Udacity Data Analyst Nanodegree

P3: Wrangle and Analyze Data

Author: Challa Sri Venkata Divya Madhuri

Date: Feb 4, 2018

Dataset

The dataset used for wrangling (and analyzing and visualizing) is the tweet archive of Twitter user @dog_rates, also known as WeRateDogs. WeRateDogs is a Twitter account that rates people's dogs with a humorous comment about the dog. These ratings almost always have a denominator of 10. The numerators, though? Almost always greater than 10. 11/10, 12/10, 13/10, etc. Why? Because "they're good dogs Brent." WeRateDogs has over 4 million followers and has received international media coverage.

WeRateDogs downloaded their Twitter archive and sent it to Udacity via email exclusively for you to use in this project. This archive contains basic tweet data (tweet ID, timestamp, text, etc.) for all 5000+ of their tweets as they stood on August 1, 2017.

WeRateDogs™ @dog_rates · Apr 16
Say hello to Alice. I'm told she enjoys car rides and smells good. 12/10 would give her everything she could ever want

WeRateDogs™ @dog_rates · Apr 16
A photographer took pictures before and after he told his bunny he's a good boy. Here are the results. 13/10

Data Wrangling

1. Gathering

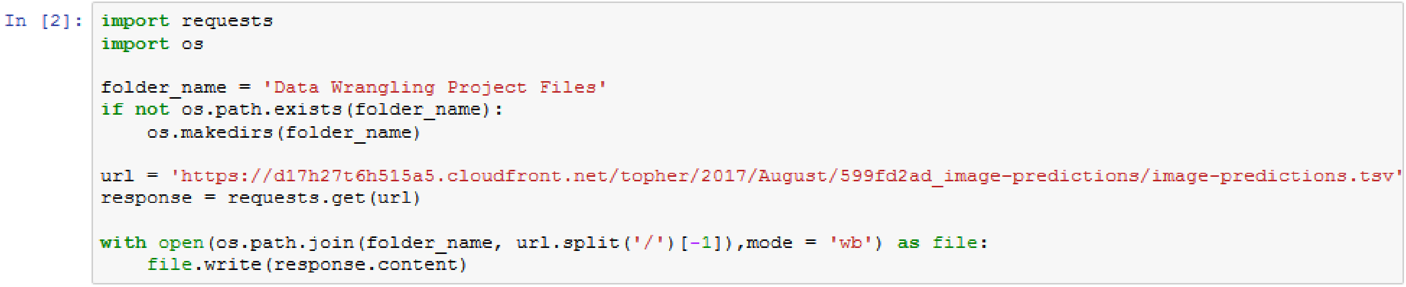
There are 3 data files (obtained from 3 different sources) used in this project :-

Enhanced Twitter Archive

The WeRateDogs Twitter archive contains basic tweet data for all 5000+ of their tweets, but not everything. One column of the archive does contain though: each tweet's text, which is used to extract rating, dog name, and dog "stage" (i.e. doggo, floofer, pupper, and puppo) to make this Twitter archive "enhanced." The 5000+ tweets have been filtered for tweets with ratings only (there are 2356). This is downloaded manually and then stored as ‘twitter_archive.csv’ file.

Image Predictions File

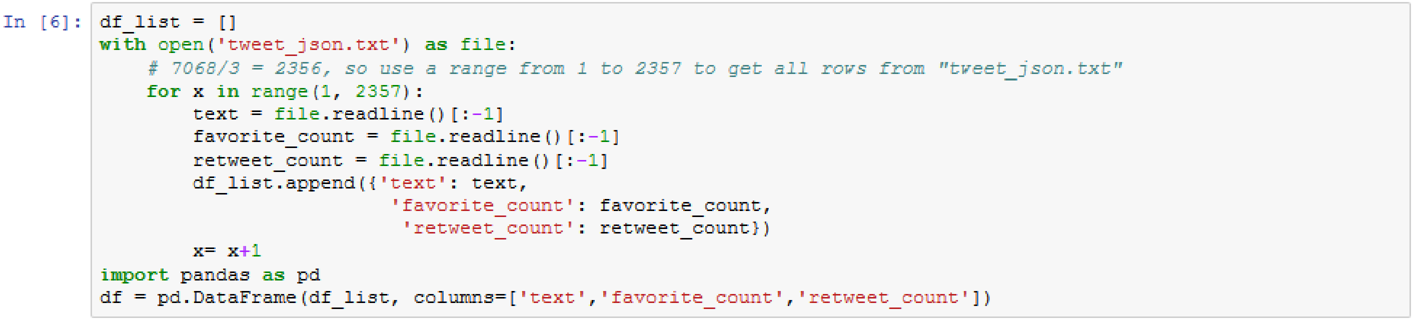
This file is full of image predictions (the top three only) alongside each tweet ID, image URL, and the image number that corresponded to the most confident prediction (numbered 1 to 4 since tweets can have up to four images). This file is downloaded programmatically using the Requests library and the following URL: https://d17h27t6h515a5.cloudfront.net/topher/2017/August/599fd2ad_image-predictions/image-predictions.tsv and then saved as ‘image_pred.tsv’ .



```
In [2]: import requests
        import os

        folder_name = 'Data Wrangling Project Files'
        if not os.path.exists(folder_name):
            os.makedirs(folder_name)

        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)
```

