1- Importing the required libraries.

2- Gathering Data

2-1 Gathering data from twitter-archive csv format

2-2 Gathering the tweet image predictions programmatically hosted on udacity.

2-3 Gathering for each tweet's

How many favorites the tweet had?

How many Count of the retweet?

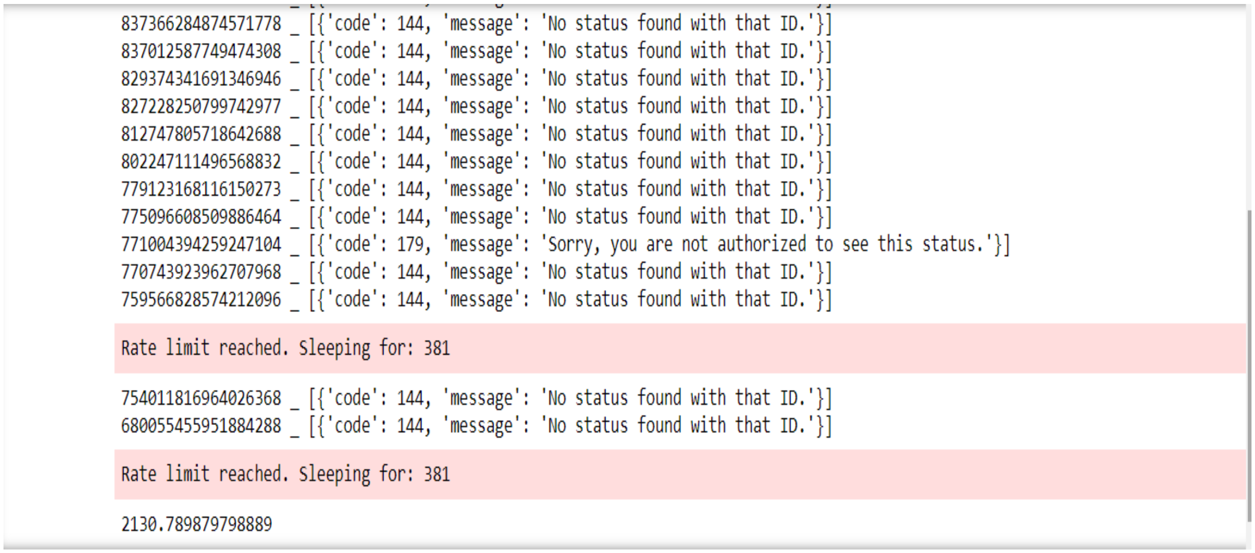
How many followers the user had?

How many favorites the user had?

The date and time of the creation?

Using the tweet IDs in the WeRateDogs Twitter archive

Using Twitter API



```
837366284874571778 _ [{'code': 144, 'message': 'No status found with that ID.'}]
837012587749474308 _ [{'code': 144, 'message': 'No status found with that ID.'}]
829374341691346946 _ [{'code': 144, 'message': 'No status found with that ID.'}]
827228250799742977 _ [{'code': 144, 'message': 'No status found with that ID.'}]
812747805718642688 _ [{'code': 144, 'message': 'No status found with that ID.'}]
802247111496568832 _ [{'code': 144, 'message': 'No status found with that ID.'}]
779123168116150273 _ [{'code': 144, 'message': 'No status found with that ID.'}]
775096608509886464 _ [{'code': 144, 'message': 'No status found with that ID.'}]
771004394259247104 _ [{'code': 179, 'message': 'Sorry, you are not authorized to see this status.'}]
770743923962707968 _ [{'code': 144, 'message': 'No status found with that ID.'}]
759566828574212096 _ [{'code': 144, 'message': 'No status found with that ID.'}]

Rate limit reached. Sleeping for: 381

754011816964026368 _ [{'code': 144, 'message': 'No status found with that ID.'}]
680055455951884288 _ [{'code': 144, 'message': 'No status found with that ID.'}]

Rate limit reached. Sleeping for: 381

2130.789879798889
```

3- Assess

3-1 Quality Issue

Dataset	Quality Issue	Quality dimensions
“twitter_archive”	1. All dog stages using 'None' instead of 'NaN'.	Consistency
	2. In name change 'None' to NaN.	Consistency
	3. There are many invalid names present {they all start with lower case letters}.	Validity
	4. Columns pertaining to retweets and expanded URLs are unnecessary.	Consistency
	5. Timestamp is string not date time	Consistency
	6. Rating_numerator there are 39 different numerators [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 17, 20, 24, 26, 27, 44, 45, 50, 60, 75, 80, 84, 88, 99, 121, 143, 144, 165, 182, 204, 420, 666, 960, 1776] two numerators are equal to 0 tweet_id: 835152434251116546 (rating in the text column is 0/10) tweei_id: 746906459439529985 (rating in the text column is 0/10)	Precision
	7. Rating_denominator there are 18 different denumerators , [0, 2, 7, 10, 11, 15, 16, 20, 40, 50, 70, 80, 90,110, 120, 130, 150,170] Valid rates are [2,7,10] one of them is equal to 0 tweet_id: 835246439529840640 (rating in the text column is 13/10)	Precision
	8. There are a few cases, where a dog has more than one style: tweet_id: 854010172552949000 (doggo, floofer) tweet_id: 808106460588765000 (doggo, pupper) tweet_id: 801115127852503000 (doggo, pupper) tweet_id: 781308096455073000 (doggo, pupper)	Validity
	9. There could be encoding problem for 757354760399941633 Devón 720389942216527872 Ralphé 717047459982213120 Flávio 694352839993344000 Oliviér	Validity

