wrangle_act

September 10, 2021

1 Project: Wrangling and Analyze Data

```
In [1]: import pandas as pd
        import numpy as np
        import requests
        import matplotlib.pyplot as plt
        import seaborn as sb
```

1.1 Data Gathering

In the cell below, gather **all** three pieces of data for this project and load them in the notebook. **Note:** the methods required to gather each data are different. 1. Directly download the WeRateDogs Twitter archive data (twitter_archive_enhanced.csv)

```
In [2]: #Download twitter-archive-enhanced using read_csv pandas's method
        twitter_archive = pd.read_csv('twitter-archive-enhanced.csv')
        #View the first couple of rows
        twitter_archive.head()
```

```
Out[2]:              tweet_id  in_reply_to_status_id  in_reply_to_user_id  \
        0  892420643555336193                    NaN                  NaN
        1  892177421306343426                    NaN                  NaN
        2  891815181378084864                    NaN                  NaN
        3  891689557279858688                    NaN                  NaN
        4  891327558926688256                    NaN                  NaN

                          timestamp  \
        0  2017-08-01 16:23:56 +0000
        1  2017-08-01 00:17:27 +0000
        2  2017-07-31 00:18:03 +0000
        3  2017-07-30 15:58:51 +0000
        4  2017-07-29 16:00:24 +0000

                                                      source  \
        0  <a href="http://twitter.com/download/iphone" r...
        1  <a href="http://twitter.com/download/iphone" r...
        2  <a href="http://twitter.com/download/iphone" r...
```

```
3  <a href="http://twitter.com/download/iphone" r...
4  <a href="http://twitter.com/download/iphone" r...

                                               text  retweeted_status_id  \
0  This is Phineas. He's a mystical boy. Only eve...                  NaN
1  This is Tilly. She's just checking pup on you...                  NaN
2  This is Archie. He is a rare Norwegian Pouncin...                  NaN
3  This is Darla. She commenced a snooze mid meal...                  NaN
4  This is Franklin. He would like you to stop ca...                  NaN

   retweeted_status_user_id retweeted_status_timestamp  \
0                       NaN                        NaN
1                       NaN                        NaN
2                       NaN                        NaN
3                       NaN                        NaN
4                       NaN                        NaN

                                       expanded_urls  rating_numerator  \
0  https://twitter.com/dog_rates/status/892420643...                13
1  https://twitter.com/dog_rates/status/892177421...                13
2  https://twitter.com/dog_rates/status/891815181...                12
3  https://twitter.com/dog_rates/status/891689557...                13
4  https://twitter.com/dog_rates/status/891327558...                12

   rating_denominator      name doggo floofer pupper puppo
0                  10   Phineas  None    None   None  None
1                  10     Tilly  None    None   None  None
2                  10    Archie  None    None   None  None
3                  10     Darla  None    None   None  None
4                  10  Franklin  None    None   None  None
```

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

```
In [3]: #first save the url link
        url = "https://d17h27t6h515a5.cloudfront.net/topher/2017/August/599fd2ad_image-predictio
        response = requests.get(url)
        with open('image_predictions.tsv', 'wb') as file:
            file.write(response.content)
        image_predictions = pd.read_csv('image_predictions.tsv', sep='\t')
        #View the first couple of lines in image_predictions data
        image_predictions.head()
```

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

```
   img_num                      p1   p1_conf  p1_dog                  p2  \
0        1  Welsh_springer_spaniel  0.465074    True              collie   
1        1                 redbone  0.506826    True  miniature_pinscher   
2        1         German_shepherd  0.596461    True            malinois   
3        1     Rhodesian_ridgeback  0.408143    True             redbone   
4        1      miniature_pinscher  0.560311    True          Rottweiler   

    p2_conf  p2_dog                   p3   p3_conf  p3_dog  
0  0.156665    True    Shetland_sheepdog  0.061428    True  
1  0.074192    True  Rhodesian_ridgeback  0.072010    True  
2  0.138584    True           bloodhound  0.116197    True  
3  0.360687    True   miniature_pinscher  0.222752    True  
4  0.243682    True             Doberman  0.154629    True  
```

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

```
In [4]: #Reading the json file by using read_json pandas's method
        tweet_json = pd.read_json('tweet-json.txt',lines=True)
        #View the first 4 rows
        tweet_json.head(4)
```

```
Out[4]:   contributors  coordinates          created_at display_text_range  \
        0          NaN          NaN 2017-08-01 16:23:56            [0, 85]   
        1          NaN          NaN 2017-08-01 00:17:27           [0, 138]   
        2          NaN          NaN 2017-07-31 00:18:03           [0, 121]   
        3          NaN          NaN 2017-07-30 15:58:51            [0, 79]   

                                                    entities  \
        0  {'hashtags': [], 'symbols': [], 'user_mentions...   
        1  {'hashtags': [], 'symbols': [], 'user_mentions...   
        2  {'hashtags': [], 'symbols': [], 'user_mentions...   
        3  {'hashtags': [], 'symbols': [], 'user_mentions...   

                                           extended_entities  favorite_count  \
        0  {'media': [{'id': 892420639486877696, 'id_str'...           39467   
        1  {'media': [{'id': 892177413194625024, 'id_str'...           33819   
        2  {'media': [{'id': 891815175371796480, 'id_str'...           25461   
        3  {'media': [{'id': 891689552724799489, 'id_str'...           42908   

           favorited                                          full_text  geo  \
        0      False  This is Phineas. He's a mystical boy. Only eve...  NaN   
        1      False  This is Tilly. She's just checking pup on you...   NaN   
        2      False  This is Archie. He is a rare Norwegian Pouncin...  NaN   
        3      False  This is Darla. She commenced a snooze mid meal...  NaN   

                                ...                           \
        0                       ...                            
```

```
1                        ...
2                        ...
3                        ...

   possibly_sensitive_appealable  quoted_status quoted_status_id  \
0                            0.0            NaN              NaN
1                            0.0            NaN              NaN
2                            0.0            NaN              NaN
3                            0.0            NaN              NaN

   quoted_status_id_str  retweet_count  retweeted  retweeted_status  \
0                   NaN           8853      False               NaN
1                   NaN           6514      False               NaN
2                   NaN           4328      False               NaN
3                   NaN           8964      False               NaN

                                              source truncated  \
0  <a href="http://twitter.com/download/iphone" r...     False
1  <a href="http://twitter.com/download/iphone" r...     False
2  <a href="http://twitter.com/download/iphone" r...     False
3  <a href="http://twitter.com/download/iphone" r...     False

                                                user
0  {'id': 4196983835, 'id_str': '4196983835', 'na...
1  {'id': 4196983835, 'id_str': '4196983835', 'na...
2  {'id': 4196983835, 'id_str': '4196983835', 'na...
3  {'id': 4196983835, 'id_str': '4196983835', 'na...

[4 rows x 31 columns]
```

1.2 Assessing Data

In this section, detect and document at least **eight (8) quality issues and two (2) tidiness issue**. You must use **both** visual assessment programmatic assessement to assess the data.

Note: pay attention to the following key points when you access the data.

- You only want original ratings (no retweets) that have images. Though there are 5000+ tweets in the dataset, not all are dog ratings and some are retweets.
- Assessing and cleaning the entire dataset completely would require a lot of time, and is not necessary to practice and demonstrate your skills in data wrangling. Therefore, the requirements of this project are only to assess and clean at least 8 quality issues and at least 2 tidiness issues in this dataset.
- The fact that the rating numerators are greater than the denominators does not need to be cleaned. This unique rating system is a big part of the popularity of WeRateDogs.
- You do not need to gather the tweets beyond August 1st, 2017. You can, but note that you won't be able to gather the image predictions for these tweets since you don't have access to the algorithm used.

1.2.1 Quality issues

1.**twitter_archive:** timestamp as object (string), needs to be converted to DateTime datatype.
2.**twitter_archive:** tweet_id as int64, needs to be converted to String datatype.
3.**twitter_archive:** deals with records that has a denomiator higher than 10.
4.**twitter_archive:** data in source column has a href html tag, needs to be fixed.
5.**twitter_archive:** delete all retweeted tweets 'duplicate tweets'.