Additional Data via the Twitter API

This file contains 2 columns - retweet count and favorite count which are gathered by queryingTwitter's API using tweet_ids from twitter_archive file . Then its stored as a text file called ‘tweet_json.txt’ . Then this .txt file is read line by line into a pandas DataFrame with (at minimum) tweet ID, retweet count, and favorite count.



```
In [6]: df_list = []
        with open('tweet_json.txt') as file:
            # 7068/3 = 2356, so use a range from 1 to 2357 to get all rows from "tweet_json.txt"
            for x in range(1, 2357):
                text = file.readline()[:-1]
                favorite_count = file.readline()[:-1]
                retweet_count = file.readline()[:-1]
                df_list.append({'text': text,
                                 'favorite_count': favorite_count,
                                  'retweet_count': retweet_count})
                x= x+1
        import pandas as pd
        df = pd.DataFrame(df_list, columns=['text','favorite_count','retweet_count'])
```

2. Assessing

This step involves assessing the 3 datasets collected above (from 3 different sources) both visually and programmatically and find out quality and tidiness issues. Then these issues are documented as shown below:-

Quality issues

'twitter_archive' table

- tweet_id column should be of object type, not integer.
- Missing values in 'in_reply_to_status_id', 'in_reply_to_user_id', 'retweeted_status_id','retweeted_status_user_id', and 'retweeted_status_timestamp' columns can instead be left as null or filled with new values.
- 'timestamp' and 'retweeted_status_timestamp' are of object type instead of datetime.
- We only want to deal with non retweet tweets, hence the first rows where retweeted_status_id, retweeted_status_user_id, and retweeted_status_timestamp are not empty, must be removed.
- The columns doggo, floofer, pupper and puppo can be converted to have true/false values instead of None values.

'tweet_json' table

- Missing values in 'favorite_count' and 'retweet_count' columns. They can be left as null, or added with new values.
- 'favorite_count' and 'retweet_count' are object type and should be converted to numeric type.

'image_pred' table

- Nondescriptive column headers p1, p2, p3, p1_conf, p2_conf, p3_conf, p1_dog, p2_dog, p3_dog .
- The columns p1,p2 and p3 are various breeds of dogs and hence must be categorical instead of object type.
- The columns p1_dog, p2_dog and p3_dog must be of type bool.
- The columns 'p1', 'p2', 'p3' represent things which are not dogs. We can use False values in p1_dog, p2_dog and p3_dog to remove these non- dog entries.

