
River',
       'Jebberson', 'Remington', 'Farfle', 'Jiminus', 'Harper', 'Cla
rkus',
       'Finnegus', 'Cupcake', 'Kathmandu', 'Ellie', 'Katie', 'Kara',
       'Adele', 'Zara', 'Ambrose', 'Jimothy', 'Bode', 'Terrenth', 'R
eese',
       'Chesterson', 'Lucia', 'Bisquick', 'Ralphson', 'Socks', 'Ramb
o',
       'Rudy', 'Fiji', 'Rilo', 'Bilbo', 'Coopson', 'Yoda', 'Millie',
        'Chet', 'Crouton', 'Daniel', 'Kaia', 'Murphy', 'Dotsy', 'Eazy
',
        'Coops', 'Fillup', 'Miley', 'Charl', 'Reagan', 'Yukon', 'CeCe
',
        'Cuddles', 'Claude', 'Jessiga', 'Carter', 'Ole', 'Pherb', 'Bl
ipson',
        'Reptar', 'Trevith', 'Berb', 'Bob', 'Colin', 'Brian', 'Olivié
r',
        'Grady', 'Kobe', 'Freddery', 'Bodie', 'Dunkin', 'Wally', 'Tup
awc',
        'Amber', 'Herschel', 'Edgar', 'Teddy', 'Kingsley', 'Brockly',
        'Richie', 'Molly', 'Vinscent', 'Cedrick', 'Hazel', 'Lolo', 'E
riq',
        'Phred', 'Oddie', 'Maxwell', 'Geoff', 'Covach', 'Durg', 'Fynn
',
        'Ricky', 'Herald', 'Lucky', 'Ferg', 'Trip', 'Clarence', 'Hamr
ick',
        'Brad', 'Pubert', 'Frönq', 'Derby', 'Lizzie', 'Ember', 'Blake
ly',
        'Opal', 'Marq', 'Kramer', 'Barry', 'Tyrone', 'Gordon', 'Baxte
r',
        'Mona', 'Horace', 'Crimson', 'Birf', 'Hammond', 'Lorelei', 'M
arty',
        'Brooks', 'Petrick', 'Hubertson', 'Gerbald', 'Oreo', 'Bruiser
',
        'Perry', 'Bobby', 'Jeph', 'Obi', 'Tino', 'Kulet', 'Sweets', '
Lupe',
        'Tiger', 'Jiminy', 'Griffin', 'Banjo', 'Brandy', 'Lulu', 'Dar
rel',
        'Taco', 'Joey', 'Patrick', 'Kreg', 'Todo', 'Tess', 'Ulysses',
        'Toffee', 'Apollo', 'Carly', 'Asher', 'Glacier', 'Chuck', 'Ch
amp',
        'Ozzie', 'Griswold', 'Cheesy', 'Moofasa', 'Hector', 'Goliath'
,
        'Kawhi', 'Emmie', 'Penelope', 'Willie', 'Rinna', 'Mike', 'Wil
liam',
        'Dwight', 'Evy', 'Hurley', 'Rubio', 'Chompsky', 'Rascal', 'Li
nda',
        'Tug', 'Tango', 'Grizz', 'Jerome', 'Crumpet', 'Jessifer', 'Iz
zy',
        'Ralph', 'Sandy', 'Humphrey', 'Tassy', 'Juckson', 'Chuq', 'Ty
rus',
        'Karl', 'Godzilla', 'Vinnie', 'Kenneth', 'Herm', 'Bert', 'Str
iker',
        'Donny', 'Pepper', 'Bernie', 'Buddah', 'Lenny', 'Arnold', 'Zu
zu',
        'Mollie', 'Laela', 'Tedders', 'Superpup', 'Rufio', 'Jeb', 'Ro
dman',
        'Jonah', 'Chesney', 'Kenny', 'Henry', 'Bobbay', 'Mitch', 'Kai
ya',
        'Acro', 'Aiden', 'Obie', 'Dot', 'Shnuggles', 'Kendall', 'Jeff
ri',
        'Steve', 'Eve', 'Mac', 'Fletcher', 'Kenzie', 'Pumpkin', 'Schn
```

```
ozz',
       'Gustaf', 'Cheryl', 'Ed', 'Leonidas', 'Norman', 'Caryl', 'Sco
tt',
       'Taz', 'Darby', 'Jackie', 'Jazz', 'Franq', 'Pippin', 'Rolf',
       'Snickers', 'Ridley', 'Cal', 'Bradley', 'Bubba', 'Tuco', 'Pat
ch',
       'Mojo', 'Batdog', 'Dylan', 'Mark', 'JD', 'Alejandro', 'Scruff
ers',
       'Pip', 'Julius', 'Tanner', 'Sparky', 'Anthony', 'Holly', 'Jet
t',
       'Amy', 'Sage', 'Andy', 'Mason', 'Trigger', 'Antony', 'Creg',
       'Traviss', 'Gin', 'Jeffrie', 'Danny', 'Ester', 'Pluto', 'Bloo
',
       'Edd', 'Paull', 'Willy', 'Herb', 'Damon', 'Peanut', 'Nigel',
       'Butters', 'Sandra', 'Fabio', 'Randall', 'Liam', 'Tommy', 'Be
n',
       'Raphael', 'Julio', 'Andru', 'Kloey', 'Shawwn', 'Skye', 'Koll
in',
       'Ronduh', 'Billl', 'Saydee', 'Dug', 'Tessa', 'Sully', 'Kirk',
       'Ralf', 'Clarq', 'Jaspers', 'Samsom', 'Terrance', 'Harrison',
       'Chaz', 'Jeremy', 'Jaycob', 'Lambeau', 'Ruffles', 'Amélie', '
Bobb',
       'Banditt', 'Kevon', 'Winifred', 'Hanz', 'Churlie', 'Zeek', 'T
imofy',
       'Maks', 'Jomathan', 'Kallie', 'Marvin', 'Spark', 'Gòrdón', 'J
o',
       'DayZ', 'Jareld', 'Torque', 'Ron', 'Skittles', 'Cleopatricia'
,
       'Erik', 'Stu', 'Tedrick', 'Shaggy', 'Filup', 'Kial', 'Naphani
el',
       'Dook', 'Hall', 'Philippe', 'Biden', 'Fwed', 'Genevieve', 'Jo
shwa',
       'Timison', 'Bradlay', 'Pipsy', 'Clybe', 'Keet', 'Carll', 'Joc
kson',
       'Josep', 'Lugan', 'Christoper'], dtype=object)
```

***Define***

Parse the datetime column into seperate columns.

***Code***

In [102]:

```
archive_clean['date'] = archive_clean.timestamp.apply(lambda x: x.strftime('%d-%
b-%Y'))
archive_clean['time'] = archive_clean.timestamp.apply(lambda x: x.strftime('%I:%
M:%S %p'))
```

***Test***

```
In [103]:
```

```
archive_clean.sample(5)
```

```
Out[103]:
```

| | tweet_id | in_reply_to_status_id | in_reply_to_user_id | timestamp | |
|---|---|---|---|---|---|
| **696** | 786664955043049472 | NaN | NaN | 2016-10-13 20:28:35 | <a href r... |
| **517** | 810896069567610880 | NaN | NaN | 2016-12-19 17:14:23 | <a href r... |
| **1566** | 687841446767013888 | NaN | NaN | 2016-01-15 03:39:15 | <a h rel=' |
| **919** | 756526248105566208 | NaN | NaN | 2016-07-22 16:28:07 | <a href r... |
| **613** | 796865951799083009 | NaN | NaN | 2016-11-11 00:03:42 | <a href r... |

***Define***

Extract the the rating numerator from 'text' column

***Code***

```
In [104]:
```

```
# extract rating numerators from 'text' column and attribute them to'rating_nume
rator' column
archive_clean['rating_numerator'] = archive_clean.text.str.extract(".*\s(\d+)\/\
d+.*", expand = True)
```

***Test***