1.2.2 Tidiness issues

1.**twitter_archive:** doggo, floofer, pupper and puppo needs to be in one column rather than 4."Each variable is a column"

2.**twitter_archive:** remove unnecessary columns(in_reply_to_status_id, in_reply_to_user_id, retweeted_status_id, retweeted_status_user_id, retweeted_status_timestamp)

1.3 Assesing | | twitter_archive dataset

```
In [5]: #check the datatype of twitter archive df
        twitter_archive.info()
```

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

```
In [6]: #View all record that has a denomiator higher than 10 which not right according to the d
        twitter_archive.rating_denominator.value_counts()
        #As we can see the is a couple of records that has a denomiator higher, will try to
        #fix it or delete it if neccassery
```

```
Out[6]: 10     2333
        11        3
        50        3
        80        2
        20        2
        2         1
        16        1
        40        1
        70        1
        15        1
        90        1
        110       1
        120       1
        130       1
        150       1
        170       1
        7         1
        0         1
        Name: rating_denominator, dtype: int64
```

```
In [7]: #query tweets with denominator higher that 10.
        high_deno= twitter_archive.query('rating_denominator >10')
        high_deno
```

```
Out[7]:                tweet_id  in_reply_to_status_id  in_reply_to_user_id  \
        342   832088576586297345           8.320875e+17         3.058208e+07
        433   820690176645140481                    NaN                  NaN
        784   775096608509886464                    NaN                  NaN
        902   758467244762497024                    NaN                  NaN
        1068  740373189193256964                    NaN                  NaN
        1120  731156023742988288                    NaN                  NaN
        1165  722974582966214656                    NaN                  NaN
        1202  716439118184652801                    NaN                  NaN
        1228  713900603437621249                    NaN                  NaN
        1254  710658690886586372                    NaN                  NaN
        1274  709198395643068416                    NaN                  NaN
        1351  704054845121142784                    NaN                  NaN
        1433  697463031882764288                    NaN                  NaN
        1598  686035780142297088           6.860340e+17         4.196984e+09
        1634  684225744407494656           6.842229e+17         4.196984e+09
        1635  684222868335505415                    NaN                  NaN
        1662  682962037429899265                    NaN                  NaN
        1663  682808988178739200           6.827884e+17         4.196984e+09
        1779  677716515794329600                    NaN                  NaN
        1843  675853064436391936                    NaN                  NaN

                              timestamp  \
        342   2017-02-16 04:45:50 +0000
```

```
433   2017-01-15 17:52:40 +0000
784   2016-09-11 22:20:06 +0000
902   2016-07-28 01:00:57 +0000
1068  2016-06-08 02:41:38 +0000
1120  2016-05-13 16:15:54 +0000
1165  2016-04-21 02:25:47 +0000
1202  2016-04-03 01:36:11 +0000
1228  2016-03-27 01:29:02 +0000
1254  2016-03-18 02:46:49 +0000
1274  2016-03-14 02:04:08 +0000
1351  2016-02-28 21:25:30 +0000
1433  2016-02-10 16:51:59 +0000
1598  2016-01-10 04:04:10 +0000
1634  2016-01-05 04:11:44 +0000
1635  2016-01-05 04:00:18 +0000
1662  2016-01-01 16:30:13 +0000
1663  2016-01-01 06:22:03 +0000
1779  2015-12-18 05:06:23 +0000
1843  2015-12-13 01:41:41 +0000

                                                 source  \
342   <a href="http://twitter.com/download/iphone" r...
433   <a href="http://twitter.com/download/iphone" r...
784   <a href="http://twitter.com/download/iphone" r...
902   <a href="http://twitter.com/download/iphone" r...
1068  <a href="http://twitter.com/download/iphone" r...
1120  <a href="http://twitter.com/download/iphone" r...
1165  <a href="http://twitter.com/download/iphone" r...
1202  <a href="http://twitter.com/download/iphone" r...
1228  <a href="http://twitter.com/download/iphone" r...
1254  <a href="http://twitter.com/download/iphone" r...
1274  <a href="http://twitter.com/download/iphone" r...
1351  <a href="http://twitter.com/download/iphone" r...
1433  <a href="http://twitter.com/download/iphone" r...
1598  <a href="http://twitter.com/download/iphone" r...
1634  <a href="http://twitter.com/download/iphone" r...
1635  <a href="http://twitter.com/download/iphone" r...
1662  <a href="http://twitter.com/download/iphone" r...
1663  <a href="http://twitter.com/download/iphone" r...
1779  <a href="http://twitter.com/download/iphone" r...
1843  <a href="http://twitter.com/download/iphone" r...

                                                   text  retweeted_status_id  \
342            @docmisterio account started on 11/15/15                  NaN
433   The floofs have been released I repeat the flo...                  NaN
784   RT @dog_rates: After so many requests, this is...         7.403732e+17
902   Why does this never happen at my front door...                  NaN
1068  After so many requests, this is Bretagne. She ...                  NaN
```

```
1120  Say hello to this unbelievably well behaved sq...                  NaN
1165  Happy 4/20 from the squad! 13/10 for all https...                  NaN
1202  This is Bluebert. He just saw that both #Final...                  NaN
1228  Happy Saturday here's 9 puppers on a bench. 99...                  NaN
1254  Here's a brigade of puppers. All look very pre...                  NaN
1274  From left to right:\nCletus, Jerome, Alejandro...                  NaN
1351  Here is a whole flock of puppers.  60/50 I'll ...                  NaN
1433  Happy Wednesday here's a bucket of pups. 44/40...                  NaN
1598  Yes I do realize a rating of 4/20 would've bee...                  NaN
1634  Two sneaky puppers were not initially seen, mo...                  NaN
1635  Someone help the girl is being mugged. Several...                  NaN
1662  This is Darrel. He just robbed a 7/11 and is i...                  NaN
1663  I'm aware that I could've said 20/16, but here...                  NaN
1779  IT'S PUPPERGEDDON. Total of 144/120 ...I think...                  NaN
1843  Here we have an entire platoon of puppers. Tot...                  NaN

      retweeted_status_user_id retweeted_status_timestamp  \
342                        NaN                        NaN
433                        NaN                        NaN
784               4.196984e+09  2016-06-08 02:41:38 +0000
902                        NaN                        NaN
1068                       NaN                        NaN
1120                       NaN                        NaN
1165                       NaN                        NaN
1202                       NaN                        NaN
1228                       NaN                        NaN
1254                       NaN                        NaN
1274                       NaN                        NaN
1351                       NaN                        NaN
1433                       NaN                        NaN
1598                       NaN                        NaN
1634                       NaN                        NaN
1635                       NaN                        NaN
1662                       NaN                        NaN
1663                       NaN                        NaN
1779                       NaN                        NaN
1843                       NaN                        NaN

                                          expanded_urls  rating_numerator  \
342                                                 NaN                11
433   https://twitter.com/dog_rates/status/820690176...                84
784   https://twitter.com/dog_rates/status/740373189...                 9
902   https://twitter.com/dog_rates/status/758467244...               165
1068  https://twitter.com/dog_rates/status/740373189...                 9
1120  https://twitter.com/dog_rates/status/731156023...               204
1165  https://twitter.com/dog_rates/status/722974582...                 4
1202  https://twitter.com/dog_rates/status/716439118...                50
1228  https://twitter.com/dog_rates/status/713900603...                99
```

```
1254  https://twitter.com/dog_rates/status/710658690...                80
1274  https://twitter.com/dog_rates/status/709198395...                45
1351  https://twitter.com/dog_rates/status/704054845...                60
1433  https://twitter.com/dog_rates/status/697463031...                44
1598                                                NaN                 4
1634  https://twitter.com/dog_rates/status/684225744...               143
1635  https://twitter.com/dog_rates/status/684222868...               121
1662  https://twitter.com/dog_rates/status/682962037...                 7
1663                                                NaN                20
1779  https://twitter.com/dog_rates/status/677716515...               144
1843  https://twitter.com/dog_rates/status/675853064...                88

      rating_denominator      name doggo floofer pupper puppo
342                   15      None  None    None   None  None
433                   70      None  None    None   None  None
784                   11      None  None    None   None  None
902                  150      None  None    None   None  None
1068                  11      None  None    None   None  None
1120                 170      this  None    None   None  None
1165                  20      None  None    None   None  None
1202                  50  Bluebert  None    None   None  None
1228                  90      None  None    None   None  None
1254                  80      None  None    None   None  None
1274                  50      None  None    None   None  None
1351                  50         a  None    None   None  None
1433                  40      None  None    None   None  None
1598                  20      None  None    None   None  None
1634                 130      None  None    None   None  None
1635                 110      None  None    None   None  None
1662                  11    Darrel  None    None   None  None
1663                  16      None  None    None   None  None
1779                 120      None  None    None   None  None
1843                  80      None  None    None   None  None
```

```
In [8]: tweet_id= high_deno['tweet_id']
        text= high_deno['text']

        for point in zip(tweet_id, text):
            print("tweet Id:{} \n text: {} \n ----------------".format(*point))
```

```
tweet Id:832088576586297345
 text: @docmisterio account started on 11/15/15
 ----------------
tweet Id:820690176645140481
 text: The floofs have been released I repeat the floofs have been released. 84/70 https://t.co/
 ----------------
tweet Id:775096608509886464
 text: RT @dog_rates: After so many requests, this is Bretagne. She was the last surviving 9/11
 ----------------
tweet Id:758467244762497024
 text: Why does this never happen at my front door... 165/150 https://t.co/HmwrdfEfUE
 ----------------
tweet Id:740373189193256964
 text: After so many requests, this is Bretagne. She was the last surviving 9/11 search dog, and
 ----------------
tweet Id:731156023742988288
 text: Say hello to this unbelievably well behaved squad of doggos. 204/170 would try to pet all
 ----------------
tweet Id:722974582966214656
 text: Happy 4/20 from the squad! 13/10 for all https://t.co/eV1diwds8a
 ----------------
tweet Id:716439118184652801
 text: This is Bluebert. He just saw that both #FinalFur match ups are split 50/50. Amazed af. 1
 ----------------
tweet Id:713900603437621249
 text: Happy Saturday here's 9 puppers on a bench. 99/90 good work everybody https://t.co/mpvaVx
 ----------------
tweet Id:710658690886586372
 text: Here's a brigade of puppers. All look very prepared for whatever happens next. 80/80 http
 ----------------
tweet Id:709198395643068416
 text: From left to right:
Cletus, Jerome, Alejandro, Burp, &amp; Titson
None know where camera is. 45/50 would hug all at once https://t.co/sedre1ivTK
 ----------------
tweet Id:704054845121142784
 text: Here is a whole flock of puppers.  60/50 I'll take the lot https://t.co/9dpcw6MdWa
 ----------------
tweet Id:697463031882764288
 text: Happy Wednesday here's a bucket of pups. 44/40 would pet all at once https://t.co/HppvrYu
 ----------------
tweet Id:686035780142297088
 text: Yes I do realize a rating of 4/20 would've been fitting. However, it would be unjust to g
 ----------------
tweet Id:684225744407494656
 text: Two sneaky puppers were not initially seen, moving the rating to 143/130. Please forgive
 ----------------
tweet Id:684222868335505415
 text: Someone help the girl is being mugged. Several are distracting her while two steal her sh
 ----------------
tweet Id:682962037429899265
 text: This is Darrel. He just robbed a 7/11 and is in a high speed police chase. Was just spott
 ----------------
tweet Id:682808988178739200
 text: I'm aware that I could've said 20/16, but here at WeRateDogs we are very professional. An
 ----------------
```

```
tweet Id:677716515794329600
 text: IT'S PUPPERGEDDON. Total of 144/120 ...I think https://t.co/ZanVtAtvIq
 ----------------
tweet Id:675853064436391936
 text: Here we have an entire platoon of puppers. Total score: 88/80 would pet all at once https
 ----------------
```