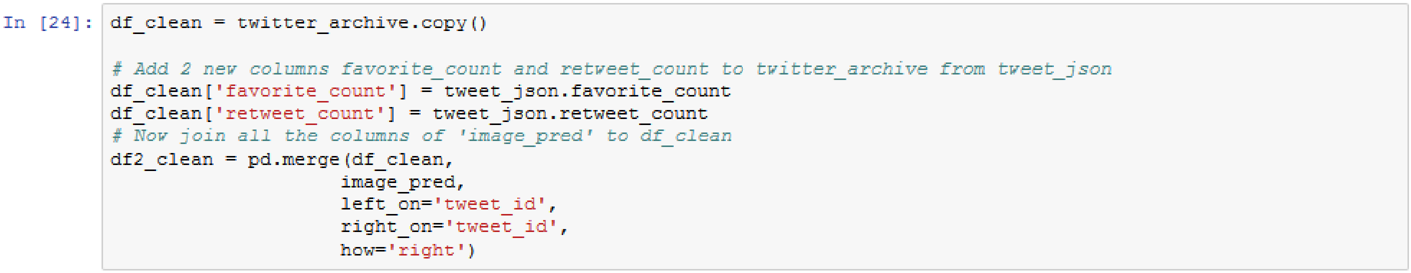
Tidiness issues

- All the 3 tables namely 'twitter_archive', 'image_pred' and 'tweet_json' have the same observational unit 'tweet_id'.
- The rating_numerator and rating_denominator must be combined into single column Overall_rating.