```
In [105]:
```

```
# now the numerator in the 'rating_numerator' column should be exactly the same
as in 'text' column
(archive_clean.rating_numerator == archive_clean.text.str.extract(".*\s(\d+)\/\d
+.*")).any()
```

```
/Users/shilinli/anaconda3/lib/python3.6/site-packages/ipykernel_laun
cher.py:2: FutureWarning: currently extract(expand=None) means expan
d=False (return Index/Series/DataFrame) but in a future version of p
andas this will be changed to expand=True (return DataFrame)
```

```
Out[105]:
```

```
True
```

***Define***

Convert the data type in both 'rating_numerator' and 'rating_denominator' columns as float

***Code***

```
In [106]:
```

```
archive_clean.rating_numerator = archive_clean.rating_numerator.astype(float)
archive_clean.rating_denominator = archive_clean.rating_denominator.astype(float
)
```

***Test***

In [107]:

```
archive_clean.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2175 entries, 0 to 2355
Data columns (total 19 columns):
tweet_id                      2175 non-null int64
in_reply_to_status_id         78 non-null float64
in_reply_to_user_id           78 non-null float64
timestamp                     2175 non-null datetime64[ns]
source                        2175 non-null object
text                          2175 non-null object
retweeted_status_id           0 non-null float64
retweeted_status_user_id      0 non-null float64
retweeted_status_timestamp    0 non-null object
expanded_urls                 2117 non-null object
rating_numerator              2134 non-null float64
rating_denominator            2175 non-null float64
name                          1390 non-null object
doggo                         2175 non-null object
floofer                       2175 non-null object
pupper                        2175 non-null object
puppo                         2175 non-null object
date                          2175 non-null object
time                          2175 non-null object
dtypes: datetime64[ns](1), float64(6), int64(1), object(11)
memory usage: 339.8+ KB
```

***Define***

Melt dog stage column into a single column.

***Code***

In [108]:

```
# melt the 4 stage columns and transfer the values under each tweet_id into a sm
all stage list
# append each stage list into a bigger list
stage = pd.melt(archive_clean, id_vars=['tweet_id'], value_vars=['doggo', 'floof
er', "pupper", "puppo"])
stage_list = []
for ids in stage.tweet_id.unique():
    stage_list.append(stage.query('tweet_id == @ids').value.tolist())
```

In [109]:

```
# remove repeat value in each stage list
stage_list2 = []
for e in stage_list:
    stage_list2.append(list(set(e)))
```

In [110]:

```
# remove the 'None' in stage list which contains any stage value, such as 'doggo
', 'floofer', "pupper", "puppo"
stage_list3 = []
for e in stage_list2:
    if len(e) > 1:
        e.remove('None')
        stage_list3.append(e)
    else:
        stage_list3.append(e)
```

In [111]:

```
# remove '[]' which wrapped outside the stage value
stage_list4 = []
for e in stage_list3:
    if len(e) > 1:
        stage_list4.append(",".join(e))
    else:
        stage_list4.append(e[0])
```

In [112]:

```
archive_clean['stage'] = stage_list4
```

***Test***

In [113]:

```
archive_clean.sample(5)
```

```
Out[113]:
```

| | tweet_id | in_reply_to_status_id | in_reply_to_user_id | timestamp | |
|---|---|---|---|---|---|
| **1754** | 678798276842360832 | NaN | NaN | 2015-12-21 04:44:55 | <a href= r... |
| **1673** | 682303737705140231 | NaN | NaN | 2015-12-30 20:54:22 | <a href= r... |
| **719** | 783391753726550016 | NaN | NaN | 2016-10-04 19:42:03 | <a href= r... |
| **1551** | 689143371370250240 | NaN | NaN | 2016-01-18 17:52:38 | <a href= r... |
| **135** | 866450705531457537 | NaN | NaN | 2017-05-22 00:28:40 | <a href= r... |

```
In [114]:
```

```
# see if certain none NA values contains more than one stage, as some dogs posse
ssed more than one stage
archive_clean.query('stage != "None"').stage
```

```
Out[114]:

9                    doggo
12                   puppo
```

```
14              puppo
29             pupper
43              doggo
46            floofer
49             pupper
56             pupper
71              puppo
82             pupper
92             pupper
94              puppo
98             pupper
99              doggo
107            pupper
108             doggo
110             doggo
121             doggo
129             puppo
135            pupper
168             puppo
172             doggo
191       doggo,puppo
199            pupper
200     doggo,floofer
220            pupper
240             doggo
248             doggo
249            pupper
293            pupper
              ...
1875           pupper
1880           pupper
1889           pupper
1897           pupper
1903           pupper
1907           pupper
1915           pupper
1921           pupper
1930           pupper
1936           pupper
1937           pupper
1945           pupper
1948           pupper
1954           pupper
1956           pupper
1960           pupper
1967           pupper
1970           pupper
1974           pupper
1977           pupper
1980           pupper
1981           pupper
1985           pupper
1991           pupper
```

```
1992            pupper

1995            pupper
2002            pupper
2009            pupper
2015            pupper
2017            pupper
Name: stage, Length: 344, dtype: object
```

***Define***

Convert 'None' in 'stage' column to programmable NA values.

***Code***

In [115]:

```
archive_clean.stage = archive_clean.stage.replace('None', np.nan)
```

***Test***

In [116]:

```
# 'None' should be replaced by nan
archive_clean.stage.unique()
```

Out[116]:

```
array([nan, 'doggo', 'puppo', 'pupper', 'floofer', 'doggo,puppo',
       'doggo,floofer', 'doggo,pupper'], dtype=object)
```

***Define***

Convert tweet_id data type from integer to string.

***Code***

In [117]:

```
archive_clean.tweet_id = archive_clean.tweet_id.astype(str)
```

***Test***

In [118]:

```
archive_clean.tweet_id.dtype
```

Out[118]:

```
dtype('O')
```

***Define***

Extract gender info from 'source' column.

***Code***

In [119]:

```
# all kinds of gender expression in 'text' column
male = ['He', 'he', 'Him', 'him', "He's", "he's", 'His', 'his', 'Himself', 'hims
elf', ]
female = ['She', 'she', 'Her', 'her', 'Hers', 'hers', 'Herself', 'herself', "She
's", "she's"]

gender = []

# design a function to extract gender info
def gender_fun(data):
    for text in data.text:
        if (set(text.split()) & set(male)):
            gender.append('male')
        elif (set(text.split()) & set(female)):
            gender.append('female')
        else:
            gender.append('None')


gender_fun(archive_clean)

# add gender column and attribute the values from gender list
archive_clean['gender'] = gender

# Convert 'None' to NA
archive_clean.gender = archive_clean.gender.replace('None', np.nan)
```

***Test***

In [120]:

```
# the length of 'gender' column should be the same as dataframe
# 'gender' coulumn should have three types of values, namely male, female and No
ne
archive_clean.shape, len(gender), set(gender)
```

Out[120]:

```
((2175, 21), 2175, {'None', 'female', 'male'})
```

In [121]:

```
# check the details of composition in 'gender' column
Counter(gender)
```

Out[121]:

```
Counter({'None': 771, 'female': 347, 'male': 1057})
```

In [122]:

```
# check if None has been converted to programmable NA
archive_clean.query('gender != gender').sample().gender
```

Out[122]:

```
973    NaN
Name: gender, dtype: object
```

***Test***

***Define***

Keep only the necessary columns for analysis, 'tweet_id', 'time_stamp', 'rating_numerator', 'rating_denominator', 'name', 'date', 'time', 'stage'.

***Code***

```
In [123]:
```

```
archive_clean.drop(['in_reply_to_status_id',
 'in_reply_to_user_id',
 'source',
 'text',
 'retweeted_status_id',
 'retweeted_status_user_id',
 'retweeted_status_timestamp',
 'expanded_urls',
 'doggo',
 'floofer',
 'pupper',
 'puppo'], 1, inplace = True)
```

***Test***