- **Tweet Id** 832088576586297345: This tweet needs to be deleted, no rating provided.
- **Tweet Id** 820690176645140481: Wrong rating provided, needs to be deleted.
- **Tweet Id** 775096608509886464: I've noticed that this a retweeted tweet for tweet with index **740373189193256964**, need to delete all retweeted tweets since its a duplicate ones.
- **Tweet Id** 758467244762497024: Wrong rating provided, needs to be deleted.
- **Tweet Id** 740373189193256964: Wrong captured data from tweet, actual rating is 14/10.
- **Tweet Id** 731156023742988288: Wrong rating provided, needs to be deleted.
- **Tweet Id** 722974582966214656: Wrong data captured, actual rating is 13/10.
- **Tweet Id** 716439118184652801: Wrong data captured, actual rating is 11/10.
- **Tweet Id** 713900603437621249: Wrong rating provided, needs to be deleted.
- **Tweet Id** 710658690886586372: rating can be change to 10/10 since it same as 80/80.
- **Tweet Id** 709198395643068416: Wrong rating provided, needs to be deleted.
- **Tweet Id** 704054845121142784: Wrong rating provided, needs to be deleted.
- **Tweet Id** 697463031882764288: Wrong rating provided, needs to be deleted.
- **Tweet Id** 686035780142297088: Wrong rating provided, needs to be deleted.
- **Tweet Id** 684225744407494656: Wrong rating provided, needs to be deleted.
- **Tweet Id** 684222868335505415: Wrong rating provided, needs to be deleted.
- **Tweet Id** 682962037429899265:tweet isn't clear for me, I guess the acutal rating is 10/10 not 7/11
- **Tweet Id** 682808988178739200:Wrong rating provided, needs to be deleted.
- **Tweet Id** 677716515794329600:Wrong rating provided, needs to be deleted.
- **Tweet Id** 675853064436391936:Wrong rating provided, needs to be deleted.

```
In [9]: #Checkif there is any duplicated values
        twitter_archive.duplicated().sum()
```

```
Out[9]: 0
```

1.4 Assesing2 || image_predictions dataset

1.4.1 Quality issues

1.**image_predictions:** Remove duplicate jpg_url.
2.**image_predicitons:** Change datatype of tweet_id column

1.4.2 Tidness issues

1.**image_prediction:** extract the breed of dog from the p,p_conf and p_dog columns.

```
In [10]: #check datatypes of columns
         image_predictions.info()
```

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

```
In [11]: #check for duplicate reocrds.
         image_predictions.duplicated().sum()
```

```
Out[11]: 0
```

```
In [12]: #check for duplicate jpg_url since the prediction is based on it.
         image_predictions['jpg_url'].duplicated().sum()
         #there is 66 duplicate photo that has to be deleted.
```

```
Out[12]: 66
```

```
In [13]: image_predictions.describe()
```

```
Out[13]:            tweet_id      img_num      p1_conf       p2_conf       p3_conf
         count  2.075000e+03  2075.000000  2075.000000  2.075000e+03  2.075000e+03
         mean   7.384514e+17     1.203855     0.594548  1.345886e-01  6.032417e-02
         std    6.785203e+16     0.561875     0.271174  1.006657e-01  5.090593e-02
         min    6.660209e+17     1.000000     0.044333  1.011300e-08  1.740170e-10
         25%    6.764835e+17     1.000000     0.364412  5.388625e-02  1.622240e-02
         50%    7.119988e+17     1.000000     0.588230  1.181810e-01  4.944380e-02
         75%    7.932034e+17     1.000000     0.843855  1.955655e-01  9.180755e-02
         max    8.924206e+17     4.000000     1.000000  4.880140e-01  2.734190e-01
```

1.5 Assesing3 | | tweet_json dataset

1.5.1 Quality issues

1.**tweet_json:** rename "id" column to tweet_id and change datatype to String.
2.**tweet_json:** remove href tag from source column.

1.5.2 Tidness issues

1.**tweet_json:** We only need favorite_count, retweet count,id "tweet_id" (after renamed), and source columns.

```
In [14]: tweet_json.head(5)
```

```
Out[14]:    contributors  coordinates          created_at display_text_range  \
         0           NaN          NaN 2017-08-01 16:23:56            [0, 85]
         1           NaN          NaN 2017-08-01 00:17:27           [0, 138]
         2           NaN          NaN 2017-07-31 00:18:03           [0, 121]
         3           NaN          NaN 2017-07-30 15:58:51            [0, 79]
         4           NaN          NaN 2017-07-29 16:00:24           [0, 138]

                                                     entities  \
         0  {'hashtags': [], 'symbols': [], 'user_mentions...
         1  {'hashtags': [], 'symbols': [], 'user_mentions...
         2  {'hashtags': [], 'symbols': [], 'user_mentions...
         3  {'hashtags': [], 'symbols': [], 'user_mentions...
         4  {'hashtags': [{'text': 'BarkWeek', 'indices': ...

                                            extended_entities  favorite_count  \
         0  {'media': [{'id': 892420639486877696, 'id_str'...           39467
         1  {'media': [{'id': 892177413194625024, 'id_str'...           33819
         2  {'media': [{'id': 891815175371796480, 'id_str'...           25461
         3  {'media': [{'id': 891689552724799489, 'id_str'...           42908
         4  {'media': [{'id': 891327551943041024, 'id_str'...           41048

            favorited                                          full_text  geo  \
         0      False  This is Phineas. He's a mystical boy. Only eve...  NaN
         1      False  This is Tilly. She's just checking pup on you...  NaN
         2      False  This is Archie. He is a rare Norwegian Pouncin...  NaN
         3      False  This is Darla. She commenced a snooze mid meal...  NaN
         4      False  This is Franklin. He would like you to stop ca...  NaN

                                  ...                          \
         0                        ...
         1                        ...
         2                        ...
         3                        ...
         4                        ...

            possibly_sensitive_appealable  quoted_status quoted_status_id  \
         0                            0.0            NaN              NaN
         1                            0.0            NaN              NaN
         2                            0.0            NaN              NaN
         3                            0.0            NaN              NaN
         4                            0.0            NaN              NaN
```

```
  quoted_status_id_str  retweet_count  retweeted  retweeted_status  \
0                  NaN           8853      False               NaN
1                  NaN           6514      False               NaN
2                  NaN           4328      False               NaN
3                  NaN           8964      False               NaN
4                  NaN           9774      False               NaN

                                              source truncated  \
0  <a href="http://twitter.com/download/iphone" r...     False
1  <a href="http://twitter.com/download/iphone" r...     False
2  <a href="http://twitter.com/download/iphone" r...     False
3  <a href="http://twitter.com/download/iphone" r...     False
4  <a href="http://twitter.com/download/iphone" r...     False

                                                user
0  {'id': 4196983835, 'id_str': '4196983835', 'na...
1  {'id': 4196983835, 'id_str': '4196983835', 'na...
2  {'id': 4196983835, 'id_str': '4196983835', 'na...
3  {'id': 4196983835, 'id_str': '4196983835', 'na...
4  {'id': 4196983835, 'id_str': '4196983835', 'na...

[5 rows x 31 columns]
```

```
In [15]: tweet_json.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2354 entries, 0 to 2353
Data columns (total 31 columns):
contributors                     0 non-null float64
coordinates                      0 non-null float64
created_at                       2354 non-null datetime64[ns]
display_text_range               2354 non-null object
entities                         2354 non-null object
extended_entities                2073 non-null object
favorite_count                   2354 non-null int64
favorited                        2354 non-null bool
full_text                        2354 non-null object
geo                              0 non-null float64
id                               2354 non-null int64
id_str                           2354 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                  2354 non-null bool
lang                             2354 non-null object
place                            1 non-null object
```

```
possibly_sensitive                 2211 non-null float64
possibly_sensitive_appealable      2211 non-null float64
quoted_status                      28 non-null object
quoted_status_id                   29 non-null float64
quoted_status_id_str               29 non-null float64
retweet_count                      2354 non-null int64
retweeted                          2354 non-null bool
retweeted_status                   179 non-null object
source                             2354 non-null object
truncated                          2354 non-null bool
user                               2354 non-null object
dtypes: bool(4), datetime64[ns](1), float64(11), int64(4), object(11)
memory usage: 505.8+ KB
```

```
In [16]: #check for duplicate tweets.
         tweet_json['id'].duplicated().sum()
         #no duplicate id.

Out[16]: 0

In [17]: #see the count for all unique values for source column.
         tweet_json['source'].value_counts()

Out[17]: <a href="http://twitter.com/download/iphone" rel="nofollow">Twitter for iPhone</a>
         <a href="http://vine.co" rel="nofollow">Vine - Make a Scene</a>
         <a href="http://twitter.com" rel="nofollow">Twitter Web Client</a>
         <a href="https://about.twitter.com/products/tweetdeck" rel="nofollow">TweetDeck</a>
         Name: source, dtype: int64
```

1.6 Cleaning Data

In this section, clean **all** of the issues you documented while assessing.

