wrangle_act

February 4, 2019

1 Project: Wrangle and Analyze Data

This Udacity project aims to practice a typical data wrangling process. This includes gathering data from a variety of sources, such as CSV-files, TSV-files and an API: The Twitter-API. Then its cleanly- and tidiness will be assessed in order to correct quality issues, converting the data to a tidy dataset. This happens according to the rules of tidiness https://cran.r-project.org/web/packages/tidyr/vignettes/tidy-data.html. Once quality and tidiness is assured, the sources will be combined to one or more master datasets before finally analyzing and visualizing it.

Note: Assessing and wrangling the entirety of the dataset would require more time than provided by Udacity. They, therefore, required to find at least eight quality and two tidiness issues to be assessed and cleaned. Keep in mind, there may be more issues still present.

1.1 Table of Contents


```
<li>Joining Data For Analysis</li>
<li>Analysis</li>
<li>Gathering</li>
<li>References</li>
```

1.2 Gathering Data

```
In [451]: #Importing libraries
          import pandas as pd
          import numpy as np
          import requests
          import json
          import tweepy
          import time
          import matplotlib.pyplot as plt
          import seaborn as sns
          from IPython.display import Image
```

Twitter Archive

In [452]:
```
df_twitter_archive = pd.read_csv('input_raw_data/twitter-archive-enhanced.csv')
```

In [453]:
```
df_twitter_archive.sample(3)
```

Out[453]:
```
                tweet_id  in_reply_to_status_id  in_reply_to_user_id  \
607   798209839306514432                    NaN                  NaN
788   774314403806253056                    NaN                  NaN
1646  683834909291606017                    NaN                  NaN

                      timestamp  \
607   2016-11-14 17:03:50 +0000
788   2016-09-09 18:31:54 +0000
1646  2016-01-04 02:18:42 +0000

                                                 source  \
607   <a href="http://twitter.com/download/iphone" r...
788   <a href="http://twitter.com/download/iphone" r...
1646  <a href="http://twitter.com/download/iphone" r...

                                                   text  retweeted_status_id  \
607   This is Cooper. His bow tie was too heavy for ...                  NaN
788   I WAS SENT THE ACTUAL DOG IN THE PROFILE PIC B...                  NaN
1646  Here we see a faulty pupper. Might need to rep...                  NaN

      retweeted_status_user_id retweeted_status_timestamp  \
607                        NaN                        NaN
788                        NaN                        NaN
1646                       NaN                        NaN

                                          expanded_urls  rating_numerator  \
607   https://twitter.com/dog_rates/status/798209839...                13
788   https://twitter.com/dog_rates/status/774314403...                14
1646  https://twitter.com/dog_rates/status/683834909...                 9

      rating_denominator    name doggo floofer  pupper puppo
607                   10  Cooper  None    None    None  None
788                   10    None  None    None    None  None
1646                  10    None  None    None  pupper  None
```

Image Predictions

In [454]:
```
downloaded_response = requests.get("https://d17h27t6h515a5.cloudfront.net/topher/2017/
with open('input_raw_data/image-predictions.tsv', 'wb') as file:
    file.write(downloaded_response.content)

# Putting contents of the downloaded file into a data frame, using the read_csv-functi
# A tsv-file differs only by the separator from a csv-file.
    df_image_predictions = pd.read_csv('input_raw_data/image-predictions.tsv', sep='\t
```

In [455]: df_image_predictions.head()

Out[455]:

```
             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
```

Twitter-API: Retrieving Additional Post Data

In [ ]:
```
#Authentication
consumer_key = ""
consumer_secret = ""
access_token = ""
access_secret = ""

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

api = tweepy.API(auth, wait_on_rate_limit=True, wait_on_rate_limit_notify=True)
#including handling parameters for timeouts
```

In [ ]:
```
# Testing functionality
test = api.get_status(892420643555336193, wait_on_rate_limit=True, wait_on_rate_limit_no
test != "" # Check
```

In [ ]:
```
# Copying the tweet ids of the archvie to a list, iterating through it
tweet_ids = []
tweet_ids = df_twitter_archive.tweet_id.copy()

# Creating a list, to put to the tweets in (as JSON-objects)
tweets_json_lake = []
```

```
In [ ]: # Measuring elapsed time for retrieving information from the Twitter API.
        start_time_twitter_api = time.time()

        # Iterating through all tweet ids to retrieve the entire post as an object.
        for tweet_id in tweet_ids:
            # Using the try-and-except method to handle non-existing tweet_ids: Cases where no i
            try:
                # Using the get_status function of tweepy to retrieve the post, including the re
                tweet_status = api.get_status(tweet_id, wait_on_rate_limit=True, wait_on_rate_li
                # Appending the tweet (as a json-object to the list of tweets)
                tweets_json_lake.append(tweet_status._json)
                # Empyting variable for the next iteration
                tweet_status = ""
                #print(str(tweet_id)) #Can be used to monitor
            except:
                print("Retrieving information of tweet ID " + str(tweet_id) + " failed.")
```

```
In [ ]: # Creating or opening the file tweets_json.text in write mode, to put the list of tweets
        with open ('input_raw_data/tweet_json.txt', 'w') as outfile:
            json.dump(tweets_json_lake, outfile)

        print("Tweet information stored to file.")
        #Stop timing
        end_time_twitter_api = time.time()
        print("Elapsed time for retrieving tweet information: " + str(end_time_twitter_api - sta
```

Please note: The last five cells were executed only once, in order to retrieve the information from the Twitter API. Once the information was retrieved I did not run those cells anymore, hence the missing output. This due to the fact that running the concern tweet ids throught the API takes 20-30 minutes each time.

```
In [456]: # Loading JSON-file into a list
          with open ('input_raw_data/tweet_json.txt') as json_file:
              tweet_json = json.load(json_file)

          # Converting the list to pandas data frame
          df_tweet_performance = pd.DataFrame(tweet_json)
```

```
In [457]: df_tweet_performance.sample(3)
```

```
Out[457]:      contributors coordinates                       created_at  \
          2329         None        None  Mon Nov 16 01:22:45 +0000 2015
          2261         None        None  Thu Nov 19 20:20:22 +0000 2015
          941          None        None  Sat Jul 09 01:08:47 +0000 2016

                                                   entities  \
          2329  {'hashtags': [], 'symbols': [], 'user_mentions...
          2261  {'hashtags': [], 'symbols': [], 'user_mentions...
          941   {'hashtags': [], 'symbols': [], 'user_mentions...
```

```
                                      extended_entities  favorite_count  \
2329  {'media': [{'id': 666063820255862784, 'id_str'...             465
2261  {'media': [{'id': 667437270979485701, 'id_str'...             450
941   {'media': [{'id': 751583840003584000, 'id_str'...            4638

      favorited   geo                  id              id_str  \
2329      False  None  666063827256086533  666063827256086533
2261      False  None  667437278097252352  667437278097252352
941       False  None  751583847268179968  751583847268179968

                            ...                          quoted_status  \
2329                        ...                                    NaN
2261                        ...                                    NaN
941                         ...                                    NaN

      quoted_status_id quoted_status_id_str  retweet_count retweeted  \
2329               NaN                  NaN            213     False
2261               NaN                  NaN            242     False
941                NaN                  NaN           1196     False

      retweeted_status                                              source  \
2329               NaN  <a href="http://twitter.com/download/iphone" r...
2261               NaN  <a href="http://twitter.com/download/iphone" r...
941                NaN  <a href="http://twitter.com/download/iphone" r...

                                                 text truncated  \
2329  This is the happiest dog you will ever see. Ve...     False
2261  Never seen this breed before. Very pointy pup...     False
941   Please stop sending it pictures that don't eve...     False

                                                 user
2329  {'id': 4196983835, 'id_str': '4196983835', 'na...
2261  {'id': 4196983835, 'id_str': '4196983835', 'na...
941   {'id': 4196983835, 'id_str': '4196983835', 'na...

[3 rows x 30 columns]
```

In [458]:
```
# Dropping all un-needed columns from to data frame, by converting the JSON again, nee
# Needed columns: (tweet)ID, retweet_count, favorite_count
df_tweet_performance = pd.DataFrame(tweet_json, columns=["id", "retweet_count", "favor

# Renaming column id to tweet_id for merging later on.
df_tweet_performance.rename(columns = {'id':'tweet_id'}, inplace=True)

df_tweet_performance.sample(3)
```

Out[458]:
```
                 tweet_id  retweet_count  favorite_count
881   759159934323924993           1256               0
```

```
1171  718540630683709445          1074            2586
2310  666411507551481857           323             438
```

```
In [459]: # It may occur, that when the python notebook ran twice or more times, without first d
          # that duplicates will be created as the API calls are simply appended to the file.
          # The easiest way is to deduplicate the data frame before continuing the analysis.

          # It may also occur, that the python notebook ran at different days, meanwhile the per
          # has increased (i.e. has higher retweet or favorite count). Hence, the row as a whole
          # Checking if there are any duplicate tweet_ids are existent.

          print("Duplicates before cleaning: "+ str(sum(df_tweet_performance.tweet_id.duplicated
          df_tweet_performance.drop_duplicates(subset="tweet_id", inplace=True)
          print("Duplicates after cleaning: "+ str(sum(df_tweet_performance.tweet_id.duplicated(
```

```
Duplicates before cleaning: 0
Duplicates after cleaning: 0
```

1.3 Assessing Data

In this section the data sources are going to be assessed one by one. The first part of each data source outlines the process of assessing and shows my approach. The second part states observations and issues I have found be fixed. Please note: general observations cannot be categorized as issues. However, these are points which need to be adressed prior to the analysis as a kind of "General Preparation".