```
In [124]:
```

```
archive_clean.head()
```

```
Out[124]:
```

| | tweet_id | timestamp | rating_numerator | rating_denominator | name | d |
|---|---|---|---|---|---|---|
| **0** | 892420643555336193 | 2017-08-01 16:23:56 | 13.0 | 10.0 | Phineas | 0 A 2 |
| **1** | 892177421306343426 | 2017-08-01 00:17:27 | 13.0 | 10.0 | Tilly | 0 A 2 |
| **2** | 891815181378084864 | 2017-07-31 00:18:03 | 12.0 | 10.0 | Archie | 3 J 2 |
| **3** | 891689557279858688 | 2017-07-30 15:58:51 | 13.0 | 10.0 | Darla | 3 J 2 |
| **4** | 891327558926688256 | 2017-07-29 16:00:24 | 12.0 | 10.0 | Franklin | 2 J 2 |

## `image_clean` table

***Define***

Remove rows which contain duplicated 'jpg_url' values.

***Code***

```
In [125]:
```

```
image_clean = image_clean.drop_duplicates('jpg_url')
```

***Test***

```
In [126]:
```

```
sum(image_clean.jpg_url.duplicated())
```

```
Out[126]:

0
```

***Define***

Simplify the table by keeping only one prediction, according to the odds priority order is as p1 > p2 > p3.

***Code***

In [127]:

```
predictions = []
odds = []

# store the fisrt true algorithm with it's odds

# dog_prediction_confidence function:
# find the first true algorithm and append it to a list with it's odds
# if flase, predictions list will have values of NaN
def dog_prediction(data):
        if data.p1_dog == True:
            predictions.append(data.p1)
            odds.append(data.p1_conf)
        elif data.p2_dog == True:
            predictions.append(data.p2)
            odds.append(data.p2_conf)
        elif data.p3_dog == True:
            predictions.append(data.p3)
            odds.append(data.p3_conf)
        else:
            predictions.append(np.nan)
            odds.append(0)

image_clean.apply(dog_prediction, axis = 1)
image_clean['predictions'] = predictions
image_clean['odds'] = odds
```

***Test***

In [128]:

```
image_clean.sample(5)
```

Out[128]:

| | tweet_id | jpg_url | img_ |
|---|---|---|---|
| **56** | 667065535570550784 | https://pbs.twimg.com/media/CUHkkJpXIAA2w3n.jpg | 1 |
| **315** | 671735591348891648 | https://pbs.twimg.com/media/CVJ79MzW4AEpTom.jpg | 2 |
| **616** | 680191257256136705 | https://pbs.twimg.com/media/CXCGVXyWsAAAVHE.jpg | 1 |
| **586** | 679047485189439488 | https://pbs.twimg.com/media/CWx2FaLWcAEQ3vh.jpg | 1 |
| **1447** | 776088319444877312 | https://pbs.twimg.com/media/CsU4NKkW8AUI5eG.jpg | 3 |

***Define***

Convert data type in 'tweet_id' column from integer to string.

***Code***

```
In [129]:
```

```
image_clean.tweet_id = image_clean.tweet_id.astype(str)
```

***Test***

```
In [130]:
```

```
image_clean.tweet_id.dtype
```

```
Out[130]:

dtype('O')
```

***Define***

Drop the columns we don't need.

***Code***

```
In [131]:
```

```
image_clean = image_clean.loc[:,['tweet_id', 'jpg_url', 'img_num', 'predictions'
, 'odds']]
```

***Test***

In [132]:

```
image_clean.sample(5)
```

Out[132]:

| | tweet_id | jpg_url | img_ |
|---|---|---|---|
| **1776** | 828376505180889089 | https://pbs.twimg.com/media/C378BwxWMAA6CNK.jpg | 1 |
| **515** | 676263575653122048 | https://pbs.twimg.com/media/CWKSIfUUYAAiOBO.jpg | 1 |
| **1888** | 848212111729840128 | https://pbs.twimg.com/media/C8V0aI5V0AAgO9m.jpg | 1 |
| **1374** | 762699858130116608 | https://pbs.twimg.com/media/CpWnecZWIAAUFwt.jpg | 1 |
| **450** | 674739953134403584 | https://pbs.twimg.com/media/CV0oaHFW4AA9Coi.jpg | 1 |

## `tweet_df_clean` table

### *Define*

Exteact followers_count and favourites_count values from user column.

### *Code*

In [133]:

```
for key in tweet_df_clean.user.to_dict().keys():
    tweet_df_clean['followers_count'] = tweet_df_clean.user[key]['followers_coun
t']
    tweet_df_clean['favourites_count'] = tweet_df_clean.user[key]['favourites_co
unt']
```

### *Test*

In [134]:

```
list(tweet_df_clean)
```

Out[134]:

```
['contributors',
 'coordinates',
 'created_at',
 'display_text_range',
 'entities',
 'extended_entities',
 'favorite_count',
 'favorited',
 'full_text',
 'geo',
 'id',
 'id_str',
 'in_reply_to_screen_name',
 'in_reply_to_status_id',
 'in_reply_to_status_id_str',
 'in_reply_to_user_id',
 'in_reply_to_user_id_str',
 'is_quote_status',
 'lang',
 'place',
 'possibly_sensitive',
 'possibly_sensitive_appealable',
 'quoted_status',
 'quoted_status_id',
 'quoted_status_id_str',
 'quoted_status_permalink',
 'retweet_count',
 'retweeted',
 'retweeted_status',
 'source',
 'truncated',
 'user',
 'followers_count',
 'favourites_count']
```

### *Define*

Rename the id column to "tweet_id" to match the other 2 tables.

### *Code*

In [135]:

```
tweet_df_clean.rename(columns = {'id': 'tweet_id'}, inplace = True)
```

***Test***

In [136]:

```
list(tweet_df_clean)
```

Out[136]:

```
['contributors',
 'coordinates',
 'created_at',
 'display_text_range',
 'entities',
 'extended_entities',
 'favorite_count',
 'favorited',
 'full_text',
 'geo',
 'tweet_id',
 'id_str',
 'in_reply_to_screen_name',
 'in_reply_to_status_id',
 'in_reply_to_status_id_str',
 'in_reply_to_user_id',
 'in_reply_to_user_id_str',
 'is_quote_status',
 'lang',
 'place',
 'possibly_sensitive',
 'possibly_sensitive_appealable',
 'quoted_status',
 'quoted_status_id',
 'quoted_status_id_str',
 'quoted_status_permalink',
 'retweet_count',
 'retweeted',
 'retweeted_status',
 'source',
 'truncated',
 'user',
 'followers_count',
 'favourites_count']
```

***Define***

Reset the chaotic index by sequential order.

### *Code*

In [137]:

```
tweet_df_clean = tweet_df_clean.reset_index(drop=True)
```

### *Test*

In [138]:

```
tweet_df_clean.head()
```

```
Out[138]:
```

| | contributors | coordinates | created_at | display_text_range | entities | extendec |
|---|---|---|---|---|---|---|
| **0** | NaN | NaN | 2015-11-15 22:32:08 | [0, 131] | {'hashtags': [], 'media': [{'display_url': 'pi... | {'media': [{'display_ur 'pic.twitter.c |
| **1** | NaN | NaN | 2015-11-15 23:05:30 | [0, 139] | {'hashtags': [], 'media': [{'display_url': 'pi... | {'media': [{'display_ur 'pic.twitter.c |
| **2** | NaN | NaN | 2015-11-15 23:21:54 | [0, 130] | {'hashtags': [], 'media': [{'display_url': 'pi... | {'media': [{'display_ur 'pic.twitter.c |
| **3** | NaN | NaN | 2015-11-16 00:04:52 | [0, 137] | {'hashtags': [], 'media': [{'display_url': 'pi... | {'media': [{'display_ur 'pic.twitter.c |
| **4** | NaN | NaN | 2015-11-16 00:24:50 | [0, 120] | {'hashtags': [], 'media': [{'display_url': 'pi... | {'media': [{'display_ur 'pic.twitter.c |

5 rows × 34 columns

***Define***

Convert value type in id column from a integer to string.

***Code***

In [139]:

```
tweet_df_clean.tweet_id = tweet_df_clean.tweet_id.astype(str)
```

***Test***

In [140]:

```
tweet_df_clean.tweet_id.dtype
```

Out[140]:

```
dtype('O')
```

***Define***

Remove the columns we don't need.

***Code***

In [141]:

```
tweet_df_clean = tweet_df_clean.loc[:, ['tweet_id', 'favorite_count', 'retweet_c
ount', 'followers_count', 'favourites_count', \
                            'created_at']]
```

***Test***

In [142]:

```
tweet_df_clean.head()
```

Out[142]:

| | tweet_id | favorite_count | retweet_count | followers_count | favourites_co |
|---|---|---|---|---|---|
| **0** | 666020888022790144 | 2565 | 517 | 7070857 | 135451 |
| **1** | 666029285002620928 | 130 | 47 | 7070857 | 135451 |
| **2** | 666033412701032448 | 125 | 44 | 7070857 | 135451 |
| **3** | 666044226329800704 | 298 | 141 | 7070857 | 135451 |
| **4** | 666049248165822464 | 109 | 41 | 7070857 | 135451 |

***Define***

Consolidate all the 3 tables

***Code***

In [143]:

```
df_master = pd.merge(pd.merge(archive_clean, image_clean, on='tweet_id'), tweet_
df_clean, on = 'tweet_id')
```

***Test***

In [144]:

```
df_master.head()
```

Out[144]:

| | tweet_id | timestamp | rating_numerator | rating_denominator | name | da |
|---|---|---|---|---|---|---|
| **0** | 891815181378084864 | 2017-07-31 00:18:03 | 12.0 | 10.0 | Archie | 31 Ju 20 |
| **1** | 891689557279858688 | 2017-07-30 15:58:51 | 13.0 | 10.0 | Darla | 30 Ju 20 |
| **2** | 891327558926688256 | 2017-07-29 16:00:24 | 12.0 | 10.0 | Franklin | 29 Ju 20 |
| **3** | 891087950875897856 | 2017-07-29 00:08:17 | 13.0 | 10.0 | NaN | 29 Ju 20 |
| **4** | 890729181411237888 | 2017-07-28 00:22:40 | 13.0 | 10.0 | NaN | 28 Ju 20 |

***Remove redundant column***

In [145]:

```
df_master.drop(['created_at'], axis = 1, inplace = True)
df_master.head()
```

Out[145]:

| | tweet_id | timestamp | rating_numerator | rating_denominator | name | da |
|---|---|---|---|---|---|---|
| **0** | 891815181378084864 | 2017-07-31 00:18:03 | 12.0 | 10.0 | Archie | 31 Ju 20 |
| **1** | 891689557279858688 | 2017-07-30 15:58:51 | 13.0 | 10.0 | Darla | 30 Ju 20 |
| **2** | 891327558926688256 | 2017-07-29 16:00:24 | 12.0 | 10.0 | Franklin | 29 Ju 20 |
| **3** | 891087950875897856 | 2017-07-29 00:08:17 | 13.0 | 10.0 | NaN | 29 Ju 20 |
| **4** | 890729181411237888 | 2017-07-28 00:22:40 | 13.0 | 10.0 | NaN | 28 Ju 20 |

## Part II: Data Visualization

In [146]:

```
# save file
df_master.to_csv('twitter_archive_master.csv', index=False, encoding = 'utf-8')
```

In [147]:

```
# order the table by time
df_master.sort_values('timestamp', inplace =True)

# reset the index
df_master = df_master.set_index('timestamp')

# convert data type in 'gender and 'stage' columns to category for later visuali
zation
df_master.gender = df_master.gender.astype('category')
df_master.stage = df_master.stage.astype('category')
```

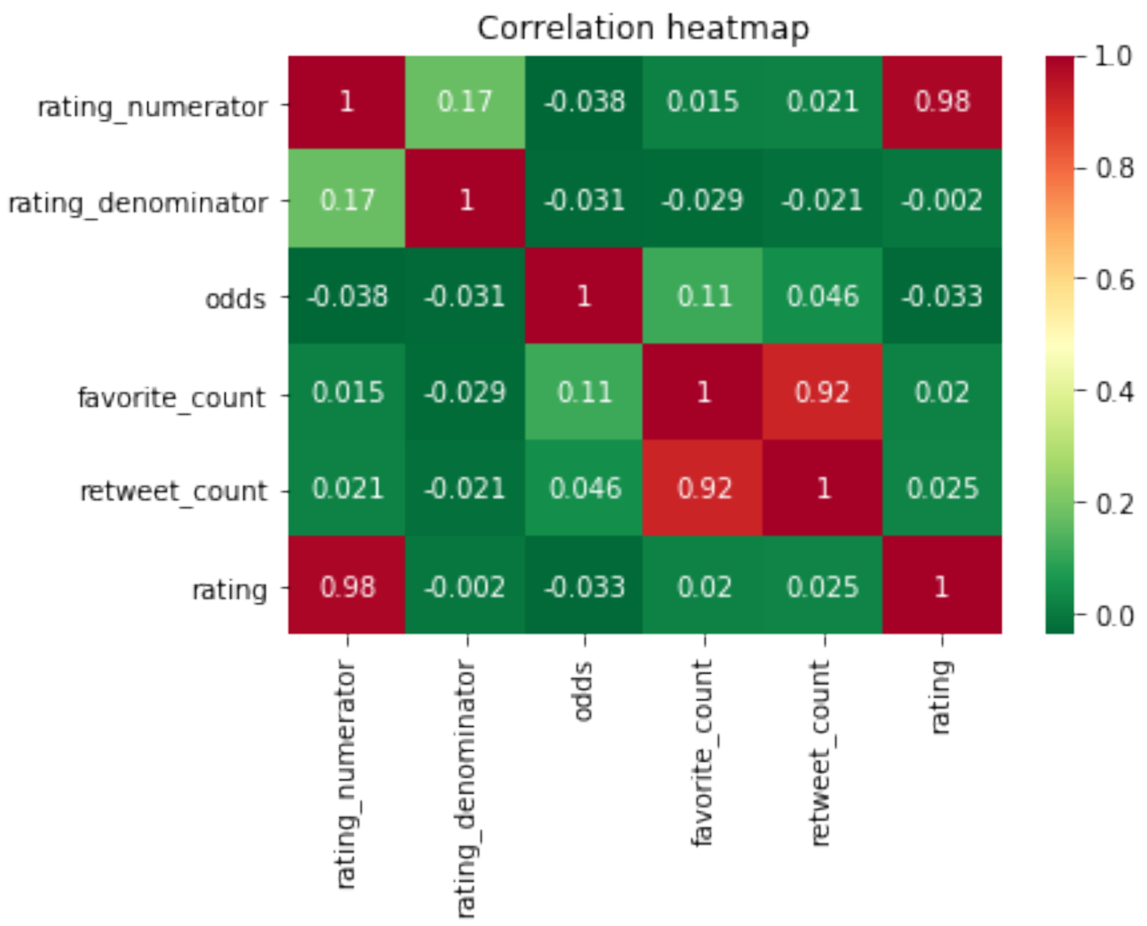
In [148]:

```
# add rating column
df_master['rating'] = df_master.rating_numerator/df_master.rating_denominator

# subset the dataframe for correlation heatmap
df_master1 = df_master.drop(['date', 'time', 'tweet_id', 'jpg_url', 'name', \