Note: Make a copy of the original data before cleaning. Cleaning includes merging individual pieces of data according to the rules of tidy data. The result should be a high-quality and tidy master pandas DataFrame (or DataFrames, if appropriate).

1.6.1 Quality issues

1.**twitter_archive:** timestamp as object (string), needs to be converted to DateTime datatype.
2.**twitter_archive:** tweet_id as int64, needs to be converted to String datatype.
3.**twitter_archive:** delete all retweeted tweets 'duplicate tweets'.
4.**twitter_archive:** data in source column has a href html tag, needs to fixed.
5.**twitter_archive:** deals with records that has a denominator higher than 10.
6.**image_predictions:** Remove duplicates jpg_url.
7.**image_predicitons:** Change datatype of tweet_id column to String.
8.**tweet_json:** rename "id" column to tweet_id and change datatype to String.

1.6.2 Tidness issues

9.**twitter_archive:** doggo, floofer, pupper and puppo needs to be in one column rather than 4."Each variable is a column"

10.**twitter_archive:** remove unnecessary columns(in_reply_to_status_id, in_reply_to_user_id, retweeted_status_id, retweeted_status_user_id, retweeted_status_timestamp)

11.**image_prediction:** extract the breed of dog from the p,p_conf and p_dog columns.

12.**tweet_json:** We only need favorite_count, retweet count,id "tweet_id" (after renamed), and source columns.

```
In [18]: # Make copies of original pieces of data
         twitter_archive_clean= twitter_archive.copy()
         tweet_json_clean= tweet_json.copy()
         image_predictions_clean= image_predictions.copy()
```

1.7 Cleaning1 | | twitter_archive dataset

1.**twitter_archive:** timestamp as object (string), needs to be converted to DateTime datatype.

2.**twitter_archive:** tweet_id as int64, needs to be converted to String datatype.

3.**twitter_archive:** delete all retweeted tweets 'duplicate tweets'.

4.**twitter_archive:** data in source column has a href html tag, needs to fixed.

5.**twitter_archive:** deals with records that has a denominator higher than 10.

6.**twitter_archive:** doggo, floofer, pupper and puppo needs to be in one column rather than 4."Each variable is a column"

7.**twitter_archive:** remove unnecessary columns(in_reply_to_status_id, in_reply_to_user_id, retweeted_status_id, retweeted_status_user_id, retweeted_status_timestamp).

1.7.1 Issue #1:

timestamp as object (string), needs to be converted to DateTime datatype.

Define: change timestamp datatype from String to datetime by method to_dateTime()

Code

```
In [19]: #converting datatype to datetime
         twitter_archive_clean['timestamp'] = pd.to_datetime(twitter_archive_clean['timestamp'])
```

Test

```
In [20]: #check datatypes
         twitter_archive_clean.dtypes
```

```
Out[20]: tweet_id                                  int64
         in_reply_to_status_id                   float64
         in_reply_to_user_id                     float64
         timestamp                        datetime64[ns]
         source                                object
         text                                  object
         retweeted_status_id                  float64
         retweeted_status_user_id             float64
         retweeted_status_timestamp            object
         expanded_urls                         object
         rating_numerator                       int64
         rating_denominator                     int64
         name                                  object
         doggo                                 object
         floofer                               object
         pupper                                object
         puppo                                 object
         dtype: object
```

1.7.2 Issue #2:

tweet_id as int64, needs to be converted to String datatype.

Define convert tweet_id from int to String by method astype

Code

```
In [21]: #change the datatype to String by using method astype to column 'tweet_id'
         twitter_archive_clean['tweet_id']= twitter_archive_clean['tweet_id'].astype(str)
```

Test

```
In [22]: twitter_archive_clean.dtypes
```

```
Out[22]: tweet_id                              object
         in_reply_to_status_id                float64
         in_reply_to_user_id                  float64
         timestamp                     datetime64[ns]
         source                                object
         text                                  object
         retweeted_status_id                  float64
         retweeted_status_user_id             float64
         retweeted_status_timestamp            object
         expanded_urls                         object
         rating_numerator                       int64
         rating_denominator                     int64
         name                                  object
         doggo                                 object
         floofer                               object
         pupper                                object
         puppo                                 object
         dtype: object
```

1.7.3 Issue #3:

delete all retweeted tweets 'duplicate tweets'.

Define delete retweeted tweets.

Code

```
In [23]: #cleaning the retweeted tweets by selecting rows that have null in reteeted_status_user
         twitter_archive_clean = twitter_archive_clean[pd.isnull(twitter_archive_clean['retweete
```

Test

```
In [24]: #all nan values
         twitter_archive_clean['retweeted_status_user_id'].unique()
         #there is only nan value in retweeted ststus user id

Out[24]: array([ nan])
```

1.7.4 Issue #4:

data in source column has a href html tag, needs to fixed.

Define source is in form of href, needs to be changed to twitter for iPhone, vine, twitter for Web

Code

```
In [25]: #first check all unique values.
         twitter_archive_clean.source.value_counts()

Out[25]: <a href="http://twitter.com/download/iphone" rel="nofollow">Twitter for iPhone</a>
         <a href="http://vine.co" rel="nofollow">Vine - Make a Scene</a>
         <a href="http://twitter.com" rel="nofollow">Twitter Web Client</a>
         <a href="https://about.twitter.com/products/tweetdeck" rel="nofollow">TweetDeck</a>
         Name: source, dtype: int64

In [26]: #by using the method loc to change the value of certain column if a condition is met.
         twitter_archive_clean.loc[twitter_archive_clean['source'].str.contains('iPhone') , 'sou
         twitter_archive_clean.loc[twitter_archive_clean['source'].str.contains('Vine') , 'sourc
         twitter_archive_clean.loc[twitter_archive_clean['source'].str.contains('Web') , 'source
         twitter_archive_clean.loc[twitter_archive_clean['source'].str.contains('TweetDeck') , '
```

Test

```
In [27]: #Now check again to see if it works.
         twitter_archive_clean.source.value_counts()

Out[27]: Twitter for iPhone    2042
         Vine                    91
         Twitter for Web         31
         TweetDeck               11
         Name: source, dtype: int64
```

1.7.5 Issue #5:

deals with records that has a denomiator higher than 10.

- **Tweet Id** 832088576586297345: This tweet needs to be deleted, no rating provided.
- **Tweet Id** 820690176645140481: Wrong rating provided, needs to be deleted.
- **Tweet Id** 775096608509886464: I've noticed that this a retweeted tweet for tweet with Id **740373189193256964**, need to delete all retweeted tweets since its a duplicate ones.
- **Tweet Id** 758467244762497024: Wrong rating provided, needs to be deleted.
- **Tweet Id** 740373189193256964: Wrong captured data from tweet, actual rating is 14/10.
- **Tweet Id** 731156023742988288: Wrong rating provided, needs to be deleted.
- **Tweet Id** 722974582966214656: Wrong data captured, actual rating is 13/10.
- **Tweet Id** 716439118184652801: Wrong data captured, actual rating is 11/10.
- **Tweet Id** 713900603437621249: Wrong rating provided, needs to be deleted.
- **Tweet Id** 710658690886586372: rating can be change to 10/10 since it same as 80/80.
- **Tweet Id** 709198395643068416: Wrong rating provided, needs to be deleted.
- **Tweet Id** 704054845121142784: Wrong rating provided, needs to be deleted.
- **Tweet Id** 697463031882764288: Wrong rating provided, needs to be deleted.
- **Tweet Id** 686035780142297088: Wrong rating provided, needs to be deleted.
- **Tweet Id** 684225744407494656: Wrong rating provided, needs to be deleted.
- **Tweet Id** 684222868335505415: Wrong rating provided, needs to be deleted.
- **Tweet Id** 682962037429899265:tweet isn't clear for me, I guess the acutal rating is 10/10 not 7/11
- **Tweet Id** 682808988178739200:Wrong rating provided, needs to be deleted.
- **Tweet Id** 677716515794329600:Wrong rating provided, needs to be deleted.
- **Tweet Id** 675853064436391936:Wrong rating provided, needs to be deleted.

Define delete all records that has wrong rating and fixed recrods that have captured wrong data from tweets.