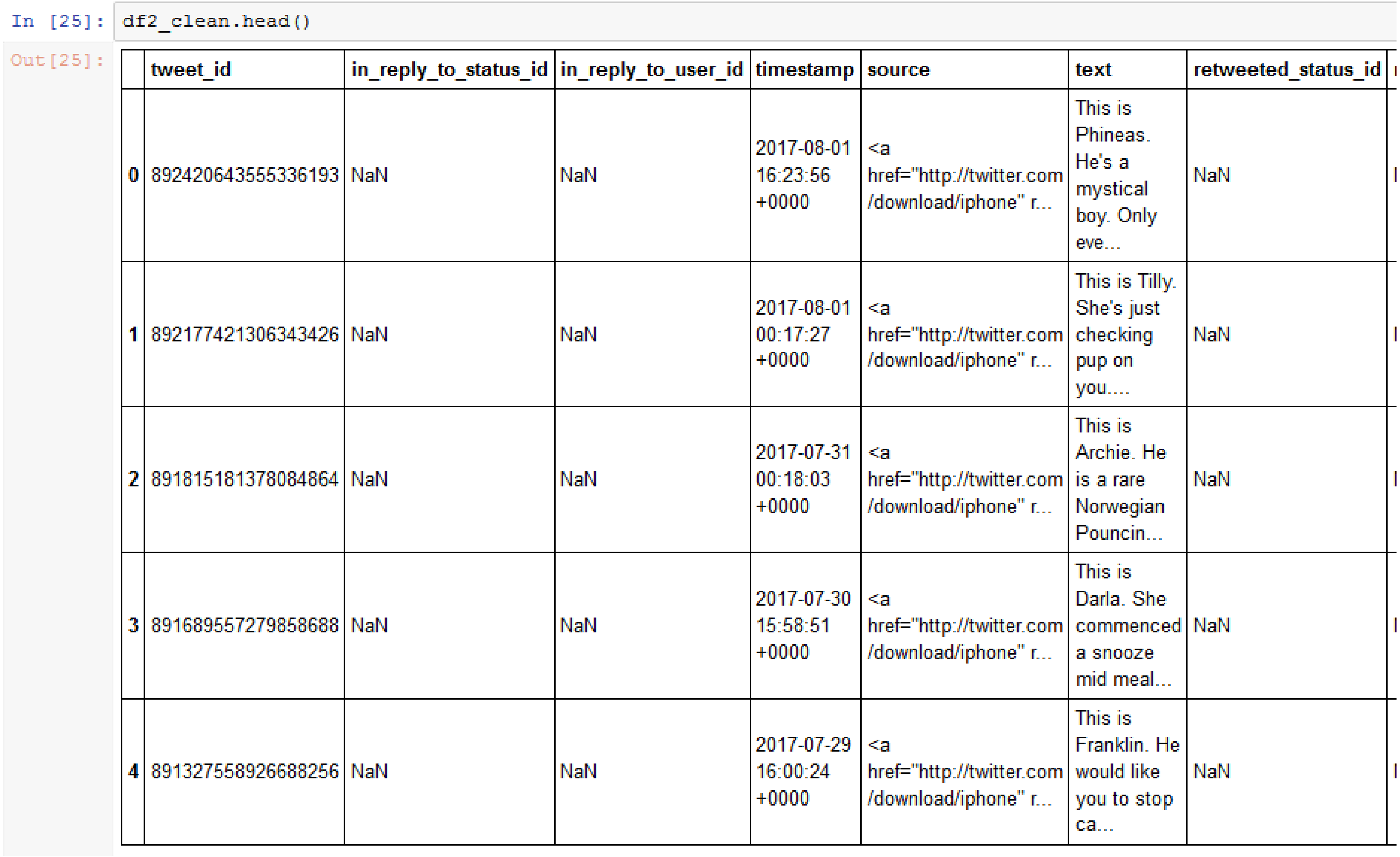
3. **Cleaning**

Define and Code

First, the tidiness issues are cleaned by combining all the 3 tables 'twitter_archive', 'image_pred' and 'tweet_json' into a single table as shown below :-



```
In [24]: df_clean = twitter_archive.copy()

         # Add 2 new columns favorite_count and retweet_count to twitter_archive from tweet_json
         df_clean['favorite_count'] = tweet_json.favorite_count
         df_clean['retweet_count'] = tweet_json.retweet_count
         # Now join all the columns of 'image_pred' to df_clean
         df2_clean = pd.merge(df_clean,
                              image_pred,
                              left_on='tweet_id',
                              right_on='tweet_id',
                              how='right')
```