                              'stage', 'predictions', 'img_num', 'followers_count
', 'favourites_count'], axis = 1)
```

### Correlation with rating

In [149]:

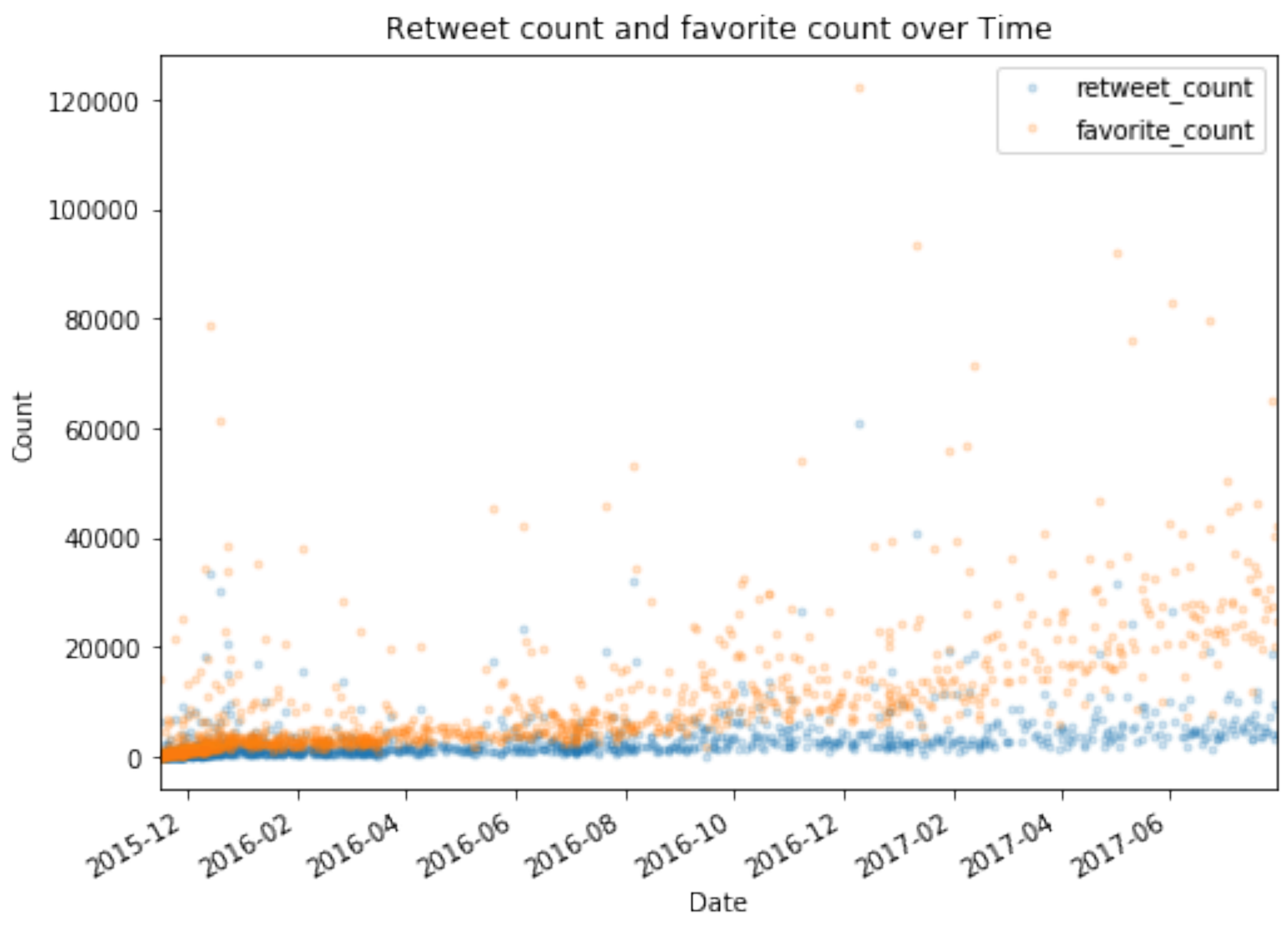
```
# correlation heatmap
corr = df_master1.corr()
plt.title('Correlation heatmap')
sns.heatmap(corr,
            xticklabels=corr.columns.values,
            yticklabels=corr.columns.values,
           annot = True,
           cmap='RdYlGn_r');
```



### Retweet_count VS favorite_count

In [150]:

```
df_master.plot.line(y =['retweet_count', 'favorite_count'], style = '.', alpha =
.2, figsize=(8,6))

plt.title('Retweet count and favorite count over Time')
plt.xlabel('Date')
plt.ylabel('Count');
```



## Top 10 predictions

In [151]:

```
df_master.sort_values('odds', ascending = False)[0:9][['predictions', 'odds']]
```

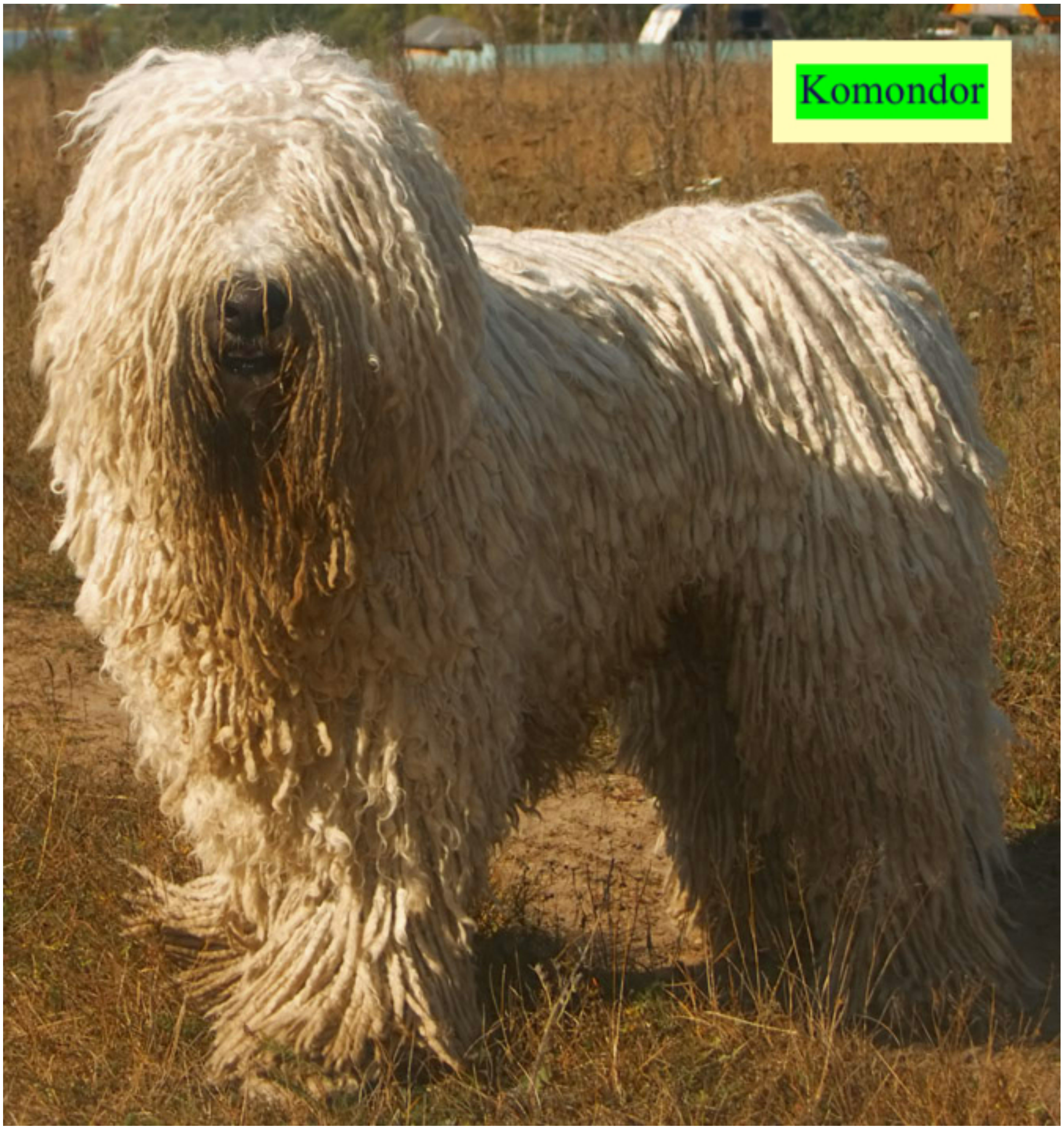
Out[151]:

| | **predictions** | **odds** |
|---|---|---|
| **timestamp** | | |
| **2015-11-23 03:46:18** | komondor | 0.999956 |
| **2016-02-10 16:51:59** | Labrador_retriever | 0.999885 |
| **2016-03-15 02:25:31** | chow | 0.999837 |
| **2016-12-31 00:08:17** | dalmatian | 0.999828 |
| **2016-02-10 03:22:44** | Great_Dane | 0.999223 |
| **2017-07-15 23:25:31** | French_bulldog | 0.999201 |
| **2017-06-08 14:20:41** | pug | 0.999120 |
| **2015-11-19 20:14:03** | Rottweiler | 0.999091 |
| **2016-01-08 01:16:17** | pug | 0.999044 |

In [152]:

```
Image(filename="Komondor.jpg")
```

```
Out[152]:
```



The above image is adapted from Nikki68 (https://commons.wikimedia.org/wiki/File:Komondor_delvin.jpg).

In [2]:

```
Image(filename="Pug.jpg")
```

Out[2]:

The above image is adapted from wikimedia (https://commons.wikimedia.org/wiki/File:Mops-duke-mopszucht-vom-maegdebrunnen.jpg).

## Descriptive statistics of the table

In [154]:

```
df_master.describe()
```

Out[154]:

| | rating_numerator | rating_denominator | img_num | odds | favorite_cou |
|---|---|---|---|---|---|
| **count** | 1277.000000 | 1299.000000 | 1299.000000 | 1299.000000 | 1299.000000 |
| **mean** | 12.837118 | 10.545804 | 1.187067 | 0.460938 | 8262.740570 |
| **std** | 51.553379 | 7.874498 | 0.540746 | 0.338136 | 11420.680957 |
| **min** | 1.000000 | 2.000000 | 1.000000 | 0.000000 | 80.000000 |
| **25%** | 10.000000 | 10.000000 | 1.000000 | 0.140538 | 1714.500000 |
| **50%** | 11.000000 | 10.000000 | 1.000000 | 0.456092 | 3821.000000 |
| **75%** | 12.000000 | 10.000000 | 1.000000 | 0.767926 | 10236.00000 |
| **max** | 1776.000000 | 170.000000 | 4.000000 | 0.999956 | 122431.0000 |

## Investigate the outlier

In [155]:

```
df_master[df_master['rating_numerator'] == df_master['rating_numerator'].max()]
```

Out[155]:

| | tweet_id | rating_numerator | rating_denominator | name | date |
|---|---|---|---|---|---|
| **timestamp** | | | | | |
| **2016-07-04 15:00:45** | 749981277374128128 | 1776.0 | 10.0 | Atticus | 04-Jul-2016 |

```
In [156]:
```

```
df_master[df_master['rating_denominator'] == df_master['rating_denominator'].max
()]
```

```
Out[156]:
```

| | tweet_id | rating_numerator | rating_denominator | name | date |
|---|---|---|---|---|---|
| **timestamp** | | | | | |
| **2016-05-13 16:15:54** | 731156023742988288 | 204.0 | 170.0 | NaN | 13-May-2016 |

```
In [157]:
```

```
df_master[df_master['favorite_count'] == df_master['favorite_count'].max()]
```

```
Out[157]:
```

| | tweet_id | rating_numerator | rating_denominator | name | date |
|---|---|---|---|---|---|
| **timestamp** | | | | | |
| **2016-12-09 06:17:20** | 807106840509214720 | 13.0 | 10.0 | Stephan | 09-Dec 201 |

```
In [158]:
```

```
df_master[df_master['favorite_count'] == df_master['favorite_count'].min()]
```

```
Out[158]:
```

| | tweet_id | rating_numerator | rating_denominator | name | date |
|---|---|---|---|---|---|
| **timestamp** | | | | | |
| **2015-11-16 03:55:04** | 666102155909144576 | 11.0 | 10.0 | NaN | 16-Nov-2015 |

In [159]:

```
df_master[df_master['retweet_count'] == df_master['retweet_count'].max()]
```

Out[159]:

| | tweet_id | rating_numerator | rating_denominator | name | date |
|---|---|---|---|---|---|
| **timestamp** | | | | | |
| **2016-12-09 06:17:20** | 807106840509214720 | 13.0 | 10.0 | Stephan | 09-Dec 201 |

In [160]:

```
df_master[df_master['retweet_count'] == df_master['retweet_count'].min()]
```

Out[160]:

| | tweet_id | rating_numerator | rating_denominator | name | date | |
|---|---|---|---|---|---|---|
| **timestamp** | | | | | | |
| **2015-11-16 03:55:04** | 666102155909144576 | 11.0 | 10.0 | NaN | 16-Nov-2015 | |

In [161]:

```
df_master[df_master['rating'] == df_master['rating'].max()]
```

Out[161]:

| | tweet_id | rating_numerator | rating_denominator | name | date |
|---|---|---|---|---|---|
| **timestamp** | | | | | |
| **2016-07-04 15:00:45** | 749981277374128128 | 1776.0 | 10.0 | Atticus | 04-Jul-2016 |

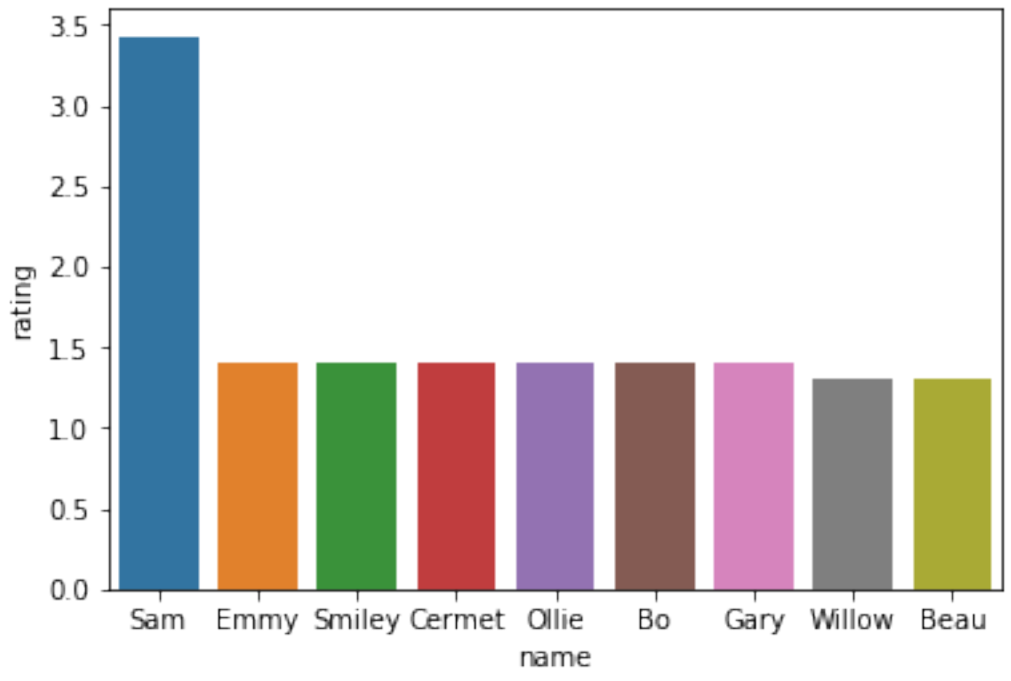
## Top 10 rating dog names

In [162]:

```
# firstly, exclude rows in which dog names are missing
# then select the rows of which contain the top 10 rating dog names, but we don'
t take the outlier into consideration
# this time
top10_rating = df_master.query('name == name').sort_values('rating', ascending =
False)[1:10]
```

In [163]:

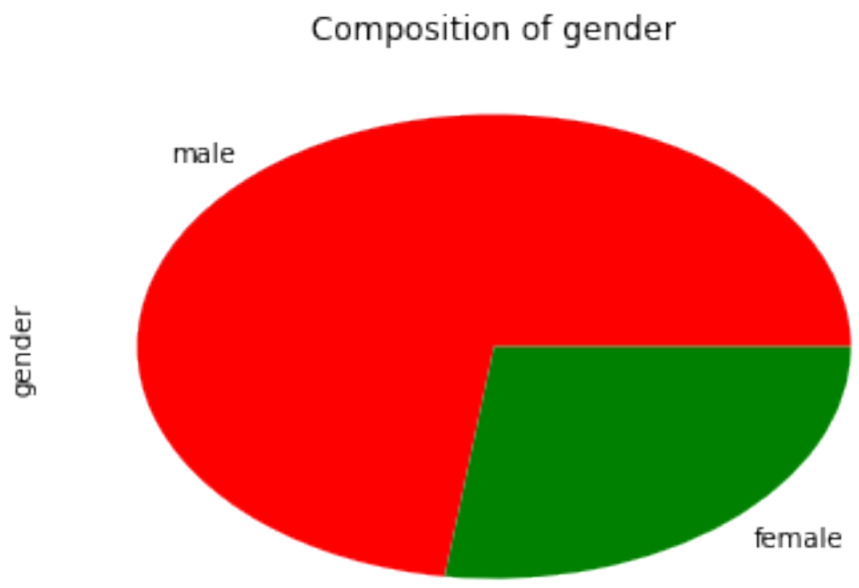
```
sns.barplot(x="name", y="rating", data=top10_rating);
```



## Gender composition

In [164]:

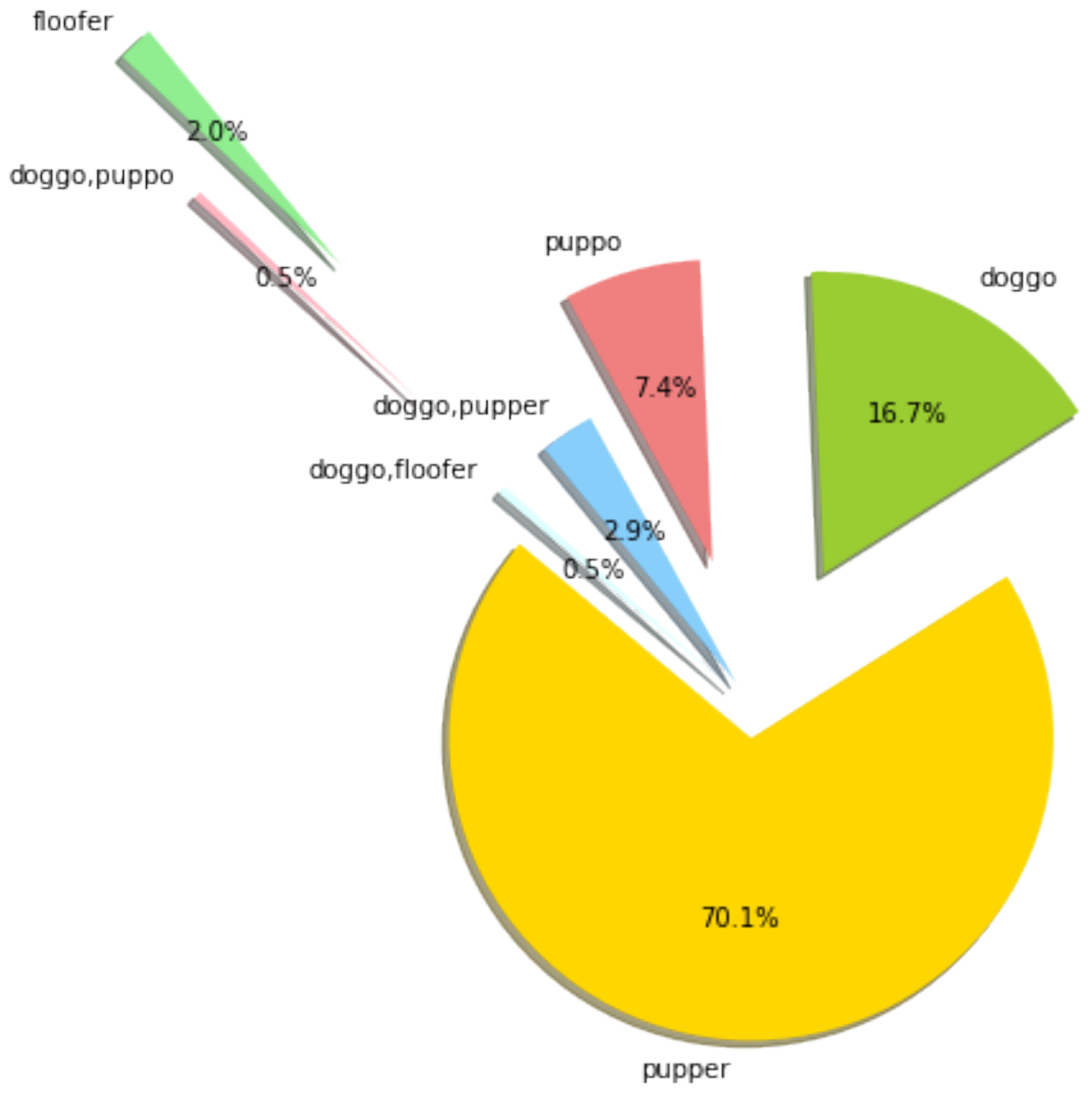
```
# exclude NA value first and then visualize the gender composition
df_master[df_master.gender.notnull()].gender.value_counts().plot(kind = 'pie', c
olors=tuple(["r", "g"]));
plt.title('Composition of gender');
```



## Stage composition

In [165]:

```
# exclude NA value first and then visualize the stage composition
labels = df_master[df_master.stage.notnull()].stage.value_counts().index.tolist(
)
sizes = df_master[df_master.stage.notnull()].stage.value_counts().tolist()

colors = ['gold', 'yellowgreen', 'lightcoral', 'lightskyblue', 'lightgreen', 'li
ghtpink', 'lightcyan']
#cs=cm.Set1(np.arange(40)/40.)
explode = (0.1, 0.5, 0.5, 0.1, 2, 1.5, 0.1)  # explode 1st slice


# Plot
plt.figure(figsize=(8, 8))
matplotlib.rcParams['font.size'] = 10
plt.pie(sizes, explode=explode, labels=labels, colors=colors,
        autopct='%1.1f%%', shadow=True, startangle=140)