1.3.1 Table 1: Twitter Archive

```
In [460]: # Checking the overall table structure, data types and for missing values.
          # Missing values in serveral columns. The interpretation of data types are plausible.
          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
```

There are missing values in several columns. The missing values in the columns starting with "in_reply..." and "retweeted_" are of no further interest. The instructions state only "original Tweets" shall be part of the analysis. Furthermore there values missing in the column "expanded_urls".

In [461]:
```
# Deep dive into the column expanded URLs to identify the shortcomings in detail.
df_twitter_archive.expanded_urls.value_counts()
```

Out[461]:
```
https://twitter.com/dog_rates/status/740373189193256964/photo/1,https://twitter.com/do
https://twitter.com/dog_rates/status/782305867769217024/photo/1,https://twitter.com/do
https://twitter.com/dog_rates/status/698195409219559425/photo/1
https://twitter.com/dog_rates/status/759923798737051648/photo/1
https://twitter.com/dog_rates/status/762464539388485633/photo/1,https://twitter.com/do
https://twitter.com/dog_rates/status/681523177663676416/photo/1
https://twitter.com/dog_rates/status/704761120771465216/photo/1,https://twitter.com/do
https://twitter.com/dog_rates/status/775733305207554048/photo/1
https://www.gofundme.com/lolas-life-saving-surgery-funds,https://twitter.com/dog_rates
https://www.loveyourmelon.com/pages/ourstory,https://twitter.com/dog_rates/status/8203
https://twitter.com/dog_rates/status/759447681597108224/photo/1
https://twitter.com/dog_rates/status/786709082849828864/photo/1
https://twitter.com/dog_rates/status/673295268553605120/photo/1
https://twitter.com/dog_rates/status/817827839487737858/video/1
https://twitter.com/dog_rates/status/739238157791694849/video/1
https://twitter.com/dog_rates/status/809220051211603969/photo/1,https://twitter.com/do
https://twitter.com/dog_rates/status/761672994376806400/video/1
https://twitter.com/dog_rates/status/771380798096281600/photo/1,https://twitter.com/do
https://twitter.com/dog_rates/status/773308824254029826/photo/1
https://twitter.com/dog_rates/status/786233965241827333/photo/1
https://twitter.com/dog_rates/status/718631497683582976/photo/1
https://twitter.com/dog_rates/status/756288534030475264/photo/1,https://twitter.com/do
https://twitter.com/dog_rates/status/809920764300447744/photo/1
https://twitter.com/dog_rates/status/767754930266464257/photo/1
https://vine.co/v/ea0OwvPTx9l
https://twitter.com/dog_rates/status/753375668877008896/photo/1
https://twitter.com/dog_rates/status/700143752053182464/photo/1
https://twitter.com/dog_rates/status/683391852557561860/photo/1
https://twitter.com/dog_rates/status/832369877331693569/photo/1
https://twitter.com/dog_rates/status/780931614150983680/photo/1

https://twitter.com/dog_rates/status/798933969379225600/photo/1,https://twitter.com/do
```

```
https://twitter.com/dog_rates/status/744234799360020481/video/1
https://twitter.com/dog_rates/status/672968025906282496/photo/1
https://twitter.com/dog_rates/status/680473011644985345/photo/1
https://twitter.com/dog_rates/status/818536468981415936/photo/1
https://twitter.com/dog_rates/status/684225744407494656/photo/1,https://twitter.com/do
https://twitter.com/dog_rates/status/804026241225523202/photo/1,https://twitter.com/do
https://twitter.com/dog_rates/status/708711088997666817/photo/1,https://twitter.com/do
https://www.petfinder.com/petdetail/34918210,https://twitter.com/dog_rates/status/8320
https://twitter.com/dog_rates/status/668204964695683073/photo/1
https://twitter.com/dog_rates/status/670434127938719744/photo/1
https://twitter.com/dog_rates/status/817171292965273600/photo/1
https://twitter.com/dog_rates/status/674269164442398721/photo/1
https://twitter.com/telegraph/status/832268302944579584
https://twitter.com/dog_rates/status/674291837063053312/photo/1
https://twitter.com/dog_rates/status/831322785565769729/photo/1
https://twitter.com/dog_rates/status/687109925361856513/photo/1,https://twitter.com/do
https://twitter.com/dog_rates/status/769940425801170949/photo/1,https://twitter.com/do
https://twitter.com/dog_rates/status/676617503762681856/photo/1
https://twitter.com/dog_rates/status/675145476954566656/photo/1
https://twitter.com/dog_rates/status/704819833553219584/photo/1
https://twitter.com/dog_rates/status/730573383004487680/photo/1,https://twitter.com/do
https://twitter.com/dog_rates/status/712809025985978368/photo/1
https://twitter.com/dog_rates/status/779056095788752897/photo/1,https://twitter.com/do
https://twitter.com/dog_rates/status/805826884734976000/video/1
https://twitter.com/dog_rates/status/828408677031882754/photo/1
https://twitter.com/dog_rates/status/822489057087389700/photo/1,https://twitter.com/do
https://gofundme.com/ydvmve-surgery-for-jax,https://twitter.com/dog_rates/status/89097
https://twitter.com/dog_rates/status/703769065844768768/photo/1,https://twitter.com/do
https://twitter.com/dog_rates/status/756526248105566208/photo/1
Name: expanded_urls, Length: 2218, dtype: int64
```

In [462]:

```
Image("images/urls_1.PNG")
```

Out[462]:

The screenshot above shows duplicates withtin one single cell, i.e. the url is stated more than one time per row. The redundant information can be extracted.

In [463]:

```
# Checking for duplicated rows across all columns.
sum(df_twitter_archive.duplicated())
```

Out[463]: 0

```
In [464]: # Checking for duplicates across the index-colum tweet ID. Since this column will seru
          # other tables, there should no duplicated values be present.
          sum(df_twitter_archive.tweet_id.duplicated())
```

```
Out[464]: 0
```

```
In [465]: # Checking other relevant columns for duplicated content.
          print(sum(df_twitter_archive.text.duplicated()))
          print(sum(df_twitter_archive.expanded_urls.duplicated()))
```

```
0
137
```

```
In [466]: # Generating output to sample visually. How does the issue exactly look like?
          duplicated_rows_twitter_archive_url = df_twitter_archive["expanded_urls"].duplicated(k
          df_duplicated_tw_archive_url = df_twitter_archive[duplicated_rows_twitter_archive_url]
          df_duplicated_tw_archive_url.sort_values("expanded_urls", inplace = True)
          df_duplicated_tw_archive_url.to_csv("temp_output/duplicated_expanded_urls.csv")

          # Visual assessment revealed that many duplicated image URLs are present in rows with
          # at least one real duplicate which means, cleaning duplicates will make sense anyway.
```

```
/opt/conda/lib/python3.6/site-packages/ipykernel_launcher.py:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#
  after removing the cwd from sys.path.
```

```
In [467]:  Image("images/urls_2.PNG")
```

```
Out[467]:
```

B	C	D	E	F	G	H	I	J	K
tweet_id	in_reply_to_status_id	in_reply_to_user_i	timestam	source	text	retweeted_status_id	retweeted_stat(	retweeter	expanded_urls
8,7806E+17			2017-06-23 0	<a href="http://t	This is Emmy. She was adopted today. Massive round of pupplause for Emmy and her new family. 14/10 for all involved https://t.co/cwtWnHMVpe				https://twitter.com/dog_rates/status/878
8,7913E+17			2017-06-26 0	<a href="http://t	RT @dog_rates: This is Emmy. She was adopted today. Massive round of pupplause for Emmy and her new family. 14/10 for all involved https://â€¦	8,78E+32	4196983835,0	2017-06-23 0	https://twitter.com/dog_rates/status/878
6,6859E+17			2015-11-23 0	<a href="http://v	Never forget this vine. You will not stop watching for at least 15 minutes. This is the second coveted.. 13/10 https://t.co/roqIxCvEB3				https://vine.co/v/ea0OwvPTx9l
7,9177E+17			2016-10-27 2	<a href="http://v	Vine will be deeply missed. This was by far my favorite one. 14/10 https://t.co/roqIxCvEB3				https://vine.co/v/ea0OwvPTx9l