Code

```
In [28]: #fixing the remaining row manually: -
         #first delete the rows with wrong ratings
         id_list=[832088576586297345, 820690176645140481, 758467244762497024, 731156023742988288
                  709198395643068416, 704054845121142784, 697463031882764288, 686035780142297088
                  684222868335505415, 682808988178739200, 677716515794329600, 675853064436391936
         for i in id_list:
             twitter_archive_clean=twitter_archive_clean.query('tweet_id !="{}"'.format(i))

         #Now fix recrods that caputred wrong data: -
         twitter_archive_clean.loc[twitter_archive_clean['tweet_id'] =='740373189193256964', 'ra
         twitter_archive_clean.loc[twitter_archive_clean['tweet_id'] =='740373189193256964', 'ra

         twitter_archive_clean.loc[twitter_archive_clean['tweet_id'] =='722974582966214656', 'ra
         twitter_archive_clean.loc[twitter_archive_clean['tweet_id'] =='722974582966214656', 'ra

         twitter_archive_clean.loc[twitter_archive_clean['tweet_id'] =='716439118184652801', 'ra
twitter_archive_clean.loc[twitter_archive_clean['tweet_id'] =='716439118184652801', 'ra

twitter_archive_clean.loc[twitter_archive_clean['tweet_id'] =='710658690886586372', 'ra
twitter_archive_clean.loc[twitter_archive_clean['tweet_id'] =='710658690886586372', 'ra

twitter_archive_clean.loc[twitter_archive_clean['tweet_id'] =='682962037429899265', 'ra
twitter_archive_clean.loc[twitter_archive_clean['tweet_id'] =='682962037429899265', 'ra
```

Test

```
In [29]: len(twitter_archive_clean.query('rating_denominator > 10'))
         #there is 0 records
```

```
Out[29]: 0
```

1.7.6 Issue #6:

doggo, floofer, pupper and puppo needs to be in one column rather than 4."Each variable is a column"

Define these columns represents the stage of dogs, needs to have one column named "dog_stage".

Code

```
In [30]: #create a method that set the value of column 'dog_stage' based on the velue of doggo,
         def stage(row):
             #if doggo has the value 'doggo', dog_stage column for this row is 'doggo'
             if row['doggo'] == 'doggo':
                 val = 'doggo'
             #if floofer has the value 'doggo', dog_stage column for this row is 'floofer'
             elif row['floofer'] == 'floofer':
                 val = 'floofer'
             #if pupper has the value 'doggo', dog_stage column for this row is 'pupper'
             elif row['pupper'] == 'pupper':
                 val = 'pupper'
             #if pippo has the value 'doggo', dog_stage column for this row is 'puppo'
             elif row['puppo']=='puppo':
                 val = 'puppo'
             #if all none, then the value for it is None
             else:
                 val = None
             return val
         twitter_archive_clean['dog_stage'] = twitter_archive_clean.apply(stage, axis=1)
```

Test

```
In [31]: #check if method is successfully done and see if there is a recond that has none in pup
         twitter_archive_clean.query('dog_stage == pupper and pupper == None')

         #Now delete the doggo, floofer, pupper and puppo columns
         twitter_archive_clean= twitter_archive_clean.drop(['doggo', 'floofer', 'pupper', 'puppo
```

1.7.7 Issue #7:

Remove unnecessary columns(in_reply_to_status_id, in_reply_to_user_id, retweeted_status_id, retweeted_status_user_id, retweeted_status_timestamp)

Define drop unnecessary columns in twitter archive dataset

Code

```
In [32]: twitter_archive_clean= twitter_archive_clean.drop(['in_reply_to_status_id', 'in_reply_t
                                                            'retweeted_status_id', 'retweeted_st
                                                            'retweeted_status_timestamp','expand
```

Test

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

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2161 entries, 0 to 2355
Data columns (total 8 columns):
tweet_id              2161 non-null object
timestamp             2161 non-null datetime64[ns]
source                2161 non-null object
text                  2161 non-null object
rating_numerator      2161 non-null int64
rating_denominator    2161 non-null int64
name                  2161 non-null object
dog_stage             344 non-null object
dtypes: datetime64[ns](1), int64(2), object(5)
memory usage: 151.9+ KB
```

1.8 Cleaning 2 | | image_predictions dataset

1.**image_predictions:** Remove duplicate jpg_url.
2.**image_predicitons:** Change datatype of tweet_id column to String.
3.**image_prediction:** extract the breed of dog from the p,p_conf and p_dog columns.

1.8.1 Issue #8:

Remove duplicate jpg_url.

Define remove records that has a duplicate value jpg_url.

Code

```
In [34]: image_predictions_clean= image_predictions_clean.drop_duplicates(subset='jpg_url', keep
         image_predictions_clean.head(3)
```

```
Out[34]:             tweet_id                                          jpg_url  \
         0  666020888022790149  https://pbs.twimg.com/media/CT4udn0WwAA0aMy.jpg
         1  666029285002620928  https://pbs.twimg.com/media/CT42GRgUYAA5iDo.jpg
         2  666033412701032449  https://pbs.twimg.com/media/CT4521TWwAEvMyu.jpg

            img_num                      p1   p1_conf  p1_dog                  p2  \
         0        1  Welsh_springer_spaniel  0.465074    True              collie
         1        1                 redbone  0.506826    True  miniature_pinscher
         2        1         German_shepherd  0.596461    True            malinois

             p2_conf  p2_dog                   p3   p3_conf  p3_dog
         0  0.156665    True   Shetland_sheepdog  0.061428    True
         1  0.074192    True  Rhodesian_ridgeback  0.072010    True
         2  0.138584    True          bloodhound  0.116197    True
```

Test

```
In [35]: #check for duplicate in column jpg_url
         image_predictions_clean.jpg_url.duplicated().sum()
```

```
Out[35]: 0
```

1.8.2 Issue #9:

Change datatype of tweet_id column to String.

Define Change datatype of tweet_id column to Stirng by method astype()

Code

```
In [36]: image_predictions_clean['tweet_id']= image_predictions_clean['tweet_id'].astype(str)
```

Test

```
In [37]: image_predictions_clean.dtypes
```

```
Out[37]: tweet_id     object
         jpg_url      object
         img_num       int64
         p1           object
         p1_conf     float64
         p1_dog         bool
```

```
p2           object
p2_conf     float64
p2_dog         bool
p3           object
p3_conf     float64
p3_dog         bool
dtype: object
```

1.8.3 Issue #10:

extract the breed of dog from the p,p_conf and p_dog columns.

Define extract the breed of dog using a function detect_breed and creating a new column called breed_of_dog

```
In [38]: image_predictions_clean.head(4)
```

```
Out[38]:              tweet_id                                          jpg_url  \
         0  666020888022790149  https://pbs.twimg.com/media/CT4udn0WwAA0aMy.jpg
         1  666029285002620928  https://pbs.twimg.com/media/CT42GRgUYAA5iDo.jpg
         2  666033412701032449  https://pbs.twimg.com/media/CT4521TWwAEvMyu.jpg
         3  666044226329800704  https://pbs.twimg.com/media/CT5Dr8HUEAA-lEu.jpg

            img_num                      p1   p1_conf  p1_dog                  p2  \
         0        1  Welsh_springer_spaniel  0.465074    True              collie
         1        1                 redbone  0.506826    True  miniature_pinscher
         2        1         German_shepherd  0.596461    True            malinois
         3        1     Rhodesian_ridgeback  0.408143    True             redbone

             p2_conf  p2_dog                   p3   p3_conf  p3_dog
         0  0.156665    True    Shetland_sheepdog  0.061428    True
         1  0.074192    True  Rhodesian_ridgeback  0.072010    True
         2  0.138584    True           bloodhound  0.116197    True
         3  0.360687    True   miniature_pinscher  0.222752    True
```

Code

```
In [39]: #I'm only seeing if Pn_dog since the predictions is arranged from the strongest by (pn_
         #checking the Pn_dog is enough.
         def extract_breed (row):
             breed=''
             if row['p1_dog']==True :
                 breed=row['p1']
             elif row['p2_dog']==True :
                  breed=row['p2']
             elif row['p3_dog']==True :
                 breed=row['p3']
             else:
```

```
        breed= None

    return breed
#now I'm calling the function to create the new column: -
image_predictions_clean['breed_of_dog']= image_predictions_clean.apply (lambda row: ext
#drop the the p1,p1_dog,p2_conf....etc

image_predictions_clean= image_predictions_clean.drop(['p1', 'p1_conf','p1_dog','p2','p
                                                       'p2_dog','p3','p3_conf','p3_dog'
```

Test

```
In [40]: image_predictions_clean.head(4)
```

```
Out[40]:              tweet_id                                          jpg_url  \
         0  666020888022790149  https://pbs.twimg.com/media/CT4udn0WwAA0aMy.jpg
         1  666029285002620928  https://pbs.twimg.com/media/CT42GRgUYAA5iDo.jpg
         2  666033412701032449  https://pbs.twimg.com/media/CT4521TWwAEvMyu.jpg
         3  666044226329800704  https://pbs.twimg.com/media/CT5Dr8HUEAA-lEu.jpg

                       breed_of_dog
         0  Welsh_springer_spaniel
         1                 redbone
         2         German_shepherd
         3     Rhodesian_ridgeback
```

1.8.4 Cleaning 3 | | tweet_json dataset

1.**tweet_json:** rename "id" column to tweet_id and change datatype to String.
2.**tweet_json:** We only need favorite_count, retweet count,id "tweet_id" (after renamed), columns.

1.8.5 Issue #11:

rename "id" column to tweet_id and change datatype to String.

Define Rename column id to tweet_id and change its type to String.

Code

```
In [41]: tweet_json_clean.dtypes
```

```
Out[41]: contributors                                      float64
         coordinates                                       float64
         created_at                                 datetime64[ns]
         display_text_range                                 object
         entities                                           object
         extended_entities                         object
         favorite_count                             int64
         favorited                                   bool
         full_text                                 object
         geo                                      float64
         id                                         int64
         id_str                                     int64
         in_reply_to_screen_name                   object
         in_reply_to_status_id                    float64
         in_reply_to_status_id_str                float64
         in_reply_to_user_id                      float64
         in_reply_to_user_id_str                  float64
         is_quote_status                             bool
         lang                                      object
         place                                     object
         possibly_sensitive                       float64
         possibly_sensitive_appealable            float64
         quoted_status                             object
         quoted_status_id                         float64
         quoted_status_id_str                     float64
         retweet_count                              int64
         retweeted                                   bool
         retweeted_status                          object
         source                                    object
         truncated                                   bool
         user                                      object
         dtype: object
```