plt.axis('equal')
plt.show()
```

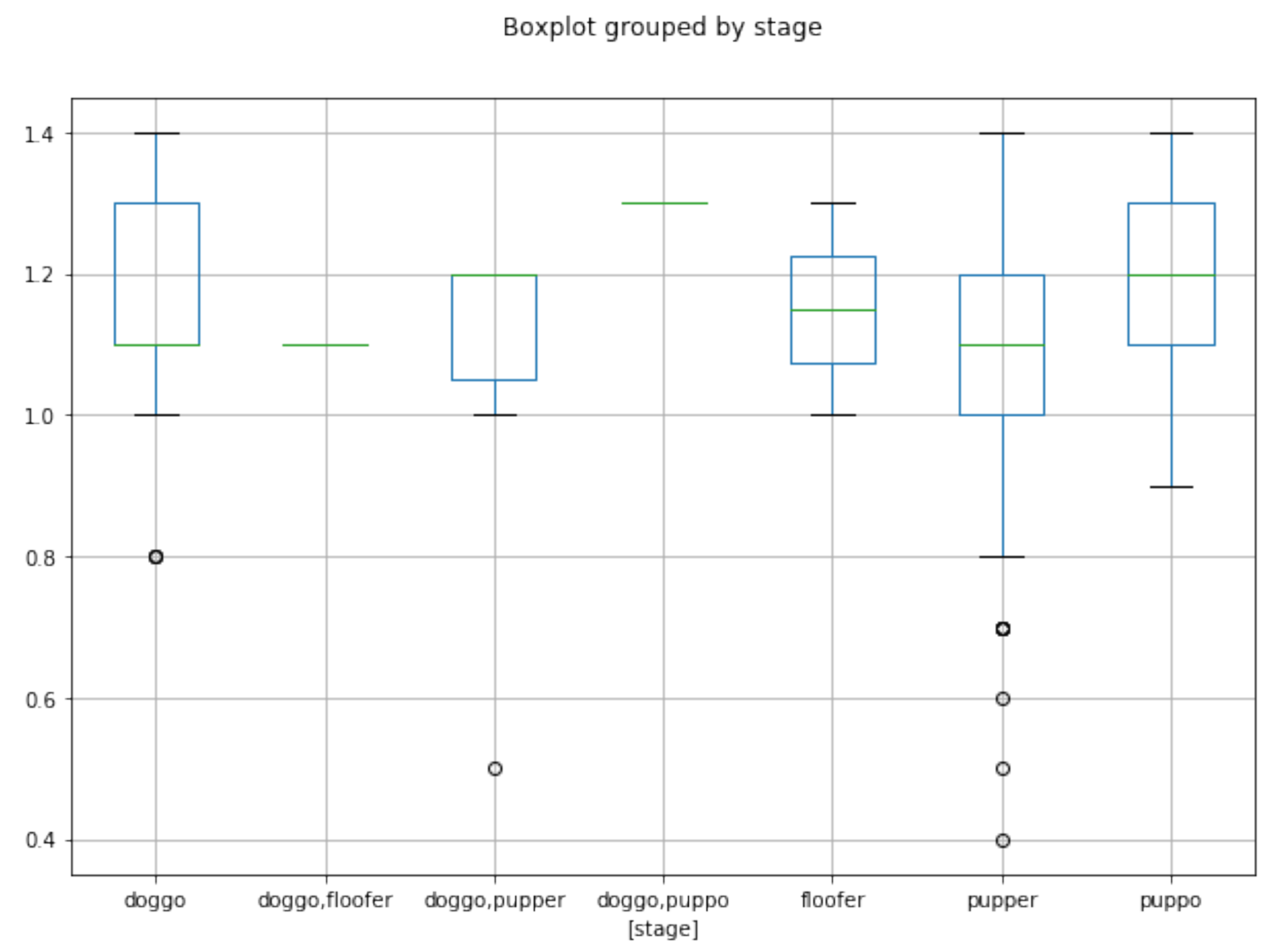


## Boxplot of stage with ratings

In [179]:

```
# Plot the dog stages with ratings
df_master[df_master['stage'].notnull()].boxplot(column = ['rating'], by = ['stag
e'], figsize=(10, 7))
plt.title('');
```

```
/Users/shilinli/anaconda3/lib/python3.6/site-packages/numpy/core/fro
mnumeric.py:57: FutureWarning: reshape is deprecated and will raise
in a subsequent release. Please use .values.reshape(...) instead
  return getattr(obj, method)(*args, **kwds)
```

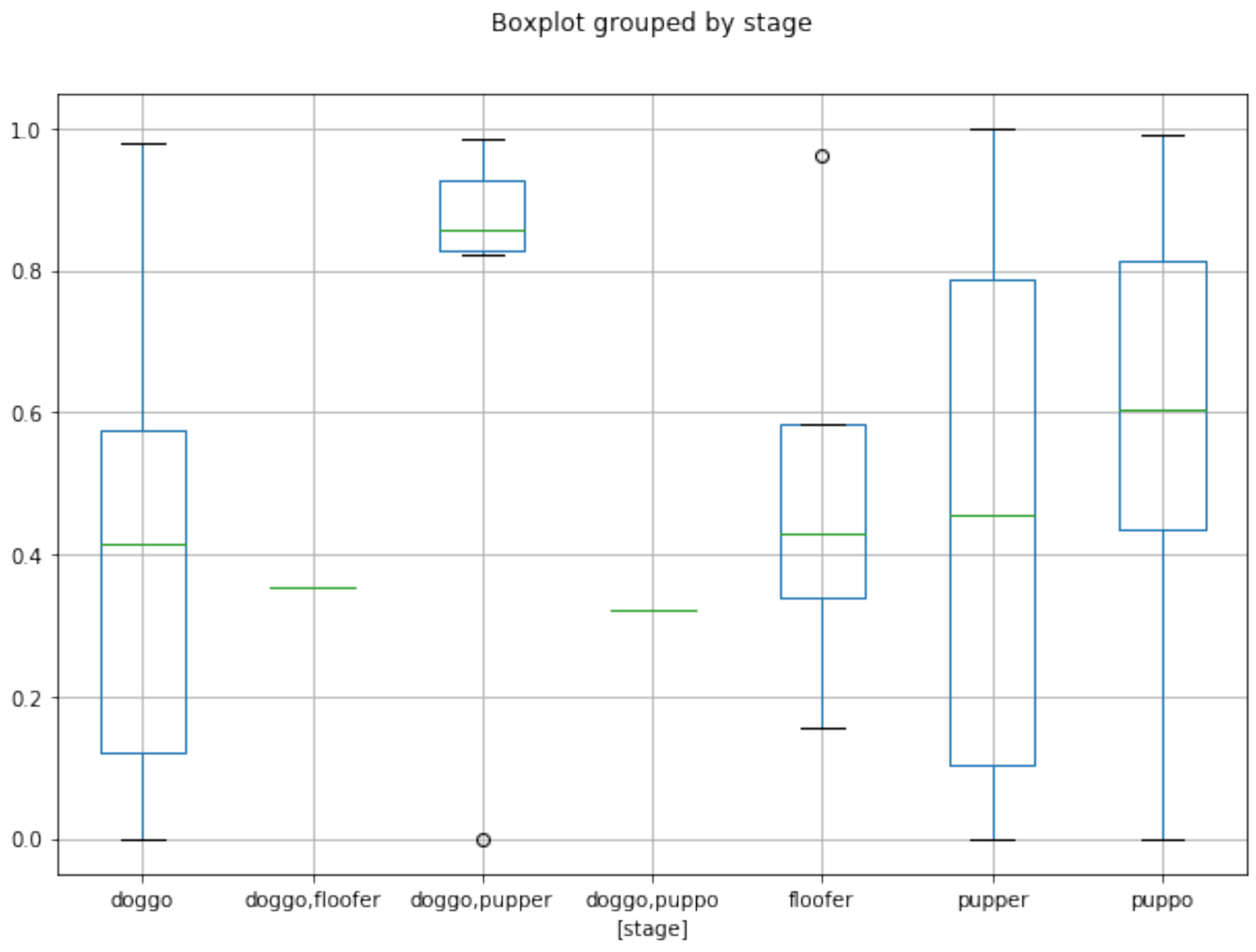


## Stage VS prediction odds

```
In [181]:
```

```
df_master[df_master['stage'].notnull()].boxplot(column = ['odds'], by = ['stage'
], figsize=(10, 7))
plt.title('');
```

```
/Users/shilinli/anaconda3/lib/python3.6/site-packages/numpy/core/fro
mnumeric.py:57: FutureWarning: reshape is deprecated and will raise
in a subsequent release. Please use .values.reshape(...) instead
  return getattr(obj, method)(*args, **kwds)
```



In [168]:

```
Image(filename="Atticus.jpg")
```

Out[168]:

The above image is adapted from here (https://pbs.twimg.com/media/CmgBZ7kWcAAlzFD.jpg).