	688547210804498433 Frönq 686050296934563840 Flávio 669371483794317312 Oliviér 668872652652679168 Amélie 668528771708952576 Gòrdón (the name value uses non-English characters)	
“image_predictions”	1. There are a few cases, where a p1_dog, p2_dog, and p3_dog are all "False" in df_image it means there is no correct prediction.	Validity
	2. Values for p1, 2, and p3 not always capitalized	Precision

3-2 Tidiness Issues

Dataset	Tidiness Issue	Tidiness dimensions
“twitter_archive”	1. Doggo, floofer, pupper, puppo are one variable spread across different columns	Each variable forms a column.
	2. Rating_numerator and rating_denominator can be combined into one column	Each variable forms a column.
image_predictions	3. “img_num" contains integer values ranging from 1 to 4 but only 1 img_url is present (this column semantics is not clear). The column may not have any use here.	
General	4. There are too many datasets and their overall structure is untidy	Each type of observational unit forms a table.
	5. After merging data frames date time in “twitter_archive” has the same values as timestamp column in “tweet_json”	Duplicate column

4- Cleaning

Starting with The last tidiness issue

“There are too many datasets and their overall structure is untidy”

Solution:

Merge all data frames into one based on “tweet_id”

Quality issues:

1. All dog stages using 'None' instead of 'NaN'.

   Solution:

   # Change ‘None’ to’ NaN’ for ['doggo', 'floofer', 'pupper', 'puppo'] columns
2. In name change 'None' to NaN.

   Solution:

   # Change ‘None’ to’ NaN’ for [‘name’] columns.
3. There are many invalid names present {they all start with lower case letters}.

   Solution:

   # assign NaN to all invalid
4. Columns pertaining to retweets and expanded URLs are unnecessary.

   Solution:

   # Delete the retweets using “retweeted_status_id”.

   # Delete duplicated tweet_id.

   # Delete tweets with no pictures.

   # Delete columns related to retweet [retweeted_status_id, retweeted_status_user_id , retweeted_status_timestamp]

5. Timestamp is string not date time.

   Solution:

   # Convert into datetime.

6. Rating_numerator there are 39 different numerators

   Solution:

   # Standardized rating values.

7. Rating_denominator there are 18 different numerators

   Solution:

   # Standardized rating values.

8. There are a few cases, where a dog has more than one style: tweet_id: 854010172552949000 (doggo, floofer) tweet_id: 808106460588765000 (doggo, pupper) tweet_id: 801115127852503000 (doggo, pupper) tweet_id: 781308096455073000 (doggo, pupper)

   Solution:

   # Concatenate all 4 columns to 1 column dog_stage.

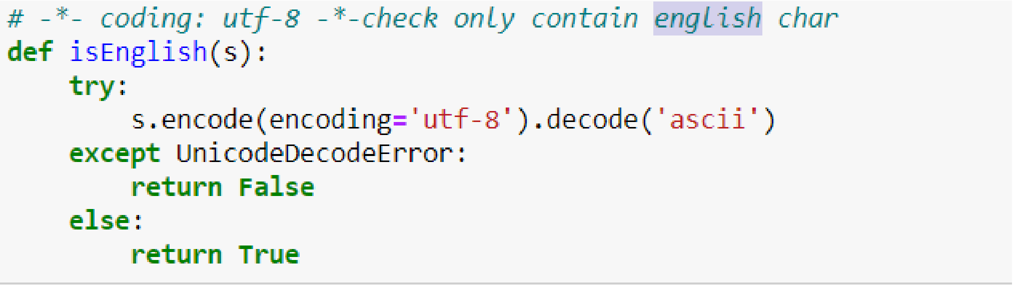
#Remove the original 4 columns of dog stages.

9. There could be encoding problem for tweet_id = 668528771708952576 (the name value uses non-English characters)

   Solution:

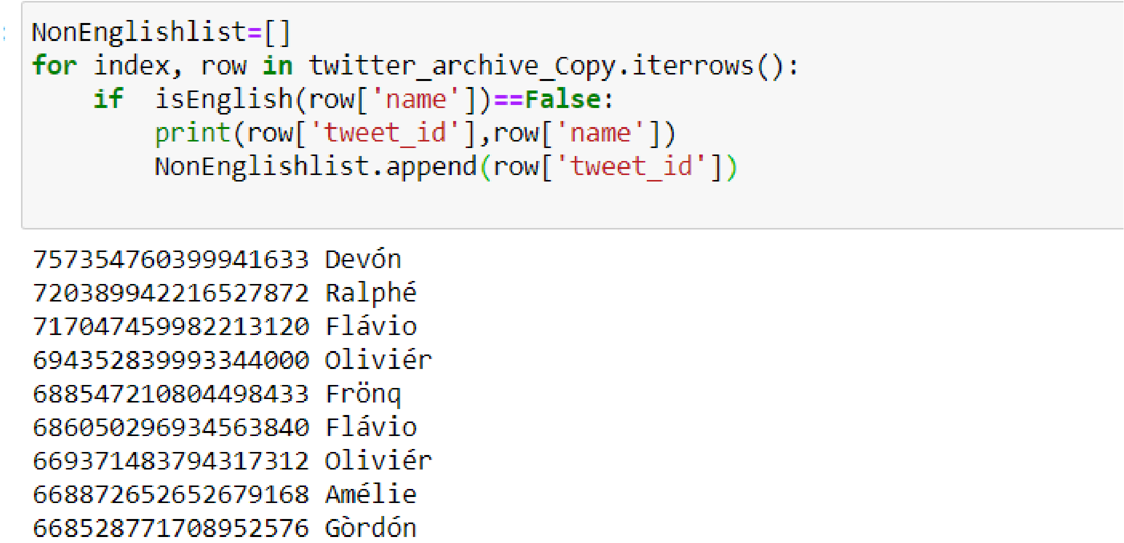
   # Delete observations which contain non-English character.