The screenshot above shows the following:

```
<li>Green: duplicate expanded_urls originating from a "retweet" row. Does not matter,Will be cle
<li>Yellow: a real duplicate entry. This issue must be fixed.
</ul>
```

In [468]:
```
# Checking the value consistency among each column. Although this approach can be much
#it is very effective value assessing.

for column in df_twitter_archive:
    print(df_twitter_archive[column].value_counts())
    print('\n')
```

```
749075273010798592    1
741099773336379392    1
798644042770751489    1
825120256414846976    1
769212283578875904    1
700462010979500032    1
780858289093574656    1
699775878809702401    1
880095782870896641    1
760521673607086080    1
776477788987613185    1
691820333922455552    1
715696743237730304    1
714606013974974464    1
760539183865880579    1
813157409116065792    1
676430933382295552    1
743510151680958465    1
837012587749474308    1
833722901757046785    1
818259473185828864    1
670704688707301377    1
667160273090932737    1
674394782723014656    1
672082170312290304    1
670093938074779648    1
759923798737051648    1
809920764300447744    1
805487436403003392    1
838085839343206401    1
                     ..
763956972077010945    1
870308999962521604    1
720775346191278080    1
785927819176054784    1
783347506784731136    1
775733305207554048    1
834209720923721728    1
825026590719483904    1
758405701903519748    1
668986018524233728    1
```

```
690938899477221376    1
667911425562669056    1
754482103782404096    1
713175907180089344    1
669015743032369152    1
672068090318987265    1
816829038950027264    1
683773439333797890    1
674291837063053312    1
837482249356513284    1
767500508068192258    1
773922284943896577    1
673342308415348736    1
886054160059072513    1
748307329658011649    1
715360349751484417    1
666817836334096384    1
794926597468000259    1
673705679337693185    1
700151421916807169    1
Name: tweet_id, Length: 2356, dtype: int64


6.671522e+17    2
8.562860e+17    1
8.131273e+17    1
6.754971e+17    1
6.827884e+17    1
8.265984e+17    1
6.780211e+17    1
6.689207e+17    1
6.658147e+17    1
6.737159e+17    1
7.590995e+17    1
8.862664e+17    1
7.384119e+17    1
7.727430e+17    1
7.468859e+17    1
8.634256e+17    1
6.693544e+17    1
6.914169e+17    1
6.920419e+17    1
6.753494e+17    1
7.291135e+17    1
8.406983e+17    1
6.747400e+17    1
7.501805e+17    1
6.744689e+17    1
```

```
7.638652e+17    1
6.747934e+17    1
8.503288e+17    1
6.747522e+17    1
8.816070e+17    1
                ..
8.380855e+17    1
8.211526e+17    1
8.558616e+17    1
8.558585e+17    1
7.032559e+17    1
6.678065e+17    1
8.018543e+17    1
7.667118e+17    1
6.855479e+17    1
6.717299e+17    1
6.715610e+17    1
6.758457e+17    1
6.924173e+17    1
7.476487e+17    1
8.381455e+17    1
6.903413e+17    1
8.476062e+17    1
8.352460e+17    1
6.813394e+17    1
8.795538e+17    1
6.860340e+17    1
8.571567e+17    1
6.765883e+17    1
7.044857e+17    1
8.707262e+17    1
8.482121e+17    1
6.715449e+17    1
6.936422e+17    1
6.849598e+17    1
7.331095e+17    1
Name: in_reply_to_status_id, Length: 77, dtype: int64


4.196984e+09    47
2.195506e+07     2
7.305050e+17     1
2.916630e+07     1
3.105441e+09     1
2.918590e+08     1
2.792810e+08     1
2.319108e+09     1
1.806710e+08     1
```

```
3.058208e+07     1
2.625958e+07     1
1.943518e+08     1
3.589728e+08     1
8.405479e+17     1
2.894131e+09     1
2.143566e+07     1
2.281182e+09     1
1.648776e+07     1
4.717297e+09     1
2.878549e+07     1
1.582854e+09     1
4.670367e+08     1
4.738443e+07     1
1.361572e+07     1
1.584641e+07     1
2.068372e+07     1
1.637468e+07     1
1.185634e+07     1
1.198989e+09     1
1.132119e+08     1
7.759620e+07     1
Name: in_reply_to_user_id, dtype: int64


2015-12-06 18:56:46 +0000    1
2016-09-27 19:54:58 +0000    1
2016-09-13 23:44:54 +0000    1
2017-03-25 16:45:08 +0000    1
2016-12-14 17:16:53 +0000    1
2016-01-03 00:47:59 +0000    1
2016-02-05 03:18:42 +0000    1
2015-12-19 22:28:09 +0000    1
2016-01-27 01:33:08 +0000    1
2016-03-08 20:09:54 +0000    1
2016-01-14 15:57:26 +0000    1
2016-03-13 23:24:56 +0000    1
2016-10-21 00:53:56 +0000    1
2016-01-30 17:31:20 +0000    1
2017-03-23 18:07:10 +0000    1
2016-11-07 03:14:10 +0000    1
2017-02-06 20:55:28 +0000    1
2015-12-12 03:29:35 +0000    1
2016-07-03 01:41:06 +0000    1
2016-09-15 00:36:55 +0000    1
2016-10-09 23:44:41 +0000    1
2016-11-17 17:04:16 +0000    1
2016-05-08 00:59:46 +0000    1
```

```
2016-03-04 16:06:36 +0000    1
2016-01-21 02:34:07 +0000    1
2016-03-29 23:29:14 +0000    1
2015-11-16 03:55:04 +0000    1
2017-01-02 02:26:09 +0000    1
2015-11-26 01:11:28 +0000    1
2015-12-06 04:34:25 +0000    1
                            ..
2017-01-05 02:09:53 +0000    1
2017-07-04 01:18:17 +0000    1
2016-07-14 01:19:12 +0000    1
2017-05-18 00:50:50 +0000    1
2017-03-04 17:49:08 +0000    1
2017-07-14 22:10:11 +0000    1
2016-02-03 16:49:55 +0000    1
2016-01-12 04:01:58 +0000    1
2016-10-22 00:45:17 +0000    1
2015-11-20 02:08:22 +0000    1
2016-09-30 20:33:43 +0000    1
2016-06-25 15:29:00 +0000    1
2017-07-19 00:47:34 +0000    1
2016-10-03 23:25:55 +0000    1
2016-08-25 00:43:02 +0000    1
2015-11-30 03:06:07 +0000    1
2016-01-02 02:23:45 +0000    1
2016-02-24 02:36:23 +0000    1
2016-11-11 00:03:42 +0000    1
2015-12-08 20:53:11 +0000    1
2015-11-21 02:07:05 +0000    1
2015-11-27 00:43:49 +0000    1
2016-01-06 04:11:43 +0000    1
2016-02-01 03:04:14 +0000    1
2015-12-27 19:22:30 +0000    1
2016-08-30 00:14:12 +0000    1
2017-03-23 18:29:57 +0000    1
2016-12-31 00:08:17 +0000    1
2015-12-02 03:40:57 +0000    1
2017-04-17 00:03:50 +0000    1
Name: timestamp, Length: 2356, dtype: int64


<a href="http://twitter.com/download/iphone" rel="nofollow">Twitter for iPhone</a>     2221
<a href="http://vine.co" rel="nofollow">Vine - Make a Scene</a>                          91
<a href="http://twitter.com" rel="nofollow">Twitter Web Client</a>                       33
<a href="https://about.twitter.com/products/tweetdeck" rel="nofollow">TweetDeck</a>      11
Name: source, dtype: int64
```