```
In [42]: tweet_json_clean = tweet_json_clean.rename(columns={'id': 'tweet_id'})
         tweet_json_clean.tweet_id =tweet_json_clean.tweet_id.astype(str)
```

Test

```
In [43]: tweet_json_clean.dtypes
```

```
Out[43]: contributors                             float64
         coordinates                              float64
         created_at                        datetime64[ns]
         display_text_range                        object
         entities                                  object
         extended_entities                         object
         favorite_count                             int64
         favorited                                   bool
         full_text                                 object
         geo                                      float64
         tweet_id                                  object
         id_str                                     int64
         in_reply_to_screen_name                   object
         in_reply_to_status_id                    float64
         in_reply_to_status_id_str                float64
         in_reply_to_user_id                      float64
         in_reply_to_user_id_str                  float64
         is_quote_status                             bool
         lang                                      object
         place                                     object
         possibly_sensitive                       float64
         possibly_sensitive_appealable            float64
         quoted_status                             object
         quoted_status_id                         float64
         quoted_status_id_str                     float64
         retweet_count                              int64
         retweeted                                   bool
         retweeted_status                          object
         source                                    object
         truncated                                   bool
         user                                      object
         dtype: object
```

1.8.6 Issue #12:

We only need favorite_count, retweet count,id "tweet_id" (after renamed), columns.

Define drop all unnecessary column to merge all dataset together later.

Code

```
In [44]: tweet_json_clean= tweet_json_clean.drop(['contributors', 'coordinates', 'created_at', '
                                                 'extended_entities', 'favorited', 'full_text',
                                                 'in_reply_to_screen_name', 'in_reply_to_status
                                                 'in_reply_to_user_id', 'in_reply_to_user_id_st
                                                 'possibly_sensitive', 'possibly_sensitive_appe
                                                 'quoted_status_id','quoted_status_id_str', 're
                                                 ,'truncated'], axis=1)
```

Test

```
In [45]: tweet_json_clean.dtypes
```

```
Out[45]: favorite_count     int64
         tweet_id          object
         retweet_count      int64
         user              object
         dtype: object
```

1.9 Storing Data

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

```
In [46]: #create a new dataframe by using method 'merge' to merge two dataset
         twitter_archive_master = pd.merge(twitter_archive_clean,
                                      image_predictions_clean,
                                      how = 'left', on = ['tweet_id'])
```

```
In [47]: twitter_archive_master = pd.merge(twitter_archive_master, tweet_json_clean,
                                      how = 'left', on = ['tweet_id'])
```

```
In [48]: #check the new dataframe
         twitter_archive_master.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2161 entries, 0 to 2160
Data columns (total 13 columns):
tweet_id              2161 non-null object
timestamp             2161 non-null datetime64[ns]
source                2161 non-null object
text                  2161 non-null object
rating_numerator      2161 non-null int64
rating_denominator    2161 non-null int64
name                  2161 non-null object
dog_stage             344 non-null object
jpg_url               1983 non-null object
breed_of_dog          1676 non-null object
favorite_count        2161 non-null int64
retweet_count         2161 non-null int64
user                  2161 non-null object
dtypes: datetime64[ns](1), int64(4), object(8)
memory usage: 236.4+ KB
```

1.10 Analyzing and Visualizing Data

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

1.10.1 Insight 1:

```
In [49]: #calculate the average retweet_count for each type of breeds of dogs.
         the_data = twitter_archive_master.groupby('breed_of_dog')['retweet_count'].mean().sort_
         the_data
```

```
Out[49]: breed_of_dog
         groenendael                      276.500000
```

```
Brabancon_griffon              287.666667
Japanese_spaniel               471.000000
Tibetan_terrier                557.750000
EntleBucher                    706.000000
Rhodesian_ridgeback            769.000000
Irish_wolfhound                819.000000
Lhasa                          826.000000
toy_terrier                    834.333333
Scottish_deerhound             872.750000
basenji                        890.111111
standard_schnauzer             928.000000
miniature_schnauzer            936.600000
miniature_poodle               975.250000
Dandie_Dinmont                1008.714286
soft-coated_wheaten_terrier   1011.142857
Welsh_springer_spaniel        1106.000000
redbone                       1113.000000
cairn                         1130.333333
curly-coated_retriever        1208.333333
Maltese_dog                   1238.263158
Sussex_spaniel                1239.000000
Scotch_terrier                1250.000000
papillon                      1298.142857
Irish_terrier                 1309.666667
West_Highland_white_terrier   1325.066667
beagle                        1352.500000
Yorkshire_terrier             1356.000000
Brittany_spaniel              1358.750000
German_short-haired_pointer   1369.875000
                                 ...
briard                        2966.666667
Pembroke                      3074.231579
Australian_terrier            3078.500000
malamute                      3106.515152
Norwich_terrier               3149.000000
Irish_setter                  3374.000000
Border_terrier                3377.142857
Norwegian_elkhound            3555.000000
golden_retriever              3759.358974
Cardigan                      3798.333333
Labrador_retriever            3828.780952
Leonberg                      3863.666667
Lakeland_terrier              4082.666667
black-and-tan_coonhound       4164.500000
Tibetan_mastiff               4193.000000
Eskimo_dog                    4235.590909
Mexican_hairless              4254.857143
Bouvier_des_Flandres          4479.000000
```

```
Irish_water_spaniel            4500.666667
flat-coated_retriever          4520.250000
Great_Pyrenees                 4784.285714
whippet                        4840.272727
Samoyed                        4843.952381
cocker_spaniel                 4858.100000
French_bulldog                 5039.677419
Saluki                         5133.750000
English_springer               5401.600000
standard_poodle                5508.909091
Afghan_hound                   5976.000000
Bedlington_terrier             7510.166667
Name: retweet_count, Length: 113, dtype: float64
```

- The breed of dogs that got the highest average in retweets is Bedlington_terrier with 7510 retweet in average.
- Where as the breed of dog that got the lowest retweet average is groenendael with 276 retweet in average.

1.10.2 Insight 2 & Visualization :

```
In [50]: #Calculate the average favorite_count for each type of breeds of dogs.
         the_data2 = twitter_archive_master.groupby('breed_of_dog')['favorite_count'].mean().sor
         the_data2
```

```
Out[50]: breed_of_dog
         Brabancon_griffon                885.000000
         groenendael                     1156.500000
         Japanese_spaniel                1362.000000
         Irish_wolfhound                 1534.000000
         Tibetan_terrier                 1841.000000
         standard_schnauzer              2046.000000
         Scottish_deerhound              2305.000000
         basenji                         2503.777778
         Lhasa                           2659.800000
         EntleBucher                     2678.000000
         Maltese_dog                     2959.684211
         toy_terrier                     3181.666667
         soft-coated_wheaten_terrier     3276.857143
         redbone                         3296.333333
         miniature_schnauzer             3409.000000
         miniature_poodle                3456.875000
         Dandie_Dinmont                  3464.571429
         Shih-Tzu                        3593.350000
         Scotch_terrier                  3624.000000
         Ibizan_hound                    3781.400000
         Rhodesian_ridgeback             4041.000000
         Sussex_spaniel                  4061.500000
```

```
papillon                      4402.571429
beagle                        4407.950000
Saint_Bernard                 4579.142857
Yorkshire_terrier             4595.000000
curly-coated_retriever        4612.000000
English_setter                4857.375000
keeshond                      4914.750000
bluetick                      5034.500000
                                 ...
Eskimo_dog                   10686.545455
Norwich_terrier              10806.000000
Pembroke                     10941.936842
bloodhound                   11080.285714
Australian_terrier           11127.500000
Norwegian_elkhound           11293.545455
basset                       11762.058824
Lakeland_terrier             11793.777778
Weimaraner                   11982.750000
Labrador_retriever           12024.142857
Tibetan_mastiff              12173.500000
golden_retriever             12451.942308
Cardigan                     12840.190476
whippet                      12911.454545
Great_Pyrenees               13117.571429
Border_terrier               13578.000000
cocker_spaniel               13580.400000
Mexican_hairless             13590.571429
Samoyed                      13902.523810
standard_poodle              13912.818182
English_springer             14138.300000
Leonberg                     14934.333333
Irish_water_spaniel          16400.000000
flat-coated_retriever        16791.625000
black-and-tan_coonhound      17012.000000
Afghan_hound                 17326.666667
Bouvier_des_Flandres         18032.000000
French_bulldog               18416.806452
Bedlington_terrier           21153.166667
Saluki                       24060.000000
Name: favorite_count, Length: 113, dtype: float64
```

- The breed of dogs that got the highest average in favorites is Saluki with 24060 favorites in average.
- Where as the breed of dog that got the lowest retweet average is brabancon_griffon with 885 retweet in average.
- it appears that there is a strong relationshipt between retweet_count and favorite_count for the tweet, since that the breed with least average in retweets came second to last for fa-

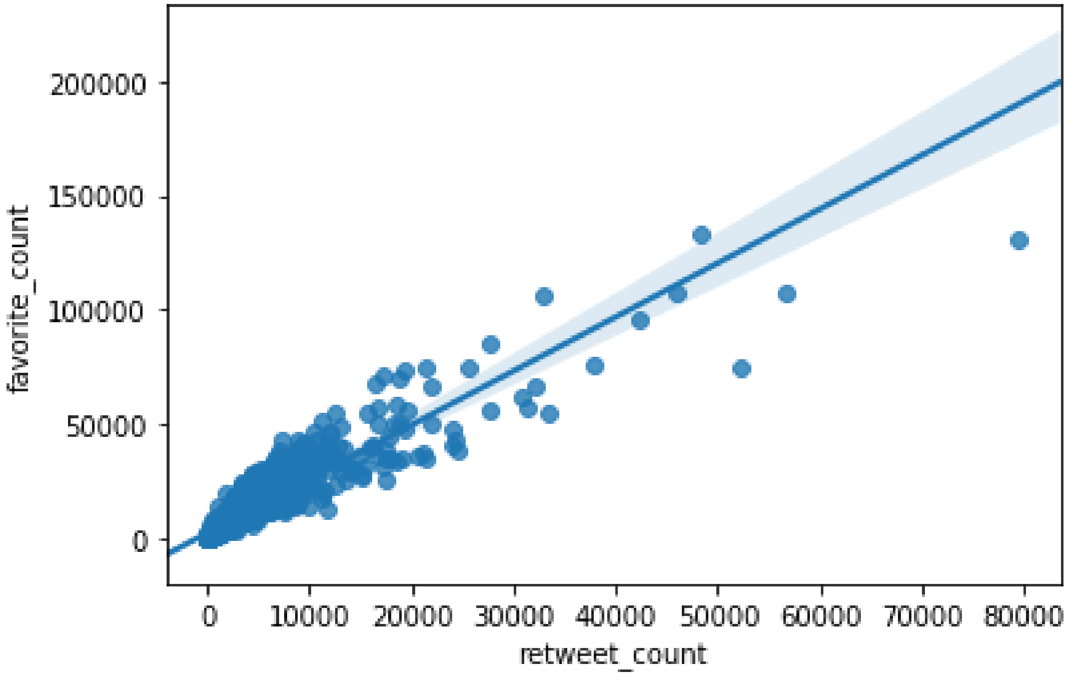
vorite_count and same for the highest average in tweets came the second highest average in favorite count. will try to confirm that in the next two cells.