   9.1 Function to check if text contain any non-English character



```
# -*- coding: utf-8 -*-check only contain english char
def isEnglish(s):
    try:
        s.encode(encoding='utf-8').decode('ascii')
    except UnicodeDecodeError:
        return False
    else:
        return True
```

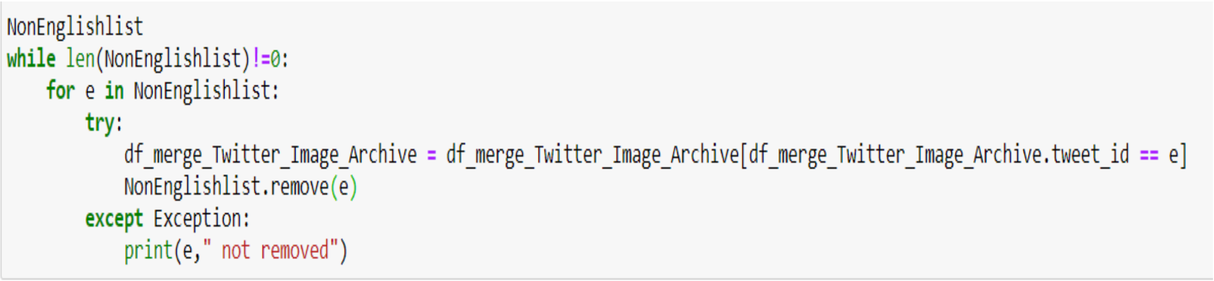
   Show all observation’s tweet id have non-English name



```
NonEnglishlist=[]
for index, row in twitter_archive_Copy.iterrows():
    if  isEnglish(row['name'])==False:
        print(row['tweet_id'],row['name'])
        NonEnglishlist.append(row['tweet_id'])
```

```
757354760399941633 Devón
720389942216527872 Ralphé
717047459982213120 Flávio
694352839993344000 Oliviér
688547210804498433 Frönq
686050296934563840 Flávio
669371483794317312 Oliviér
668872652652679168 Amélie
668528771708952576 Gòrdón
```

Remove all observation which contain non-English character



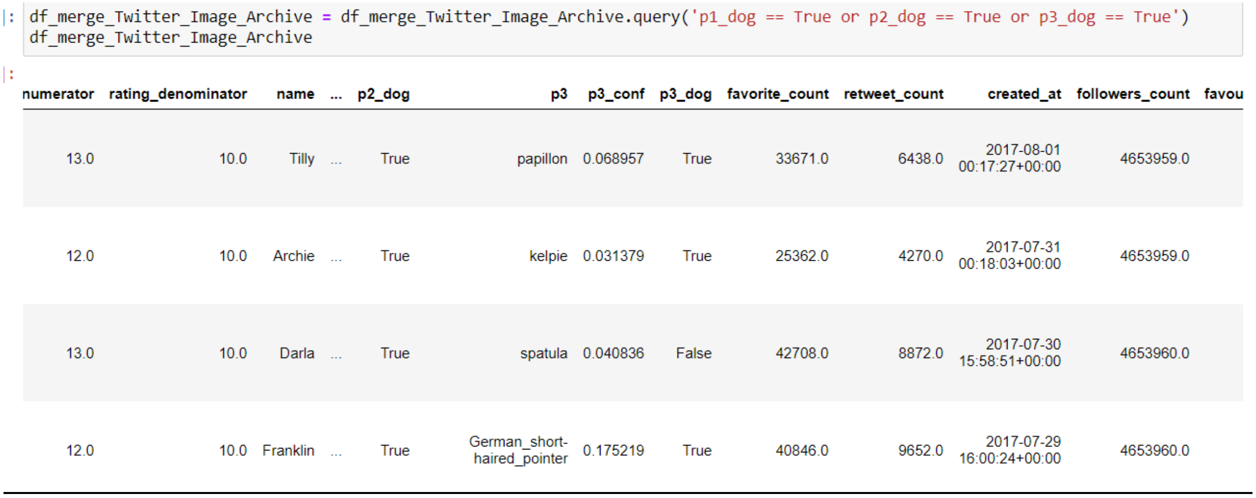
```
NonEnglishlist
while len(NonEnglishlist)!=0:
    for e in NonEnglishlist:
        try:
            df_merge_Twitter_Image_Archive = df_merge_Twitter_Image_Archive[df_merge_Twitter_Image_Archive.tweet_id == e]
            NonEnglishlist.remove(e)
        except Exception:
            print(e," not removed")
```

10. There are a few cases, where a p1_dog, p2_dog, and p3_dog are all "False" in image_predictions it means there is no correct prediction

Solution:

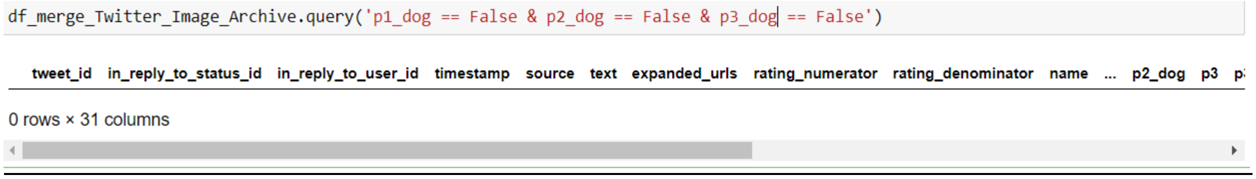
# Remove entries where p1_dog, p2_dog, and p3_dog are all "False" in image_predictions

.



```
df_merge_Twitter_Image_Archive = df_merge_Twitter_Image_Archive.query('p1_dog == True or p2_dog == True or p3_dog == True')
df_merge_Twitter_Image_Archive
```

numerator	rating_denominator	name	...	p2_dog	p3	p3_conf	p3_dog	favorite_count	retweet_count	created_at	followers_count	favou
13.0	10.0	Tilly	...	True	papillon	0.068957	True	33671.0	6438.0	2017-08-01 00:17:27+00:00	4653959.0	
12.0	10.0	Archie	...	True	kelpie	0.031379	True	25362.0	4270.0	2017-07-31 00:18:03+00:00	4653959.0	
13.0	10.0	Darla	...	True	spatula	0.040836	False	42708.0	8872.0	2017-07-30 15:58:51+00:00	4653960.0	
12.0	10.0	Franklin	...	True	German_short-haired_pointer	0.175219	True	40846.0	9652.0	2017-07-29 16:00:24+00:00	4653960.0	



```
df_merge_Twitter_Image_Archive.query('p1_dog == False & p2_dog == False & p3_dog == False')
```

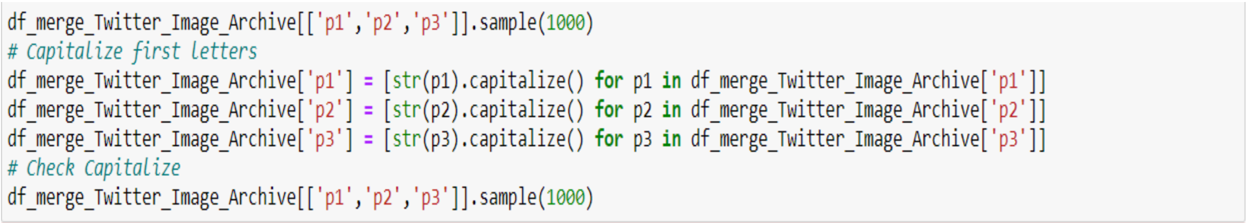
tweet_id	in_reply_to_status_id	in_reply_to_user_id	timestamp	source	text	expanded_urls	rating_numerator	rating_denominator	name	...	p2_dog	p3	p:

0 rows × 31 columns

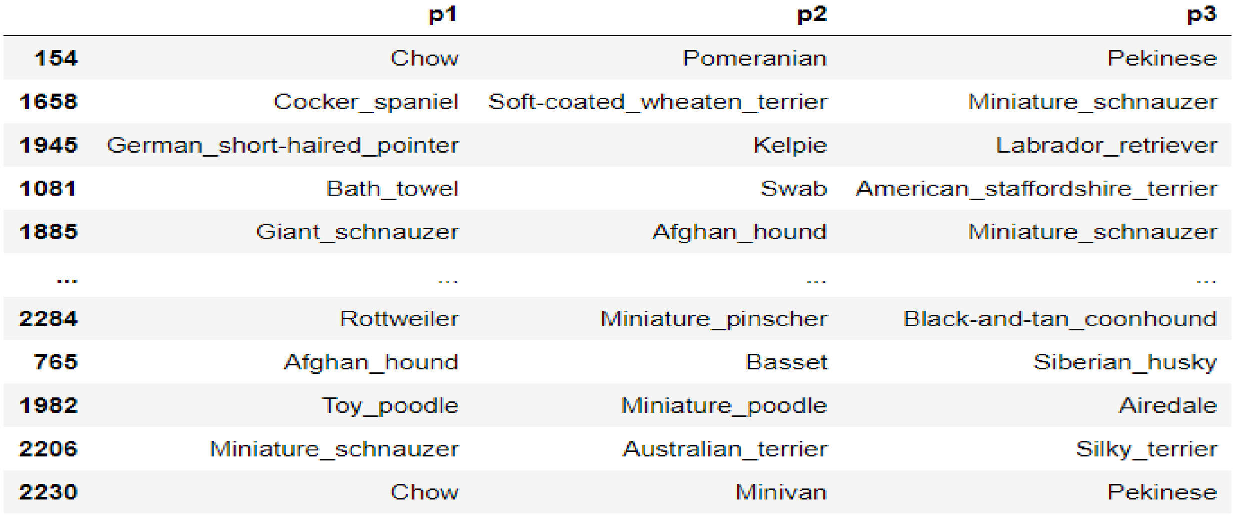
11. Values for p1, 2, and p3 not always capitalized

Solution:

Capitalize first letters for ‘p1,’p2’ and ‘p3’.