This is Robin. She's desperately trying to do me a frighten, but her tongue drastically decrease
This is Reese. He likes holding hands. 12/10 https://t.co/cbLroGCbmh
This is Tedders. He broke his leg saving babies from the Pompeii eruption. 11/10 where's his Pur
This is Lolo. She's America af. Behind in science & math but can say whatever she wants on T
Exotic pup here. Tail long af. Throat looks swollen. Might breathe fire. Exceptionally unfluffy
Meet Snickers. He's adorable. Also comes in t-shirt mode. 12/10 I would aggressively caress Snic
This is Harper. She scraped her elbow attempting a backflip off a tree. Valiant effort tho. 12/1
Yea I can't handle this job anymore your dogs are too adorable. 12/10 https://t.co/N9W5L7BLTm
Say hello to Cooper. His expression is the same wet or dry. Absolute 12/10 but Coop desperately
RT @dog_rates: Say hello to Jack (pronounced "Kevin"). He's a Virgo Episcopalian. Can summon rai
RT @dog_rates: This is Luna. It's her first time outside and a bee stung her nose. Completely h*
Here we have a mixed Asiago from the Galápagos Islands. Only one ear working. Big fan of marijua
This is Cali. She arrived preassembled. Convenient af. 12/10 appears to be rather h*ckin pettabl
Meet Oliviér. He takes killer selfies. Has a dog of his own. It leaps at random & can't bark
This is Jackson. There's nothing abnormal about him. Just your average really good dog. 10/10 ht
This is Shelby. She finds stuff to put on her head for attention. It works really well. 12/10 ta
Seriously guys? Again? We only rate dogs. Please stop submitting other things like this super go
This is Pippa. She managed to start the car but is waiting for you to buckle up before driving.
This is Linda. She just looked up and saw you glancing at your neighboring classmate's test. 10/
Happy Friday here's a sleepy pupper 12/10 https://t.co/eBcqv9SPkY
This is Oscar. He has legendary eyebrows and he h*ckin knows it. Curly af too. 12/10 would hug p
This is Bernie. He just touched a boob for the first time. 10/10 https://t.co/whQKMygnK6
Meet Koda. He's large. Looks very soft. Great bangs. Powerful owner. 11/10 would pet the hell ou
This is Bell. She likes holding hands. 12/10 would definitely pet with other hand https://t.co/B
We only rate dogs. Please stop sending in non-canines like this Alaskan Flop Turtle. This is ver
This is Bubbles. He kinda resembles a fish. Always makes eye contact with u no matter what. Snea
This is Daniel. He's a neat pup. Exotic af. Custom paws. Leaps unannounced. Would totally pet. 7
Meet Charlie. He likes to kiss all the big milk dogs with the rad earrings. Passionate af. 10/10
No no no this is all wrong. The Walmart had to have run into the dog driving the car. 10/10 some
Say hello to Clarence. He's a western Alkaline Pita. Very proud of himself for dismembering his