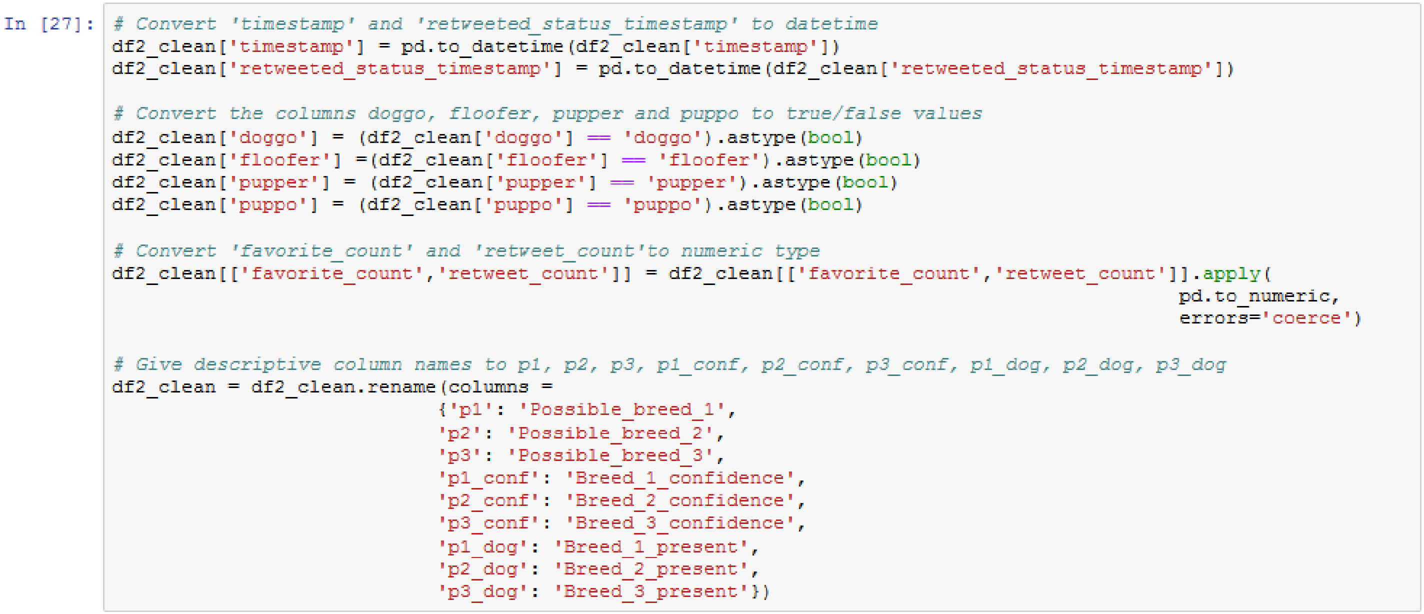


```
In [25]: df2_clean.head()
```

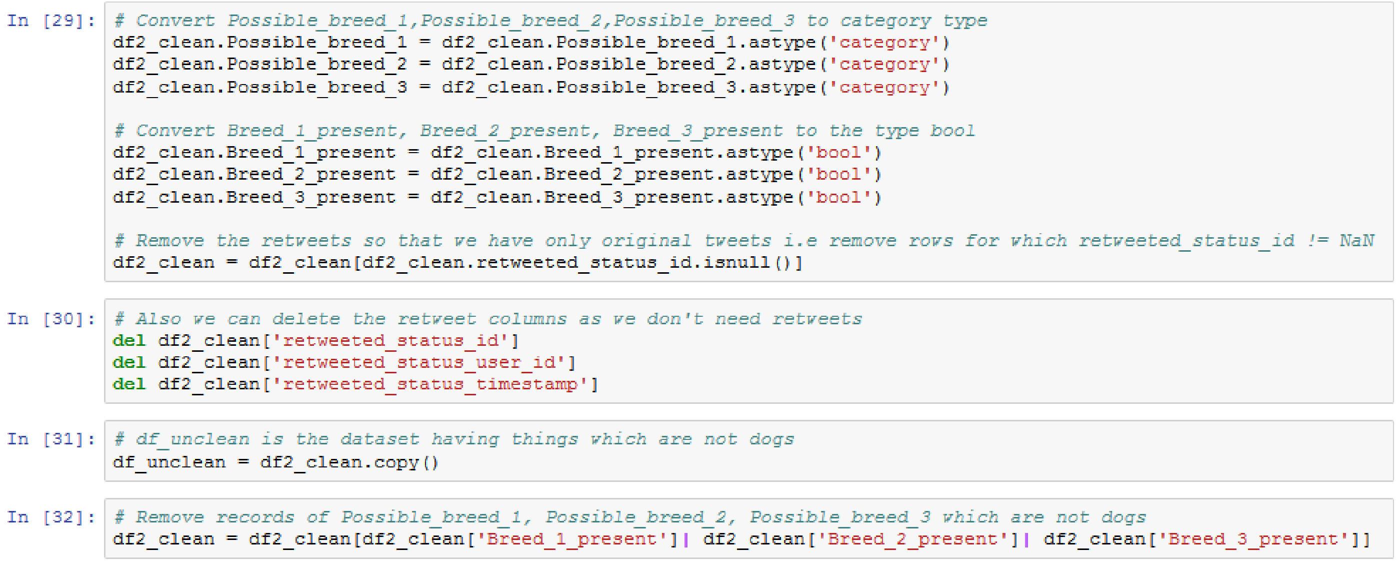
Out[25]:

	tweet_id	in_reply_to_status_id	in_reply_to_user_id	timestamp	source	text	retweeted_status_id
0	892420643555336193	NaN	NaN	2017-08-01 16:23:56 +0000	<a href="http://twitter.com/download/iphone" r...	This is Phineas. He's a mystical boy. Only eve...	NaN
1	892177421306343426	NaN	NaN	2017-08-01 00:17:27 +0000	<a href="http://twitter.com/download/iphone" r...	This is Tilly. She's just checking pup on you....	NaN
2	891815181378084864	NaN	NaN	2017-07-31 00:18:03 +0000	<a href="http://twitter.com/download/iphone" r...	This is Archie. He is a rare Norwegian Pouncin...	NaN
3	891689557279858688	NaN	NaN	2017-07-30 15:58:51 +0000	<a href="http://twitter.com/download/iphone" r...	This is Darla. She commenced a snooze mid meal...	NaN
4	891327558926688256	NaN	NaN	2017-07-29 16:00:24 +0000	<a href="http://twitter.com/download/iphone" r...	This is Franklin. He would like you to stop ca...	NaN

Then the quality issues are cleaned one by one .The columns with erroneous datatypes are converted to correct datatypes   and the records with retweets are removed as we need only original tweets. Then the entries which don’t represent dogs but things like paper-towels etc are removed. The code used for these steps is shown below :-