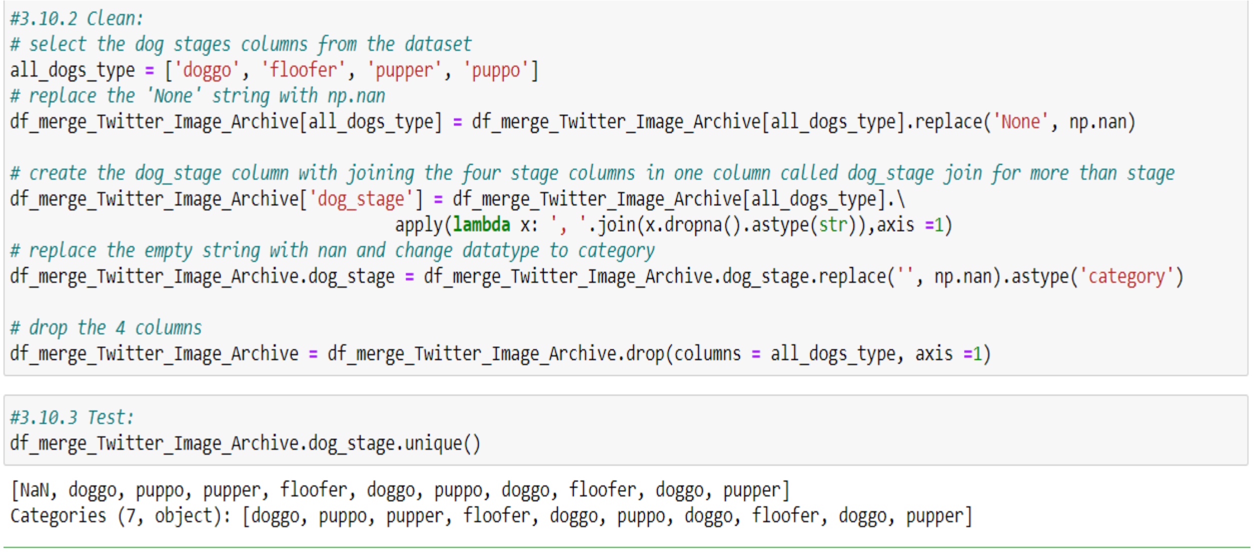
```
df_merge_Twitter_Image_Archive[['p1','p2','p3']].sample(1000)
# Capitalize first letters
df_merge_Twitter_Image_Archive['p1'] = [str(p1).capitalize() for p1 in df_merge_Twitter_Image_Archive['p1']]
df_merge_Twitter_Image_Archive['p2'] = [str(p2).capitalize() for p2 in df_merge_Twitter_Image_Archive['p2']]
df_merge_Twitter_Image_Archive['p3'] = [str(p3).capitalize() for p3 in df_merge_Twitter_Image_Archive['p3']]
# Check Capitalize
df_merge_Twitter_Image_Archive[['p1','p2','p3']].sample(1000)
```



	p1	p2	p3
154	Chow	Pomeranian	Pekinese
1658	Cocker_spaniel	Soft-coated_wheaten_terrier	Miniature_schnauzer
1945	German_short-haired_pointer	Kelpie	Labrador_retriever
1081	Bath_towel	Swab	American_staffordshire_terrier
1885	Giant_schnauzer	Afghan_hound	Miniature_schnauzer
...	...	...	...
2284	Rottweiler	Miniature_pinscher	Black-and-tan_coonhound
765	Afghan_hound	Basset	Siberian_husky
1982	Toy_poodle	Miniature_poodle	Airedale
2206	Miniature_schnauzer	Australian_terrier	Silky_terrier
2230	Chow	Minivan	Pekinese

Tidiness Issues

1. Doggo, floofer, pupper, puppo are one variable spread across different columns
   Solution:
   # Create the dog_stage column with joining the four stage columns in one column called dog_stage join for more than stage.
   # Replace the empty string with NaN and change datatype to category.
   # drop the 4 columns ['doggo', 'floofer', 'pupper', 'puppo']

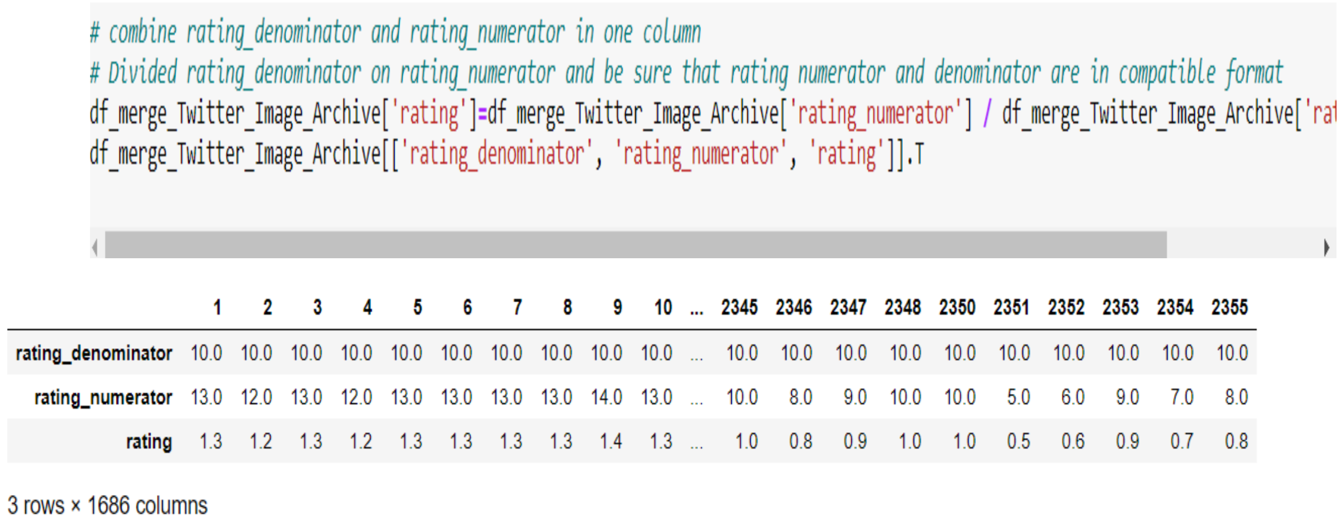


```
#3.10.2 Clean:
# select the dog stages columns from the dataset
all_dogs_type = ['doggo', 'floofer', 'pupper', 'puppo']
# replace the 'None' string with np.nan
df_merge_Twitter_Image_Archive[all_dogs_type] = df_merge_Twitter_Image_Archive[all_dogs_type].replace('None', np.nan)

# create the dog_stage column with joining the four stage columns in one column called dog_stage join for more than stage
df_merge_Twitter_Image_Archive['dog_stage'] = df_merge_Twitter_Image_Archive[all_dogs_type].\
                                        apply(lambda x: ', '.join(x.dropna().astype(str)),axis =1)
# replace the empty string with nan and change datatype to category
df_merge_Twitter_Image_Archive.dog_stage = df_merge_Twitter_Image_Archive.dog_stage.replace('', np.nan).astype('category')

# drop the 4 columns
df_merge_Twitter_Image_Archive = df_merge_Twitter_Image_Archive.drop(columns = all_dogs_type, axis =1)
```

```
#3.10.3 Test:
df_merge_Twitter_Image_Archive.dog_stage.unique()
```

```
[NaN, doggo, puppo, pupper, floofer, doggo, puppo, doggo, floofer, doggo, pupper]
Categories (7, object): [doggo, puppo, pupper, floofer, doggo, puppo, doggo, floofer, doggo, pupper]
```