Meet Glenn. Being in public scares him. Frighteningly relatable. 12/10 keep hangin in there Glen
This is Jeffrey. He wasn't prepared to execute such advanced barkour. Still 11/10 would totally
Here we have a Japanese Irish Setter. Lost eye in Vietnam (?). Big fan of relaxing on stair. 8/1
Guys this really needs to stop. We've been over this way too many times. This is a giraffe. We o
@serial @MrRoles OH MY GOD I listened to all of season 1 during a single road trip. I love you g
Meet Sonny. He's an in-home movie critic. That is his collection. He's very proud of it. 12/10 h
This is Alexanderson. He's got a weird ass birth mark. Dreadful at fetch. Won't eat kibble. 3/10
This is Gus. He likes to be close to you, which is good because you want to be close to Gus. 12/
This is Django. He accidentally opened the front facing camera. Did him quite the frighten. 12/1
I've never wanted to go to a camp more in my entire life. 12/10 for all on board https://t.co/wJ
This is Curtis. He's an Albino Haberdasher. Terrified of dandelions. They really spook him up. 1
When your roommate eats your leftover Chili's but you pretend it's no big deal cuz you fat anywa
Here we see a nifty leaping pupper. Feet look deadly. Sad that the holidays are over. 9/10 unden
This dog doesn't know how to stairs. Quite tragic really. 9/10 get it together pup https://t.co/
OMIGOD 12/10 https://t.co/SVMF4Frf1w
This is Pavlov. His floatation device has failed him. He's quite pupset about it. 11/10 would re
RT @KennyFromDaBlok: 14/10 h*ckin good hats. will wear daily @dog_rates https://t.co/rHLoU5gS30

```
This is Arnie. He's a Nova Scotian Fridge Floof. Rare af. 12/10 https://t.co/lprdOylVpS
This is Zoe. She was trying to stealthily take a picture of you but you just noticed. 9/10 not s
Meet Reggie. He's going for the world record. Must concentrate. Focus up pup. 11/10 we all belie
This is Huck. He's addicted to caffeine. Hope it's not too latte to seek help. 11/10 stay strong
"Ello this is dog how may I assist" ...10/10 https://t.co/jeAENpjH7L
Guys this is getting so out of hand. We only rate dogs. This is a Galapagos Speed Panda. Pls onl
This is Damon. The newest presidential candidate for 2016. 10/10 he gets my vote https://t.co/Z5
This is Napolean. He's a Raggedy East Nicaraguan Zoom Zoom. Runs on one leg. Built for deception
Can you spot Toby the guilty pupper? 7/10 would be higher but he made quite the mess shredding h
After so many requests... here you go.\n\nGood dogg. 420/10 https://t.co/yfAAo1gdeY
I don't know any of the backstory behind this picture but for some reason I'm crying. 13/10 for
Evolution of a pupper yawn featuring Max. 12/10 groundbreaking stuff https://t.co/t8Y4x9DmVD
This is Dakota. He's just saying hi. That's all. 12/10 someone wave back https://t.co/1tWe5zZoHv
Name: text, Length: 2356, dtype: int64


7.757333e+17    1
7.507196e+17    1
6.742918e+17    1
6.833919e+17    1
8.269587e+17    1
8.780576e+17    1
7.320056e+17    1
7.186315e+17    1
6.732953e+17    1
7.914070e+17    1
8.447048e+17    1
7.862340e+17    1
8.685523e+17    1
7.504293e+17    1
8.327664e+17    1
6.690004e+17    1
6.873173e+17    1
7.638376e+17    1
7.815247e+17    1
8.092201e+17    1
8.000650e+17    1
8.174239e+17    1
8.001414e+17    1
7.909461e+17    1
7.867091e+17    1
8.406323e+17    1
8.688804e+17    1
7.869631e+17    1
7.733088e+17    1
8.222448e+17    1
               ..
6.816941e+17    1
```

```
7.899865e+17    1
7.939622e+17    1
6.800555e+17    1
7.128090e+17    1
8.164506e+17    1
6.769365e+17    1
6.675487e+17    1
8.083449e+17    1
7.626999e+17    1
6.678667e+17    1
8.479710e+17    1
6.820881e+17    1
8.352641e+17    1
7.761133e+17    1
7.902771e+17    1
6.675484e+17    1
7.677549e+17    1
8.782815e+17    1
8.663350e+17    1
7.399792e+17    1
7.403732e+17    1
8.395493e+17    1
7.001438e+17    1
8.482894e+17    1
7.848260e+17    1
7.806013e+17    1
8.305833e+17    1
7.047611e+17    1
7.331095e+17    1
Name: retweeted_status_id, Length: 181, dtype: int64


4.196984e+09    156
4.296832e+09      2
5.870972e+07      1
6.669901e+07      1
4.119842e+07      1
7.475543e+17      1
7.832140e+05      1
7.266347e+08      1
4.871977e+08      1
5.970642e+08      1
4.466750e+07      1
1.228326e+09      1
7.992370e+07      1
2.488557e+07      1
7.874618e+17      1
3.638908e+08      1
```

```
5.128045e+08      1
8.117408e+08      1
1.732729e+09      1
1.960740e+07      1
1.547674e+08      1
3.410211e+08      1
7.124572e+17      1
2.804798e+08      1
1.950368e+08      1
Name: retweeted_status_user_id, dtype: int64


2017-02-09 01:27:41 +0000    1
2016-09-02 18:03:10 +0000    1
2016-10-07 00:06:50 +0000    1
2017-04-01 00:36:55 +0000    1
2017-01-04 01:05:59 +0000    1
2017-02-12 01:04:29 +0000    1
2016-09-28 00:46:20 +0000    1
2015-12-24 00:58:27 +0000    1
2016-09-14 17:40:06 +0000    1
2015-12-06 01:56:44 +0000    1
2017-03-04 00:21:08 +0000    1
2016-12-17 00:38:52 +0000    1
2017-01-20 17:00:46 +0000    1
2017-03-11 18:35:42 +0000    1
2015-11-27 19:11:49 +0000    1
2016-03-01 20:11:59 +0000    1
2016-02-19 18:24:26 +0000    1
2016-09-06 23:56:05 +0000    1
2016-06-07 00:36:02 +0000    1
2016-10-27 16:06:04 +0000    1
2017-06-09 16:22:42 +0000    1
2017-02-28 18:43:57 +0000    1
2017-01-07 20:18:46 +0000    1
2016-07-06 15:54:42 +0000    1
2016-09-26 17:55:00 +0000    1
2016-01-13 16:56:30 +0000    1
2017-04-24 02:13:14 +0000    1
2017-05-02 00:04:57 +0000    1
2016-08-15 16:22:20 +0000    1
2016-09-11 21:34:30 +0000    1
                            ..
2015-11-20 03:41:59 +0000    1
2016-07-27 00:40:12 +0000    1
2016-09-13 16:30:07 +0000    1
2016-07-25 15:23:28 +0000    1
2016-07-13 23:48:51 +0000    1
```

```
2016-08-01 16:42:51 +0000    1
2016-01-06 20:16:44 +0000    1
2015-12-23 00:45:35 +0000    1
2016-06-10 00:39:48 +0000    1
2016-08-05 21:19:27 +0000    1
2017-06-23 01:10:23 +0000    1
2016-01-08 05:00:14 +0000    1
2015-12-06 00:17:55 +0000    1
2016-12-17 22:43:27 +0000    1
2016-08-22 16:06:54 +0000    1
2015-11-20 01:06:48 +0000    1
2016-10-08 18:41:19 +0000    1
2016-05-19 01:38:16 +0000    1
2017-04-20 18:14:33 +0000    1
2017-04-26 02:37:47 +0000    1
2017-01-11 17:01:16 +0000    1
2016-07-13 01:34:21 +0000    1
2017-05-27 19:39:34 +0000    1
2015-12-02 03:40:57 +0000    1
2017-01-13 17:00:21 +0000    1
2015-12-29 04:31:49 +0000    1
2017-07-19 00:47:34 +0000    1
2017-05-18 01:17:25 +0000    1
2015-12-24 16:00:30 +0000    1
2016-11-22 00:10:52 +0000    1
Name: retweeted_status_timestamp, Length: 181, dtype: int64


https://twitter.com/dog_rates/status/740373189193256964/photo/1,https://twitter.com/dog_rates/st
https://twitter.com/dog_rates/status/782305867769217024/photo/1,https://twitter.com/dog_rates/st
https://twitter.com/dog_rates/status/698195409219559425/photo/1
https://twitter.com/dog_rates/status/759923798737051648/photo/1
https://twitter.com/dog_rates/status/762464539388485633/photo/1,https://twitter.com/dog_rates/st
https://twitter.com/dog_rates/status/681523177663676416/photo/1
https://twitter.com/dog_rates/status/704761120771465216/photo/1,https://twitter.com/dog_rates/st
https://twitter.com/dog_rates/status/775733305207554048/photo/1
https://www.gofundme.com/lolas-life-saving-surgery-funds,https://twitter.com/dog_rates/status/83
https://www.loveyourmelon.com/pages/ourstory,https://twitter.com/dog_rates/status/820314633777 06
https://twitter.com/dog_rates/status/759447681597108224/photo/1
https://twitter.com/dog_rates/status/786709082849828864/photo/1
https://twitter.com/dog_rates/status/673295268553605120/photo/1
https://twitter.com/dog_rates/status/817827839487737858/video/1
https://twitter.com/dog_rates/status/739238157791694849/video/1
https://twitter.com/dog_rates/status/809220051211603969/photo/1,https://twitter.com/dog_rates/st
https://twitter.com/dog_rates/status/761672994376806400/video/1
https://twitter.com/dog_rates/status/771380798096281600/photo/1,https://twitter.com/dog_rates/st
https://twitter.com/dog_rates/status/773308824254029826/photo/1
https://twitter.com/dog_rates/status/786233965241827333/photo/1
```

```
https://twitter.com/dog_rates/status/718631497683582976/photo/1
https://twitter.com/dog_rates/status/756288534030475264/photo/1,https://twitter.com/dog_rates/st
https://twitter.com/dog_rates/status/809920764300447744/photo/1
https://twitter.com/dog_rates/status/767754930266464257/photo/1
https://vine.co/v/ea0OwvPTx9l
https://twitter.com/dog_rates/status/753375668877008896/photo/1
https://twitter.com/dog_rates/status/700143752053182464/photo/1
https://twitter.com/dog_rates/status/683391852557561860/photo/1
https://twitter.com/dog_rates/status/832369877331693569/photo/1
https://twitter.com/dog_rates/status/780931614150983680/photo/1

https://twitter.com/dog_rates/status/798933969379225600/photo/1,https://twitter.com/dog_rates/st
https://twitter.com/dog_rates/status/744234799360020481/video/1
https://twitter.com/dog_rates/status/672968025906282496/photo/1
https://twitter.com/dog_rates/status/680473011644985345/photo/1
https://twitter.com/dog_rates/status/818536468981415936/photo/1
https://twitter.com/dog_rates/status/684225744407494656/photo/1,https://twitter.com/dog_rates/st
https://twitter.com/dog_rates/status/804026241225523202/photo/1,https://twitter.com/dog_rates/st
https://twitter.com/dog_rates/status/708711088997666817/photo/1,https://twitter.com/dog_rates/st
https://www.petfinder.com/petdetail/34918210,https://twitter.com/dog_rates/status/83203280282048
https://twitter.com/dog_rates/status/668204964695683073/photo/1
https://twitter.com/dog_rates/status/670434127938719744/photo/1
https://twitter.com/dog_rates/status/817171292965273600/photo/1
https://twitter.com/dog_rates/status/674269164442398721/photo/1
https://twitter.com/telegraph/status/832268302944579584
https://twitter.com/dog_rates/status/674291837063053312/photo/1
https://twitter.com/dog_rates/status/831322785565769729/photo/1
https://twitter.com/dog_rates/status/687109925361856513/photo/1,https://twitter.com/dog_rates/st
https://twitter.com/dog_rates/status/769940425801170949/photo/1,https://twitter.com/dog_rates/st
https://twitter.com/dog_rates/status/676617503762681856/photo/1
https://twitter.com/dog_rates/status/675145476954566656/photo/1
https://twitter.com/dog_rates/status/704819833553219584/photo/1
https://twitter.com/dog_rates/status/730573383004487680/photo/1,https://twitter.com/dog_rates/st
https://twitter.com/dog_rates/status/712809025985978368/photo/1
https://twitter.com/dog_rates/status/779056095788752897/photo/1,https://twitter.com/dog_rates/st
https://twitter.com/dog_rates/status/805826884734976000/video/1
https://twitter.com/dog_rates/status/828408677031882754/photo/1
https://twitter.com/dog_rates/status/822489057087389700/photo/1,https://twitter.com/dog_rates/st
https://gofundme.com/ydvmve-surgery-for-jax,https://twitter.com/dog_rates/status/890971913173991
https://twitter.com/dog_rates/status/703769065844768768/photo/1,https://twitter.com/dog_rates/st
https://twitter.com/dog_rates/status/756526248105566208/photo/1
Name: expanded_urls, Length: 2218, dtype: int64


12      558
11      464
10      461
13      351
```

```
9       158
8       102
7        55
14       54
5        37
6        32
3        19
4        17
1         9
2         9
420       2
0         2
15        2
75        2
80        1
20        1
24        1
26        1
44        1
50        1
60        1
165       1
84        1
88        1
144       1
182       1
143       1
666       1
960       1
1776      1
17        1
27        1
45        1
99        1
121       1
204       1
Name: rating_numerator, dtype: int64


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


None        745
a            55
Charlie      12
Cooper       11
Oliver       11
Lucy         11
Tucker       10
Penny        10
Lola         10
Bo            9
Winston       9
Sadie         8
the           8
an            7
Toby          7
Buddy         7
Daisy         7
Bailey        7
Milo          6
Koda          6
Stanley       6
Jax           6
Oscar         6
Scout         6
Rusty         6
Jack          6
Dave          6
Leo           6
Bella         6
George        5
            ...
JD            1
Hero          1
Noosh         1
Daniel        1
Barclay       1
```

```
Rumpole       1
Grey          1
Buckley       1
Sandra        1
Lucky         1
Brian         1
Willie        1
Chuq          1
Jessiga       1
Jeffri        1
his           1
Puff          1
Erik          1
Butters       1
Pancake       1
Einstein      1
Schnozz       1
Aqua          1
Emanuel       1
Rupert        1
Zoe           1
Strider       1
Dwight        1
Flash         1
Molly         1
Name: name, Length: 957, dtype: int64


None     2259
doggo      97
Name: doggo, dtype: int64


None       2346
floofer      10
Name: floofer, dtype: int64


None      2099
pupper     257
Name: pupper, dtype: int64


None     2326
puppo      30
Name: puppo, dtype: int64
```

In [469]:
```
# Detailed check values of column rating_numerator, since many values appear to be odd
# Filtering outlying values.
df_numerator_check = df_twitter_archive[(df_twitter_archive.rating_numerator > 20) | (

# Checking how many rows are outlying.
df_numerator_check.info()
# 26 rows have rare numerator values.
# Exporting them to CSV in order to check the details in a spreadsheet application.
df_numerator_check.to_csv('temp_output/numerator_check.csv')
# Conclusion: outlying values are either just typos, incorrect extracted or due to exa
#these rows to increase the reliability of the analysis while keeping time efforts at
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 26 entries, 188 to 2074
Data columns (total 17 columns):
tweet_id                      26 non-null int64
in_reply_to_status_id         6 non-null float64
in_reply_to_user_id           6 non-null float64
timestamp                     26 non-null object
source                        26 non-null object
text                          26 non-null object
retweeted_status_id           1 non-null float64
retweeted_status_user_id      1 non-null float64
retweeted_status_timestamp    1 non-null object
expanded_urls                 22 non-null object
rating_numerator              26 non-null int64
rating_denominator            26 non-null int64
name                          26 non-null object
doggo                         26 non-null object
floofer                       26 non-null object
pupper                        26 non-null object
puppo                         26 non-null object
dtypes: float64(4), int64(3), object(10)
memory usage: 3.7+ KB
```