```
In [51]: #calculate the correlation coeffecint between retweet_count and favorite_retweet.
         r = np.corrcoef(twitter_archive_master['retweet_count'], twitter_archive_master['favori
         #show it in the console
         r
```

```
Out[51]: array([[ 1.        ,  0.9116693],
                [ 0.9116693,  1.        ]])
```

```
In [52]: #regplot method is used to plot data and a linear regression model fit,,
         #There are a number of mutually exclusive options for estimating the regression model.
         sb.regplot(x="retweet_count", y="favorite_count", data=twitter_archive_master)
```

```
Out[52]: <matplotlib.axes._subplots.AxesSubplot at 0x7f4f9a90d240>
```

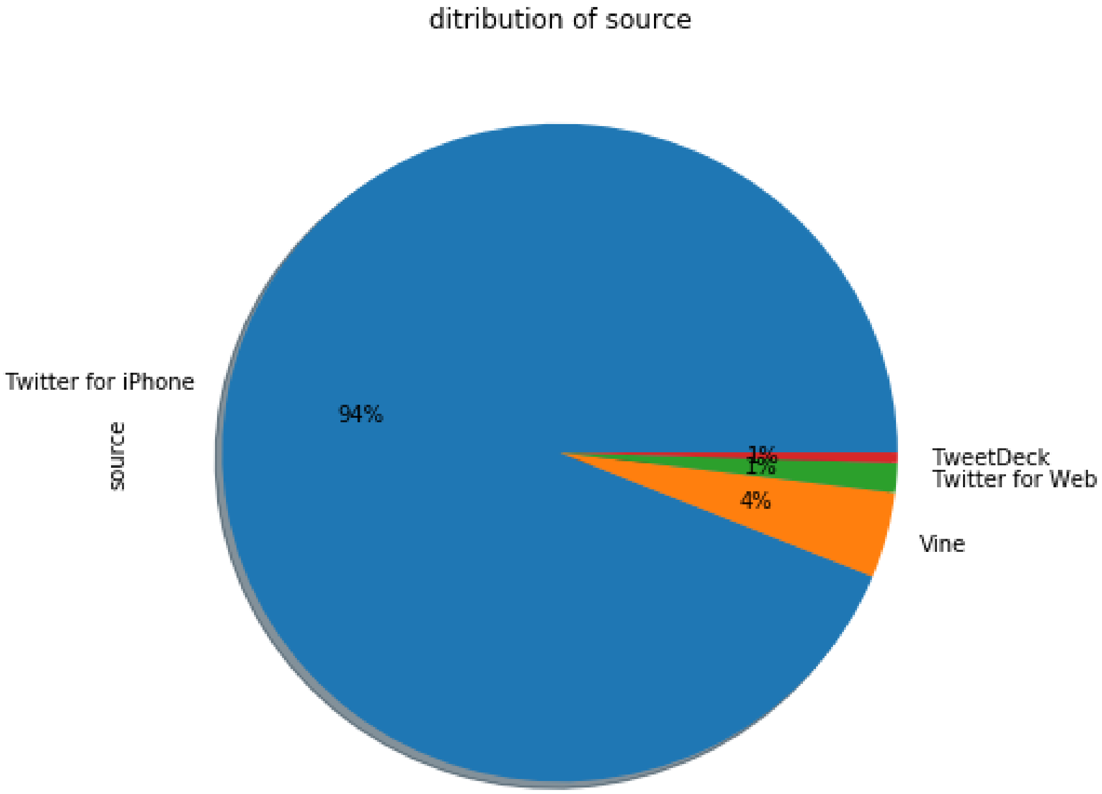


-

2 Insight 3 & Visualization

```
In [53]: #plotting the distribution of source of tweets.
         plot = twitter_archive_master.source.value_counts().plot.pie(figsize=(7, 7),autopct='%.
         plot.set_title('ditribution of source')
```

```
Out[53]: Text(0.5,1,'ditribution of source')
```

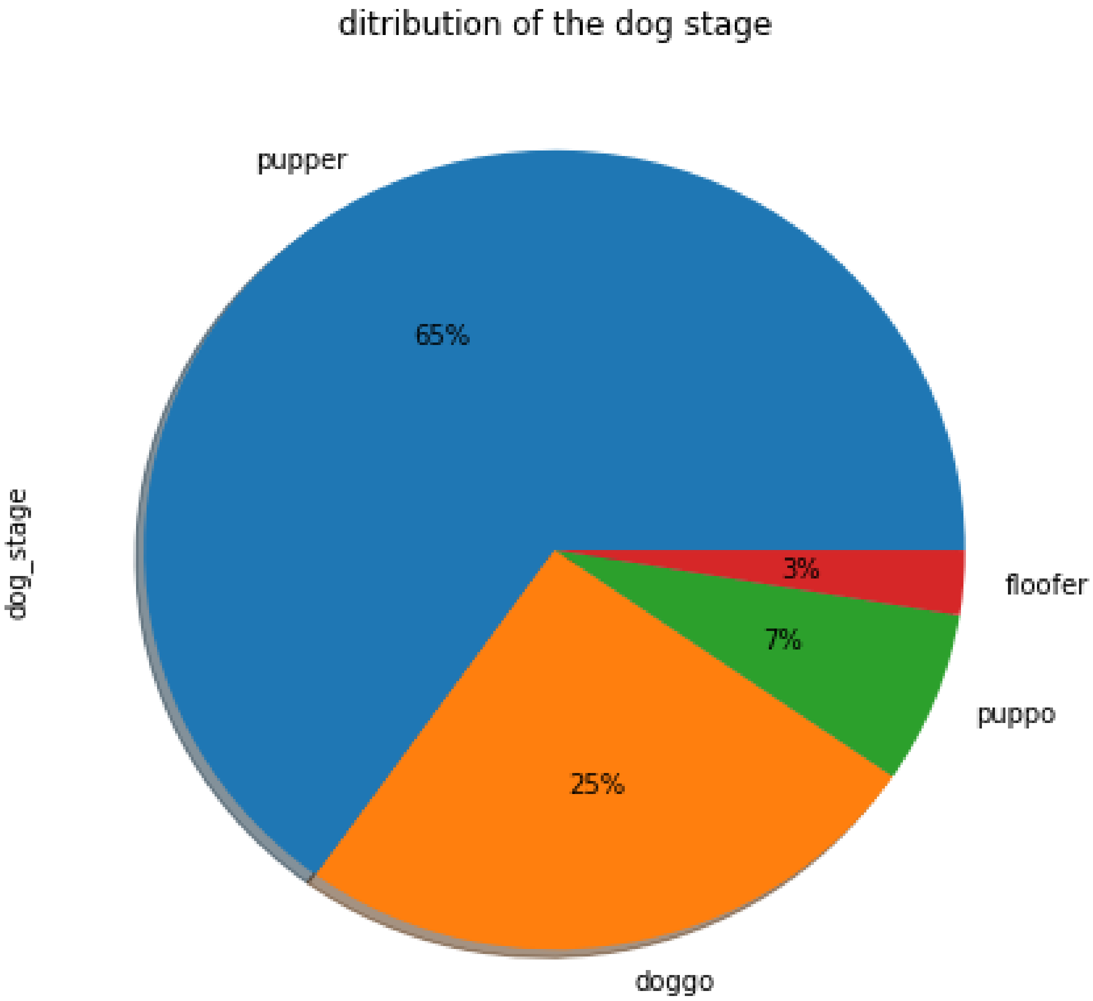


- As we can see in pie plot above, 94% of tweets in this dataset came from twitter for iPhone, which is an indicator that twitter in mobiles in general is where most of users uses the application.

2.1 Insight 4 & Visualization