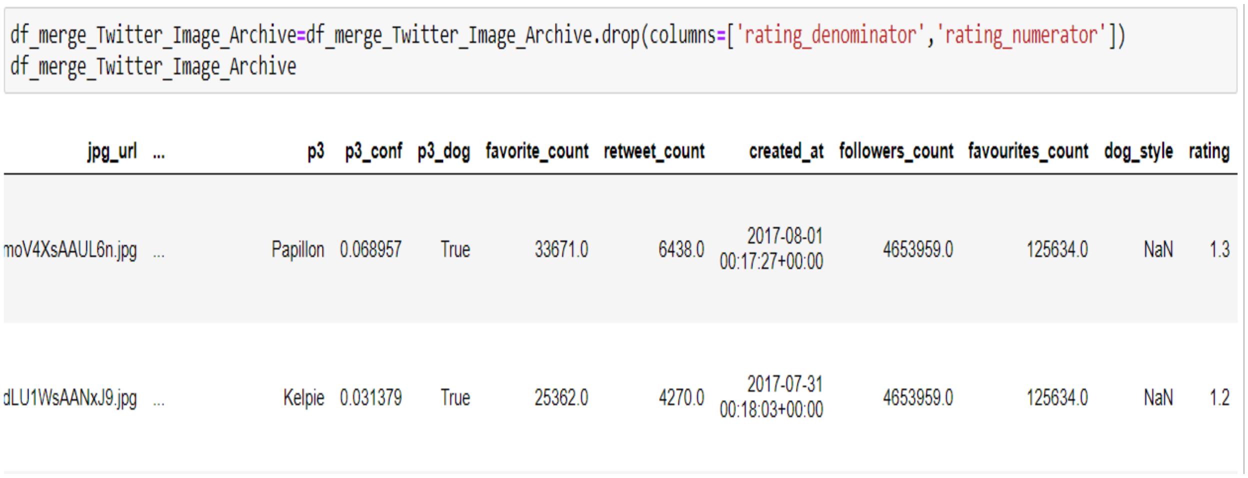
2. Rating_numerator and rating_denominator can be combined into one column
   Solution:
   #Combine rating numerator and rating denominator in to rating column.



```
# combine rating_denominator and rating_numerator in one column
# Divided rating_denominator on rating_numerator and be sure that rating numerator and denominator are in compatible format
df_merge_Twitter_Image_Archive['rating']=df_merge_Twitter_Image_Archive['rating_numerator'] / df_merge_Twitter_Image_Archive['rat
df_merge_Twitter_Image_Archive[['rating_denominator', 'rating_numerator', 'rating']].T
```

	1	2	3	4	5	6	7	8	9	10	...	2345	2346	2347	2348	2350	2351	2352	2353	2354	2355
rating_denominator	10.0	10.0	10.0	10.0	10.0	10.0	10.0	10.0	10.0	10.0	...	10.0	10.0	10.0	10.0	10.0	10.0	10.0	10.0	10.0	10.0
rating_numerator	13.0	12.0	13.0	12.0	13.0	13.0	13.0	13.0	14.0	13.0	...	10.0	8.0	9.0	10.0	10.0	5.0	6.0	9.0	7.0	8.0
rating	1.3	1.2	1.3	1.2	1.3	1.3	1.3	1.3	1.4	1.3	...	1.0	0.8	0.9	1.0	1.0	0.5	0.6	0.9	0.7	0.8