In [470]:
```
# Detailed check values of column rating_denominator, since some values appear to be o
# Filtering outlying values.
df_denominator_check = df_twitter_archive[(df_twitter_archive.rating_denominator > 10)

# Checking how many rows are outlying.
df_denominator_check.info()
# 23 rows have rare denominator values.
# Exporting them to CSV in order to check the details in a spreadsheet application.
df_denominator_check.to_csv('temp_output/denominator_check.csv')
# Conclusion: outlying values are either just typos, incorrect extracted or due to exa
#these rows to increase the reliability of the analysis while keeping time efforts at
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 23 entries, 313 to 2335
Data columns (total 17 columns):
tweet_id                      23 non-null int64
in_reply_to_status_id         5 non-null float64
in_reply_to_user_id           5 non-null float64
timestamp                     23 non-null object
source                        23 non-null object
text                          23 non-null object
retweeted_status_id           1 non-null float64
retweeted_status_user_id      1 non-null float64
retweeted_status_timestamp    1 non-null object
expanded_urls                 19 non-null object
rating_numerator              23 non-null int64
rating_denominator            23 non-null int64
name                          23 non-null object
doggo                         23 non-null object
floofer                       23 non-null object
pupper                        23 non-null object
puppo                         23 non-null object
dtypes: float64(4), int64(3), object(10)
memory usage: 3.2+ KB
```

```
In [471]: # Detailed check of the columns source. The above analysis implied, there might be odd
          df_twitter_archive.source.value_counts()
```

```
Out[471]: <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 [472]: # There are no values present, the source might be an Android app.
          # It could have been, that the users may have used mostly the iOS app for posting.
          # the "<a href ...>" code parts do not add any value.
          # Conclusion: it does not matter too much. This columns appears to not add value from
          # The column shall be dropped.
```

The observations and issues to this dataset as listed below. This serves as a summary.

General Observations The following points cannot be categorized as issues. These point are of a general nature and need to be adressed and handled prior to the analysis. The handling is described in the section "General Preparations" in the "Cleaning Data" section.

```
    <li>According to the instructions tweets beyond August 1st, 2017 shall not be included in th
<li>Missing values in columns in_reply_to_status_id, in_reply_to_user_id. Since these values are
<li>Missing values in columns retweeted_status_id, retweeted_status_user_id, retweeted_status_ti
<li>According to the instructions retweets and replies will not be part of the analysis. Rows wi
 <li>The fact that rating numerators are (mostly) greater than the denominators does not need to
```

Quality Issues

```
<li>Missing values in column expanded_urls. Every post should be related to a rated dog picture
<li>Everypost should be related to rated dog picture (there should be a value for every post id/
<li>Duplicate expanded_urls present: most duplicated image URLs are present in rows with retweet

<li>The columns rating_numerator and rating_denominator have partly rare and illogical values. A
        <li>The rating numerator and denominator represent a single variable stored in different
       <li>Column source: There are no values implying any of these tweets have been posted by a
```

Tidiness Issues

```
<b>Optional, in case analysis would be extended to replies and retweets:</b>
```

The column set of "in_reply_to_status_id, in_reply_to_user_id" and "retweeted_status_id, retweeted_status_user_id, retweeted_status_timestamp" form different sets of observational units are can therefore be separated from the rest of the data by storing each set into a new table. The column tweet_id can be used as a 1 to 1 or identifier.

Optional if analysis would need to cover the special classifications "doggo, floofer, pupper, puppo"

```
    The columns doggo, floofer, pupper, puppo are untidy as the values are stored in column name
</li>
```

Note to the reviewer: I left it integer because this type consumes less storage capacity and this there faster in handling.

1.3.2 Table 2: Image Predictions

The following section describes the process of assessing this data set. The issues are stated thereafter.

```
In [473]: # Checking the overall structure visually by 30 randomly drawn rows.
          df_image_predictions.sample(30)
```

```
Out[473]:                 tweet_id                                          jpg_url  \
          946   704499785726889984    https://pbs.twimg.com/media/CcbiOUGWoAA4fwg.jpg
          1605  800443802682937345    https://pbs.twimg.com/media/CsVO7ljW8AAckRD.jpg
          684   683852578183077888    https://pbs.twimg.com/media/CX2ISqSWYAAEtCF.jpg
          1168  735991953473572864    https://pbs.twimg.com/media/CjbExRKUoAAs089.jpg
          805   691756958957883396    https://pbs.twimg.com/media/CZmdSD8UcAAnY5R.jpg
          574   678446151570427904    https://pbs.twimg.com/media/CWpTLOYWsAEDhcU.jpg
          1215  743510151680958465  https://pbs.twimg.com/ext_tw_video_thumb/74350...
          1073  716802964044845056    https://pbs.twimg.com/media/CfKYfeBXIAAopp2.jpg
          19    666273097616637952    https://pbs.twimg.com/media/CT8T1mtUwAA3aqm.jpg
          1159  733828123016450049    https://pbs.twimg.com/media/Ci8UxxcW0AYgHDh.jpg
          1354  760252756032651264    https://pbs.twimg.com/media/Coz12OLWgAADdys.jpg
          1709  818145370475810820    https://pbs.twimg.com/media/C1qi26rW8AMaj9K.jpg
          553   677573743309385728    https://pbs.twimg.com/media/CWc5uVPXIAErLYr.jpg
```

1592	798673117451325440	https://pbs.twimg.com/media/CV_cnjHWUAADc-c.jpg
1739	822462944365645825	https://pbs.twimg.com/media/C2n5rUUXEAIXAtv.jpg
212	670037189829525505	https://pbs.twimg.com/media/CUxzQ-nWIAAgJUm.jpg
450	674739953134403584	https://pbs.twimg.com/media/CV0oaHFW4AA9Coi.jpg
1847	839290600511926273	https://pbs.twimg.com/media/C6XBt9XXEAEEW9U.jpg
1797	831552930092285952	https://pbs.twimg.com/media/C4pE-I0WQAABveu.jpg
1587	798209839306514432	https://pbs.twimg.com/media/CxPPnCYWIAAo_ao.jpg
1650	809220051211603969	https://pbs.twimg.com/media/CzrtWDbWEAAmIhy.jpg
1468	778990705243029504	https://pbs.twimg.com/media/Cs-H5uhWcAAiNY9.jpg
323	671874878652489728	https://pbs.twimg.com/media/CVL6op1WEAAUFE7.jpg
956	705428427625635840	https://pbs.twimg.com/media/CcovaMUXIAApFDl.jpg
1612	801538201127157760	https://pbs.twimg.com/media/Cx-itFWWIAAZu7l.jpg
1394	767500508068192258	https://pbs.twimg.com/media/Cqa1ofnXEAAG0yn.jpg
1880	846874817362120707	https://pbs.twimg.com/media/C8C0JYHW0AAy-7u.jpg
167	668986018524233728	https://pbs.twimg.com/media/CUi3PIrWoAAPvPT.jpg
1253	748307329658011649	https://pbs.twimg.com/media/CmKFi-FXEAAeI37.jpg
1562	793500921481273345	https://pbs.twimg.com/media/CwMU34YWIAAz1nU.jpg