```
In [54]: #plot the distribution of dog_stage in this dataset.
         plot2 = twitter_archive_master.dog_stage.value_counts().plot.pie(figsize=(7, 7),autopct
         plot2.set_title('ditribution of the dog stage')
```

```
Out[54]: Text(0.5,1,'ditribution of the dog stage')
```

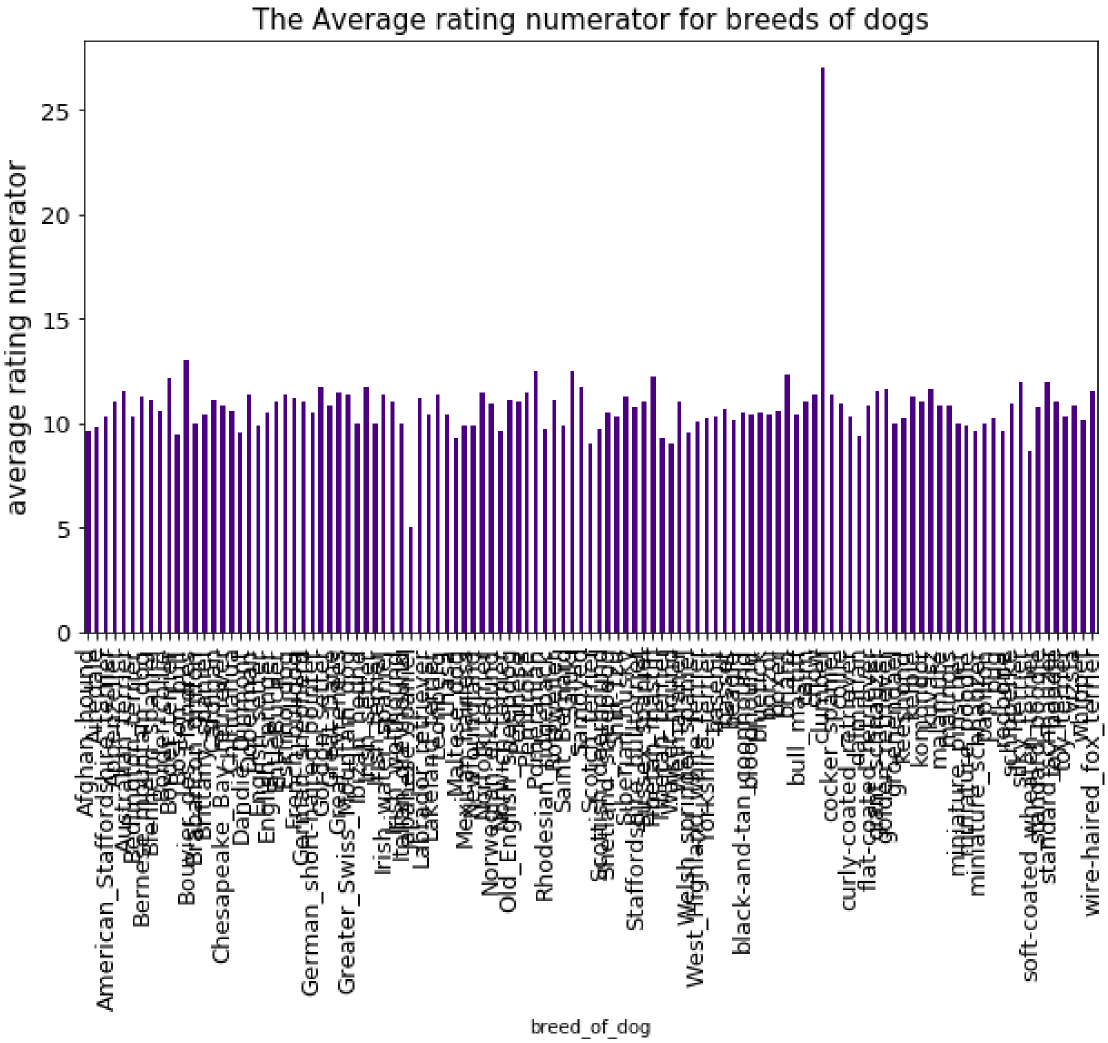


- 65% of dogs are pupper dogs, and 25% are doggo dogs.

2.1.1 Visualization

```
In [55]: #calculate the average of rating numerator for each type of breeds of dogs.
         breeds_of_dog = twitter_archive_master.groupby(['breed_of_dog'])['rating_numerator'].me


         #set the labels and title
         breeds_of_dog.set_title("The Average rating numerator for breeds of dogs", fontsize=15)
         breeds_of_dog.set_ylabel("average rating numerator", fontsize=15);
         plt.show()
```



Since we have so many breeds, we need to simplify the graph in order to see it probably.

```
In [61]: #calculate the average count for each breed of dogs
         twitter_archive_master['breed_of_dog'].value_counts().mean()
```

```
Out[61]: 14.831858407079647
```

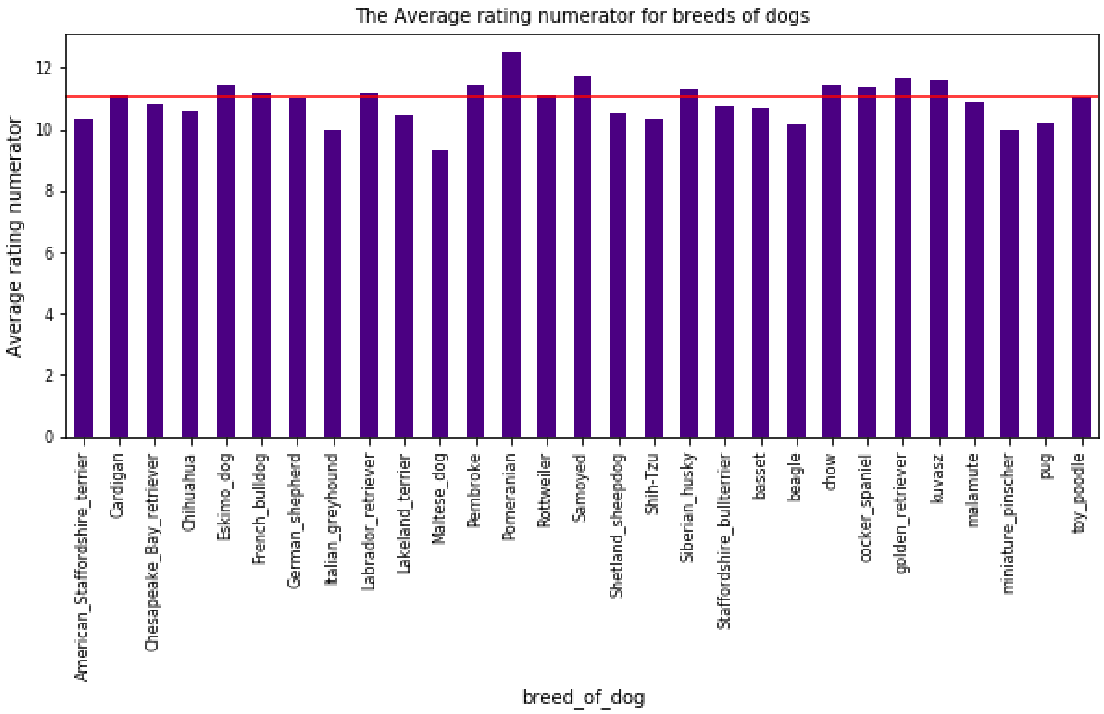
```
In [64]: #source of filter method 'https://stackoverflow.com/questions/13167391/filtering-groupe
         #take only breeds that have more than 15 records
         filtered_breed = twitter_archive_master.groupby(['breed_of_dog']).filter(lambda x: len(

         #plot the average rating for each breed.
         xx= filtered_breed.groupby(['breed_of_dog'])['rating_numerator'].mean().plot(kind='bar'

         #set the labels and title
         xx.set_title("The Average rating numerator for breeds of dogs", fontsize=10)
         xx.set_ylabel("Average rating numerator", fontsize=10);

         #calculate the average of rating for filtered_breed
         mean_rating= filtered_breed['rating_numerator'].mean()

         #plot a red line that represent the average rating for all breeds of dogs
         plt.axhline(mean_rating, color="r");
```



After filtering , we can see clearly the average rating for each type of breeds. Also, most of breeds are close the average with an exception for lakeland_terreir.

2.1.2 Insight 5

```
In [58]: twitter_archive_master.describe()
```

```
Out[58]:        rating_numerator  rating_denominator  favorite_count  retweet_count
         count       2161.000000         2161.000000     2161.000000    2161.000000
         mean          12.688107            9.990282     8779.167978    2768.202684
         std           47.228923            0.282839    12081.955511    4659.229659
         min            0.000000            0.000000       52.000000       0.000000
         25%           10.000000           10.000000     1909.000000     608.000000
         50%           11.000000           10.000000     4077.000000    1343.000000
         75%           12.000000           10.000000    11117.000000    3224.000000
         max         1776.000000           10.000000   132810.000000   79515.000000
```

- The average favorites is 8779 whereas the retweets is 2768 which shows that users tend to press the like button more often than the retweets, which is obvious since it an account for dog’s fans and like button is a gesture to indicate you like the dog.
- The average rating numerator is 12.68/10.

```
In [59]: #saving our cleaned and merged dataset: -

         #         * I comment this line to avoid saving the file multiple times.

         #twitter_archive_master.to_csv('twitter_archive_master.csv', index=True)
```

```
In [ ]:
```