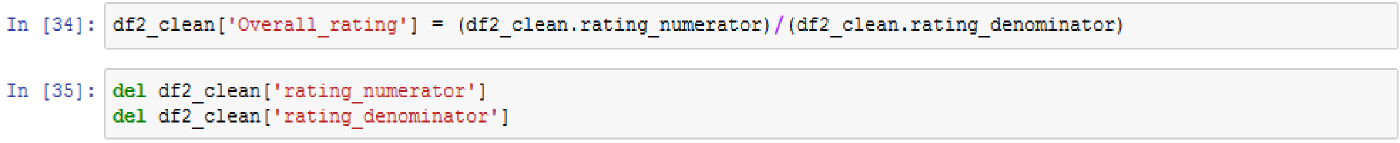


```
In [27]: # Convert 'timestamp' and 'retweeted_status_timestamp' to datetime
         df2_clean['timestamp'] = pd.to_datetime(df2_clean['timestamp'])
         df2_clean['retweeted_status_timestamp'] = pd.to_datetime(df2_clean['retweeted_status_timestamp'])

         # Convert the columns doggo, floofer, pupper and puppo to true/false values
         df2_clean['doggo'] = (df2_clean['doggo'] == 'doggo').astype(bool)
         df2_clean['floofer'] =(df2_clean['floofer'] == 'floofer').astype(bool)
         df2_clean['pupper'] = (df2_clean['pupper'] == 'pupper').astype(bool)
         df2_clean['puppo'] = (df2_clean['puppo'] == 'puppo').astype(bool)

         # Convert 'favorite_count' and 'retweet_count'to numeric type
         df2_clean[['favorite_count','retweet_count']] = df2_clean[['favorite_count','retweet_count']].apply(
                                                                                          pd.to_numeric,
                                                                                          errors='coerce')

         # Give descriptive column names to p1, p2, p3, p1_conf, p2_conf, p3_conf, p1_dog, p2_dog, p3_dog
         df2_clean = df2_clean.rename(columns =
                                      {'p1': 'Possible_breed_1',
                                      'p2': 'Possible_breed_2',
                                      'p3': 'Possible_breed_3',
                                      'p1_conf': 'Breed_1_confidence',
                                      'p2_conf': 'Breed_2_confidence',
                                      'p3_conf': 'Breed_3_confidence',
                                      'p1_dog': 'Breed_1_present',
                                      'p2_dog': 'Breed_2_present',
                                      'p3_dog': 'Breed_3_present'})
```



```
In [29]: # Convert Possible_breed_1,Possible_breed_2,Possible_breed_3 to category type
         df2_clean.Possible_breed_1 = df2_clean.Possible_breed_1.astype('category')
         df2_clean.Possible_breed_2 = df2_clean.Possible_breed_2.astype('category')
         df2_clean.Possible_breed_3 = df2_clean.Possible_breed_3.astype('category')

         # Convert Breed_1_present, Breed_2_present, Breed_3_present to the type bool
         df2_clean.Breed_1_present = df2_clean.Breed_1_present.astype('bool')
         df2_clean.Breed_2_present = df2_clean.Breed_2_present.astype('bool')
         df2_clean.Breed_3_present = df2_clean.Breed_3_present.astype('bool')

         # Remove the retweets so that we have only original tweets i.e remove rows for which retweeted_status_id != NaN
         df2_clean = df2_clean[df2_clean.retweeted_status_id.isnull()]

In [30]: # Also we can delete the retweet columns as we don't need retweets
         del df2_clean['retweeted_status_id']
         del df2_clean['retweeted_status_user_id']
         del df2_clean['retweeted_status_timestamp']

In [31]: # df_unclean is the dataset having things which are not dogs
         df_unclean = df2_clean.copy()

In [32]: # Remove records of Possible_breed_1, Possible_breed_2, Possible_breed_3 which are not dogs
         df2_clean = df2_clean[df2_clean['Breed_1_present']| df2_clean['Breed_2_present']| df2_clean['Breed_3_present']]
```

Finally the columns rating_numerator and rating_denominator are converted to a single column Overall_rating as shown :-