	img_num	p1	p1_conf	p1_dog	\
946	1	Chihuahua	0.376541	True	
1605	1	mousetrap	0.777468	False	
684	1	toy_poodle	0.551352	True	
1168	2	cocker_spaniel	0.961643	True	
805	1	Saint_Bernard	0.342571	True	
574	1	Staffordshire_bullterrier	0.284492	True	
1215	1	sea_lion	0.859046	False	
1073	2	malinois	0.619577	True	
19	1	Italian_greyhound	0.176053	True	
1159	2	beagle	0.472324	True	
1354	1	radio_telescope	0.155279	False	
1709	1	golden_retriever	0.621931	True	
553	2	patio	0.535070	False	
1592	1	dough	0.806757	False	
1739	3	Pomeranian	0.960199	True	
212	1	pot	0.273767	False	
450	1	Dandie_Dinmont	0.175915	True	
1847	1	web_site	0.670892	False	
1797	1	Chihuahua	0.257415	True	
1587	1	Pekinese	0.524583	True	
1650	1	Pomeranian	0.819511	True	
1468	2	cocker_spaniel	0.715351	True	
323	1	china_cabinet	0.996031	False	
956	1	Chihuahua	0.774792	True	
1612	1	Pembroke	0.550506	True	
1394	1	chow	0.483228	True	
1880	2	Shetland_sheepdog	0.450539	True	
167	1	doormat	0.976103	False	
1253	2	paddle	0.589066	False	

```
1562        2           golden_retriever  0.326122    True

                       p2   p2_conf  p2_dog                            p3  \
946            Siamese_cat  0.098057   False             Labrador_retriever
1605           black_widow  0.093940   False                    paddlewheel
684                  teddy  0.180678   False                miniature_poodle
1168            toy_poodle  0.011547    True    soft-coated_wheaten_terrier
805                  boxer  0.289096    True                        Pembroke
574             Rottweiler  0.189434    True  American_Staffordshire_terrier
1215                   tub  0.020405   False                    hippopotamus
1073              Leonberg  0.118089    True                    bull_mastiff
19             toy_terrier  0.111884    True                        basenji
1159          Walker_hound  0.121779    True                   Saint_Bernard
1354                   dam  0.154515   False                           crane
1709    Labrador_retriever  0.364997    True                        redbone
553          folding_chair  0.080419   False                   parallel_bars
1592                bakery  0.027907   False                     French_loaf
1739               Samoyed  0.023056    True                     Maltese_dog
212                   tray  0.092888   False                        doormat
450    black-footed_ferret  0.096534   False                     toy_poodle
1847               monitor  0.101565   False                         screen
1797              Pembroke  0.161442    True                  French_bulldog
1587              Shih-Tzu  0.102931    True                     Pomeranian
1650               Samoyed  0.141241    True                        Pembroke
1468    Labrador_retriever  0.207056    True                       Chihuahua
323   entertainment_center  0.001986   False                        bookcase
956                  quilt  0.073079   False                        Pembroke
1612              Cardigan  0.306612    True                Shetland_sheepdog
1394      golden_retriever  0.165063    True                  Norfolk_terrier
1880              papillon  0.187928    True                           collie
167              Chihuahua  0.005640    True                  Norfolk_terrier
1253                 shovel  0.038062   False                    mountain_tent
1562    Labrador_retriever  0.219904    True         Chesapeake_Bay_retriever

       p3_conf  p3_dog
946   0.085211    True
1605  0.017492   False
684   0.164095    True
1168  0.004903    True
805   0.076463    True
574   0.189430    True
1215  0.013095   False
1073  0.066508    True
19    0.111152    True
1159  0.114640    True
1354  0.098040   False
1709  0.003971    True
553   0.034796   False
```

```
1592  0.018189   False
1739  0.008945    True
212   0.050728   False
450   0.064145    True
1847  0.075306   False
1797  0.092143    True
1587  0.097893    True
1650  0.013455    True
1468  0.028519    True
323   0.001652   False
956   0.022365    True
1612  0.054230    True
1394  0.060173    True
1880  0.140068    True
167   0.003913    True
1253  0.029203   False
1562  0.163366    True
```

In [474]:
```
df_image_predictions.info()
# This table looks good at first glance. No missing values, data types are all correct
#invalid values.
# However, the predication values are partly stored within the column names and is the
```

```
<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 [475]:
```
# Checking for duplicated rows across all columns.
sum(df_image_predictions.duplicated())
```

Out[475]: 0

In [476]:
```
# Checking for duplicates across the index-column tweet ID. Since this column will ser
#other tables, there should ideally no duplicated values be present.
sum(df_image_predictions.tweet_id.duplicated())
```

Out[476]: 0

In [477]:
```
# Checking other relevant columns for duplicated content.
print(sum(df_image_predictions.jpg_url.duplicated()))
```

```
66
```

In [478]:
```
duplicated_rows_image_predictions_jpg_url = df_image_predictions["jpg_url"].duplicated
df_duplicated_image_predictions_jpg_url = df_image_predictions[duplicated_rows_image_p
df_duplicated_image_predictions_jpg_url.sort_values("jpg_url", inplace = True)
df_duplicated_image_predictions_jpg_url.to_csv("temp_output/duplicated_image_urls_imag
```

```
/opt/conda/lib/python3.6/site-packages/ipykernel_launcher.py:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#
  This is separate from the ipykernel package so we can avoid doing imports until
```

In [479]:
```
# Programmatic assessment revealed there are 66 duplicates entries present, however, e
# The visual (see screenshot below) assessment lets me assume each pair has identical
# to be deleted. That way, chances are higher that an image prediction is available fo
# twitter archive dataset.
```

In [480]:
```
Image("images/image_urls_1.PNG")
```

Out[480]:

	B	C	D	E	F	G	H	I	J	K	L	M
	tweet_id	jpg_url	img_num	p1	p1_conf	p1_dog	p2	p2_conf	p2_dog	p3	p3_conf	p3_dog
480	6,75E+17	https://pbs.twimg.com/ext_tw_video_thumb/675354114423808004/pu/img/qL1R_nGLqa6lmkOx.jpg	1	upright	0.303415	False	golden_retri	0.181351	True	Brittany_spa	0.162083999	True
1297	7,52E+17	https://pbs.twimg.com/ext_tw_video_thumb/675354114423808004/pu/img/qL1R_nGLqa6lmkOx.jpg	1	upright	0.303415	False	golden_retri	0.181351	True	Brittany_spa	0.162083999	True
1864	8,43E+17	https://pbs.twimg.com/ext_tw_video_thumb/807106774843039744/pu/img/8XZg1xW35Xp2J6JW.jpg	1	Chihuahua	0.50537	True	Pomeranian	0.120358	True	toy_terrier	0.0770081	True
1641	8,07E+17	https://pbs.twimg.com/ext_tw_video_thumb/807106774843039744/pu/img/8XZg1xW35Xp2J6JW.jpg	1	Chihuahua	0.50537	True	Pomeranian	0.120358	True	toy_terrier	0.0770081	True
1703	8,17E+17	https://pbs.twimg.com/ext_tw_video_thumb/815965888126062592/pu/img/JleSw4wRhgKDWQj5.jpg	1	Tibetan_mas	0.506312	True	Tibetan_terr	0.29569	True	otterhound	0.0362507	True
1691	8,16E+17	https://pbs.twimg.com/ext_tw_video_thumb/815965888126062592/pu/img/JleSw4wRhgKDWQj5.jpg	1	Tibetan_mas	0.506312	True	Tibetan_terr	0.29569	True	otterhound	0.0362507	True
1705	8,17E+17	https://pbs.twimg.com/ext_tw_video_thumb/817423809049493505/pu/img/5OFW0yueFu9oTUiQ.jpg	1	ice_bear	0.3362	False	Samoyed	0.201358	True	Eskimo_dog	0.186789	True
1858	8,42E+17	https://pbs.twimg.com/ext_tw_video_thumb/817423809049493505/pu/img/5OFW0yueFu9oTUiQ.jpg	1	ice_bear	0.3362	False	Samoyed	0.201358	True	Eskimo_dog	0.186789	True
1715	8,19E+17	https://pbs.twimg.com/media/C12whDoVEAALRxa.jpg	1	standard_po	0.351308	True	toy_poodle	0.271929	True	Tibetan_terr	0.0947592	True
1718	8,19E+17	https://pbs.twimg.com/media/C12whDoVEAALRxa.jpg	1	standard_po	0.351308	True	toy_poodle	0.271929	True	Tibetan_terr	0.0947592	True
1716	8,19E+17	https://pbs.twimg.com/media/C12x-JTVIAAzdfl.jpg	4	prison	0.907083	False	palace	0.0200891	False	umbrella	0.00784954	False
1717	8,19E+17	https://pbs.twimg.com/media/C12x-JTVIAAzdfl.jpg	4	prison	0.907083	False	palace	0.0200891	False	umbrella	0.00784954	False

In [481]:
```
# Check, whether the predications for each duplicate are identical:
columns_to_check = ["p1","p1_conf","p1_dog","p2","p2_conf","p2_dog","p3","p3_conf","p3
for column in columns_to_check:
    print(sum(df_image_predictions[columns_to_check].duplicated()))

# Same amounts of duplicated among all image prediction value columns impliesthat each
# equally. Duplicates will be removed.
```

```
66
66
66
```

```
66
66
66
66
66
66
```

In [482]:
```
# Checking the value consistency among each column. Although this approach can be much
# it is very effective in value assessing

for column in df_image_predictions:
    print(df_image_predictions[column].value_counts())
    print('\n')
```

```
685532292383666176    1
826598365270007810    1
692158366030913536    1
714606013974974464    1
715696743237730304    1
776477788987613185    1
772114945936949249    1
699775878809702401    1
780858289093574656    1
700462010979500032    1
732726085725589504    1
738883359779196928    1
798644042770751489    1
743510151680958465    1
837012587749474308    1
833722901757046785    1
668620235289837568    1
842765311967449089    1
685315239903100929    1
673686845050527744    1
680473011644985345    1
666051853826850816    1
675853064436391936    1
693231807727280129    1
705475953783398401    1
829449946868879360    1
759923798737051648    1
667160273090932737    1
680934982542561280    1
743545585370791937    1
                     ..
794926597468000259    1
776113305656188928    1
```