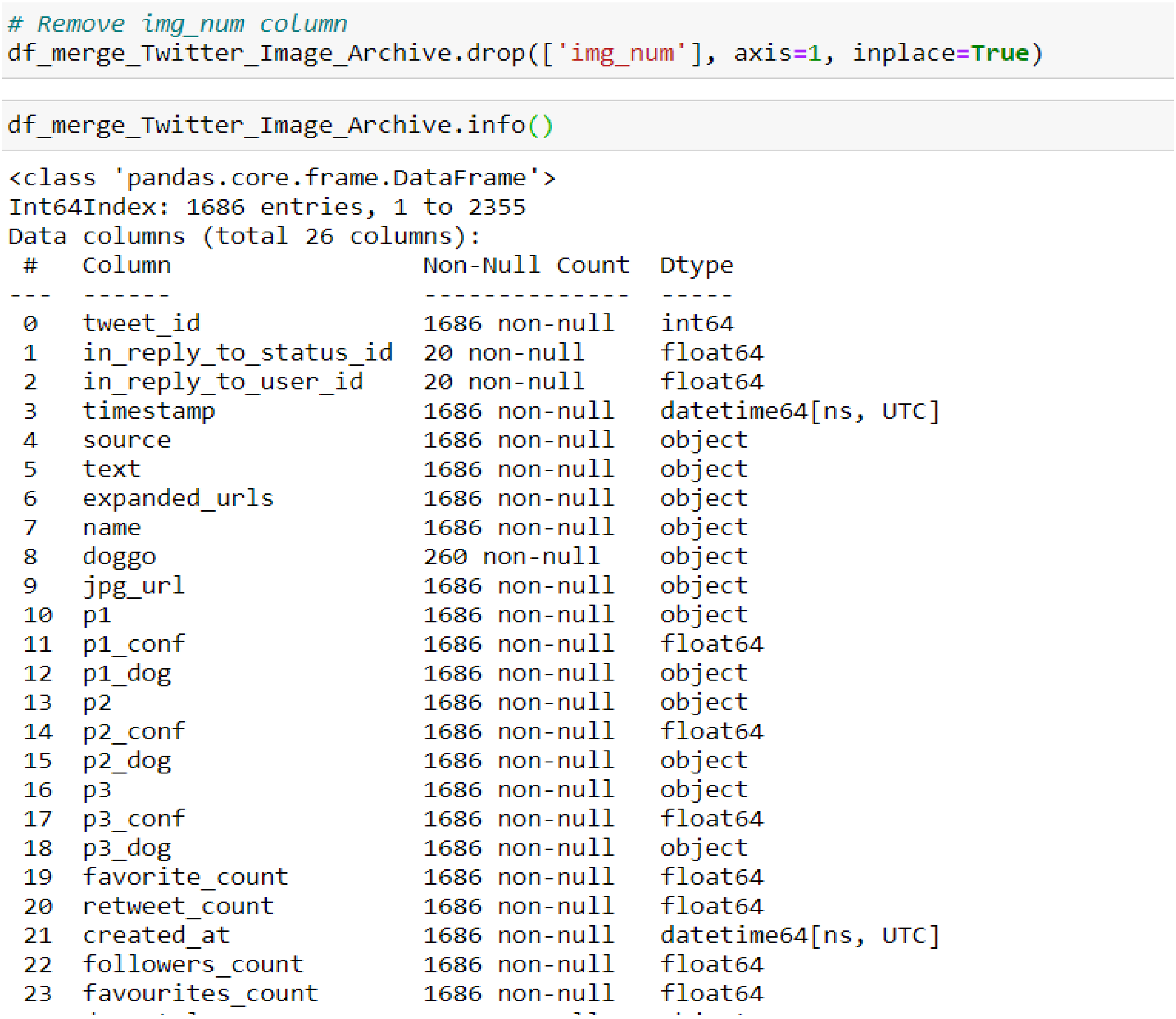
3 rows × 1686 columns



```
df_merge_Twitter_Image_Archive=df_merge_Twitter_Image_Archive.drop(columns=['rating_denominator','rating_numerator'])
df_merge_Twitter_Image_Archive
```

jpg_url	...	p3	p3_conf	p3_dog	favorite_count	retweet_count	created_at	followers_count	favourites_count	dog_style	rating
moV4XsAAUL6n.jpg	...	Papillon	0.068957	True	33671.0	6438.0	2017-08-01 00:17:27+00:00	4653959.0	125634.0	NaN	1.3
dLU1WsAANxJ9.jpg	...	Kelpie	0.031379	True	25362.0	4270.0	2017-07-31 00:18:03+00:00	4653959.0	125634.0	NaN	1.2

3. “img_num" contains integer values ranging from 1 to 4 but only 1 img_url is present (this column semantics is not clear). The column may not have any use here.
   Solution:
   # Remove img_num column.

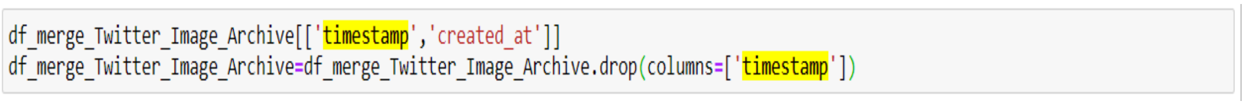


```
# Remove img_num column
df_merge_Twitter_Image_Archive.drop(['img_num'], axis=1, inplace=True)
```

```
df_merge_Twitter_Image_Archive.info()
```