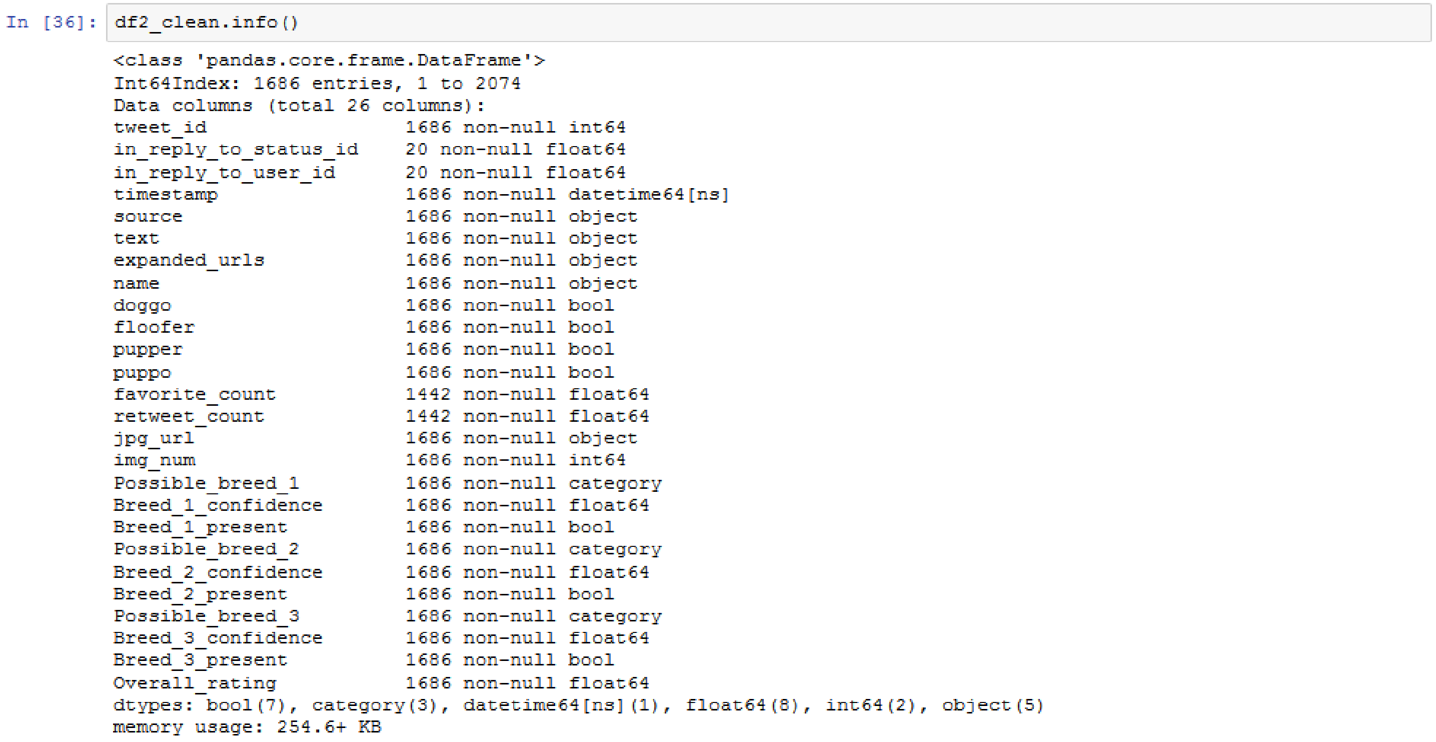


```
In [34]: df2_clean['Overall_rating'] = (df2_clean.rating_numerator)/(df2_clean.rating_denominator)

In [35]: del df2_clean['rating_numerator']
         del df2_clean['rating_denominator']
```

The missing values in 'in_reply_to_status_id', 'in_reply_to_user_id', 'retweeted_status_id', 'retweeted_status_user_id', 'retweeted_status_timestamp', 'favorite_count' and 'retweet_count' columns cannot be treated as we cannot collect and add those missing values.

Test

The final cleaned dataset is tested as shown :-



```
In [36]: df2_clean.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1686 entries, 1 to 2074
Data columns (total 26 columns):
tweet_id                 1686 non-null int64
in_reply_to_status_id    20 non-null float64
in_reply_to_user_id      20 non-null float64
timestamp                1686 non-null datetime64[ns]
source                   1686 non-null object
text                     1686 non-null object
expanded_urls            1686 non-null object
name                     1686 non-null object
doggo                    1686 non-null bool
floofer                  1686 non-null bool
pupper                   1686 non-null bool
puppo                    1686 non-null bool
favorite_count           1442 non-null float64
retweet_count            1442 non-null float64
jpg_url                  1686 non-null object
img_num                  1686 non-null int64
Possible_breed_1         1686 non-null category
Breed_1_confidence       1686 non-null float64
Breed_1_present          1686 non-null bool
Possible_breed_2         1686 non-null category
Breed_2_confidence       1686 non-null float64
Breed_2_present          1686 non-null bool
Possible_breed_3         1686 non-null category
Breed_3_confidence       1686 non-null float64
Breed_3_present          1686 non-null bool
Overall_rating           1686 non-null float64
dtypes: bool(7), category(3), datetime64[ns](1), float64(8), int64(2), object(5)
memory usage: 254.6+ KB
```