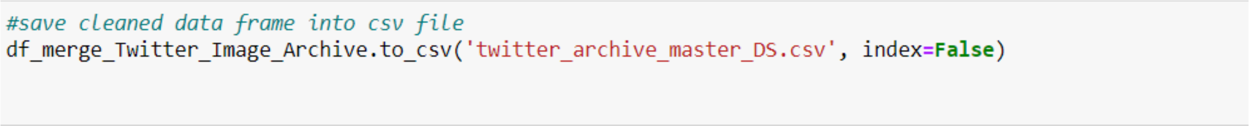
```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1686 entries, 1 to 2355
Data columns (total 26 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   tweet_id                  1686 non-null   int64
 1   in_reply_to_status_id     20 non-null     float64
 2   in_reply_to_user_id       20 non-null     float64
 3   timestamp                 1686 non-null   datetime64[ns, UTC]
 4   source                    1686 non-null   object
 5   text                      1686 non-null   object
 6   expanded_urls             1686 non-null   object
 7   name                      1686 non-null   object
 8   doggo                     260 non-null    object
 9   jpg_url                   1686 non-null   object
 10  p1                        1686 non-null   object
 11  p1_conf                   1686 non-null   float64
 12  p1_dog                    1686 non-null   object
 13  p2                        1686 non-null   object
 14  p2_conf                   1686 non-null   float64
 15  p2_dog                    1686 non-null   object
 16  p3                        1686 non-null   object
 17  p3_conf                   1686 non-null   float64
 18  p3_dog                    1686 non-null   object
 19  favorite_count            1686 non-null   float64
 20  retweet_count             1686 non-null   float64
 21  created_at                1686 non-null   datetime64[ns, UTC]
 22  followers_count           1686 non-null   float64
 23  favourites_count          1686 non-null   float64
```

4. After merging data frames date time in “twitter_archive” has the same values as timestamp column in “tweet_json”.
   Solution:
   # drop one of them.

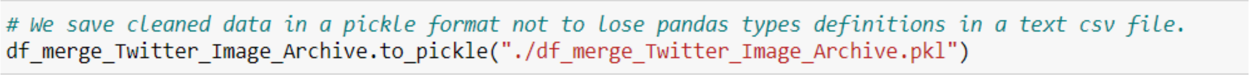


```
df_merge_Twitter_Image_Archive[['timestamp','created_at']]
df_merge_Twitter_Image_Archive=df_merge_Twitter_Image_Archive.drop(columns=['timestamp'])
```

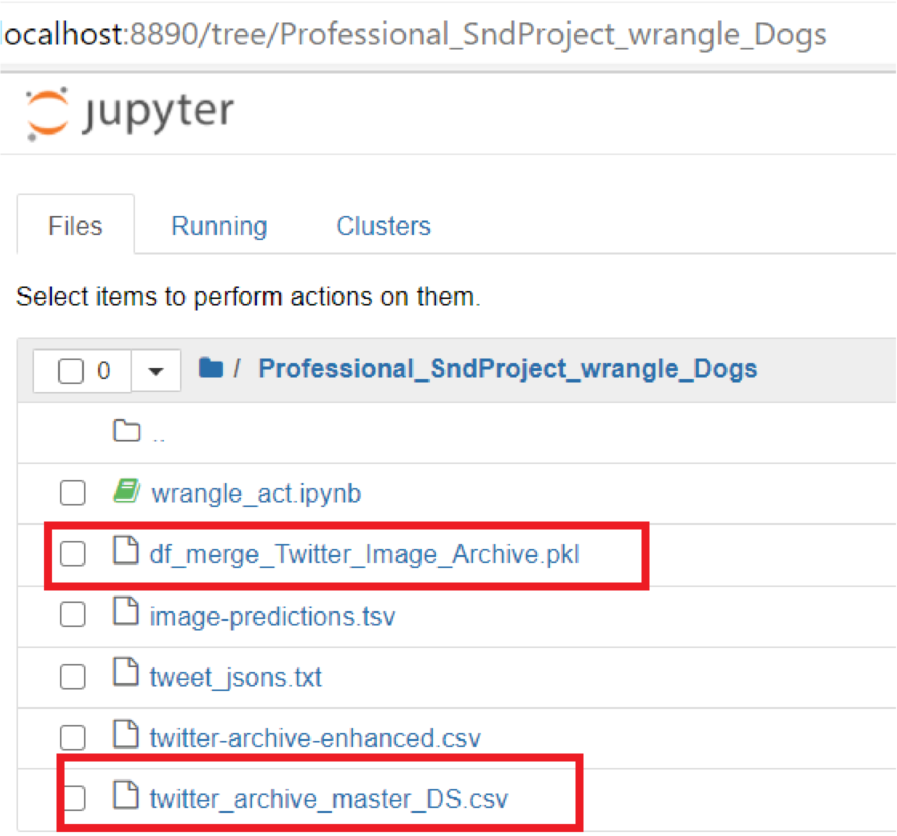
5- Store the clean DataFrame in a CSV file



```
#save cleaned data frame into csv file
df_merge_Twitter_Image_Archive.to_csv('twitter_archive_master_DS.csv', index=False)
```



```
# We save cleaned data in a pickle format not to lose pandas types definitions in a text csv file.
df_merge_Twitter_Image_Archive.to_pickle("./df_merge_Twitter_Image_Archive.pkl")
```



ocalhost:8890/tree/Professional_SndProject_wrangle_Dogs

Select items to perform actions on them.

□ 0 ▾ / Professional_SndProject_wrangle_Dogs

..

□ wrangle_act.ipynb

□ df_merge_Twitter_Image_Archive.pkl

□ image-predictions.tsv

□ tweet_jsons.txt

□ twitter-archive-enhanced.csv

□ twitter_archive_master_DS.csv

6- Analyzing, and Visualizing Data for this Project

After analyze and visualize wrangled data presented **(4) insights and (4) visualization.**