```
825026590719483904    1
834209720923721728    1
775733305207554048    1
669564461267722241    1
879492040517615616    1
720775346191278080    1
666362758909284353    1
750506206503038976    1
693155686491000832    1
793601777308463104    1
740373189193256964    1
754482103782404096    1
881536004380872706    1
843604394117681152    1
748307329658011649    1
759846353224826880    1
885984800019947520    1
773922284943896577    1
666345417576210432    1
837482249356513284    1
812781120811126785    1
870804317367881728    1
790698755171364864    1
816829038950027264    1
847971574464610304    1
713175907180089344    1
670338931251150849    1
700151421916807169    1
Name: tweet_id, Length: 2075, dtype: int64


https://pbs.twimg.com/media/CvyVxQRWEAAdSZS.jpg                                                    2
https://pbs.twimg.com/media/Ck2d7tJWUAEPTL3.jpg                                                    2
https://pbs.twimg.com/media/DFDw2tyUQAAAFke.jpg                                                    2
https://pbs.twimg.com/media/CVgdFjNWEAAxmbq.jpg                                                    2
https://pbs.twimg.com/media/CU3mITUWIAAfyQS.jpg                                                    2
https://pbs.twimg.com/media/Ct72q9jWcAAhlnw.jpg                                                    2
https://pbs.twimg.com/media/CsVO7ljW8AAckRD.jpg                                                    2
https://pbs.twimg.com/ext_tw_video_thumb/807106774843039744/pu/img/8XZg1xW35Xp2J6JW.jpg    2
https://pbs.twimg.com/ext_tw_video_thumb/815965888126062592/pu/img/JleSw4wRhgKDWQj5.jpg    2
https://pbs.twimg.com/media/CiyHLocU4AI2pJu.jpg                                                    2
https://pbs.twimg.com/tweet_video_thumb/CeBym7oXEAEWbEg.jpg                                       2
https://pbs.twimg.com/media/CwJR1okWIAA6XMp.jpg                                                    2
https://pbs.twimg.com/media/ChK1tdBWwAQ1flD.jpg                                                    2
https://pbs.twimg.com/media/Cq9guJ5WgAADfpF.jpg                                                    2
https://pbs.twimg.com/media/CzG425nWgAAnP7P.jpg                                                    2
https://pbs.twimg.com/media/CpmyNumW8AAAJGj.jpg                                                    2
https://pbs.twimg.com/media/CcG07BYW0AErrC9.jpg                                                    2
```

```
https://pbs.twimg.com/media/DA7iHL5U0AA1OQo.jpg                                                    2
https://pbs.twimg.com/ext_tw_video_thumb/817423809049493505/pu/img/5OFW0yueFu9oTUiQ.jpg    2
https://pbs.twimg.com/media/CUN4Or5UAAAa5K4.jpg                                                    2
https://pbs.twimg.com/media/CVuQ2LeUsAAIe3s.jpg                                                    2
https://pbs.twimg.com/media/CsrjryzWgAAZY00.jpg                                                    2
https://pbs.twimg.com/media/Cs_DYr1XEAA54Pu.jpg                                                    2
https://pbs.twimg.com/media/C4bTH6nWMAAX_bJ.jpg                                                    2
https://pbs.twimg.com/media/C4KHj-nWQAA3poV.jpg                                                    2
https://pbs.twimg.com/media/CwS4aqZXUAAe3IO.jpg                                                    2
https://pbs.twimg.com/media/CkNjahBXAAQ2kWo.jpg                                                    2
https://pbs.twimg.com/media/CwiuEJmW8AAZnit.jpg                                                    2
https://pbs.twimg.com/media/CWza7kpWcAAdYLc.jpg                                                    2
https://pbs.twimg.com/media/CiibOMzUYAA9Mxz.jpg                                                    2
                                                                                                  ..
https://pbs.twimg.com/media/CV6P1lnWIAAUQHk.jpg                                                    1
https://pbs.twimg.com/media/CnsIT0WWcAAul8V.jpg                                                    1
https://pbs.twimg.com/media/DFg_2PVW0AEHN3p.jpg                                                    1
https://pbs.twimg.com/media/CUjSRNCXAAQ6Y_8.jpg                                                    1
https://pbs.twimg.com/media/CwI2XCvXEAEO8mc.jpg                                                    1
https://pbs.twimg.com/media/C2tugXLXgAArJO4.jpg                                                    1
https://pbs.twimg.com/media/CUNr4-7UwAAg2lq.jpg                                                    1
https://pbs.twimg.com/media/CUOb_gUUkAACXdS.jpg                                                    1
https://pbs.twimg.com/media/CYdbvwjWcAEtjYu.jpg                                                    1
https://pbs.twimg.com/media/CYVIToGWQAAEZ_y.jpg                                                    1
https://pbs.twimg.com/media/CnWGCpdWgAAWZTI.jpg                                                    1
https://pbs.twimg.com/media/Cm4phTpWcAAgLsr.jpg                                                    1
https://pbs.twimg.com/media/C-WcS4MXoAADrBU.jpg                                                    1
https://pbs.twimg.com/media/CcpaoR9WAAAKlJJ.jpg                                                    1
https://pbs.twimg.com/media/Cas5h-wWcAA3nAc.jpg                                                    1
https://pbs.twimg.com/media/CUOPYI5UcAAj_nO.jpg                                                    1
https://pbs.twimg.com/media/CZHM6OBWIAA4AY4.jpg                                                    1
https://pbs.twimg.com/media/CYTUhn7WkAEXocW.jpg                                                    1
https://pbs.twimg.com/media/CUImtzEVAAAZNJo.jpg                                                    1
https://pbs.twimg.com/media/CO3K2-VWIAAK1iV.jpg                                                    1
https://pbs.twimg.com/media/CU8Z-OxXAAA-sd2.jpg                                                    1
https://pbs.twimg.com/media/COkFzOQUoAAt6yb.jpg                                                    1
https://pbs.twimg.com/media/CUjETvDVAAI8LIy.jpg                                                    1
https://pbs.twimg.com/ext_tw_video_thumb/760289324994879489/pu/img/3ItvBEoo4aebPfvr.jpg    1
https://pbs.twimg.com/media/Cs6r_-kVIAALh1p.jpg                                                    1
https://pbs.twimg.com/media/ChJO9YaWYAEL0zC.jpg                                                    1
https://pbs.twimg.com/media/CVmE_fAWIAAlDhU.jpg                                                    1
https://pbs.twimg.com/media/CXCGVXyWsAAAVHE.jpg                                                    1
https://pbs.twimg.com/media/C3B9ypNWEAM1bVs.jpg                                                    1
https://pbs.twimg.com/media/COkTjqIXgAAqpRi.jpg                                                    1
Name: jpg_url, Length: 2009, dtype: int64


1    1780
```

```
2     198
3      66
4      31
Name: img_num, dtype: int64


golden_retriever             150
Labrador_retriever           100
Pembroke                      89
Chihuahua                     83
pug                           57
chow                          44
Samoyed                       43
toy_poodle                    39
Pomeranian                    38
malamute                      30
cocker_spaniel                30
French_bulldog                26
Chesapeake_Bay_retriever      23
miniature_pinscher            23
seat_belt                     22
Staffordshire_bullterrier     20
German_shepherd               20
Siberian_husky                20
Cardigan                      19
web_site                      19
beagle                        18
teddy                         18
Shetland_sheepdog             18
Eskimo_dog                    18
Maltese_dog                   18
Shih-Tzu                      17
Lakeland_terrier              17
Rottweiler                    17
Italian_greyhound             16
kuvasz                        16
                            ...
boathouse                      1
mud_turtle                     1
hammer                         1
panpipe                        1
water_bottle                   1
lawn_mower                     1
barbell                        1
convertible                    1
four-poster                    1
terrapin                       1
swab                           1
```

```
sundial                        1
candle                         1
walking_stick                  1
hotdog                         1
picket_fence                   1
nail                           1
suit                           1
Japanese_spaniel               1
rapeseed                       1
rain_barrel                    1
coffee_mug                     1
bald_eagle                     1
marmot                         1
washer                         1
three-toed_sloth               1
sunglasses                     1
lacewing                       1
sandbar                        1
water_buffalo                  1
Name: p1, Length: 378, dtype: int64


0.366248    2
0.713293    2
0.375098    2
0.636169    2
0.611525    2
0.420463    2
0.581403    2
0.403698    2
0.530104    2
0.254856    2
0.346545    2
0.721188    2
0.677408    2
0.907083    2
0.243529    2
0.505370    2
0.593858    2
0.372202    2
0.274637    2
0.600276    2
0.506312    2
0.615163    2
0.556595    2
0.995143    2
0.809197    2
0.964929    2
```

```
0.777468    2
0.336200    2
0.617389    2
0.786089    2
           ..
0.483228    1
0.556524    1
0.176423    1
0.318981    1
0.733025    1
0.730152    1
0.436023    1
0.479008    1
0.162935    1
0.897162    1
0.320420    1
0.999833    1
0.995873    1
0.523206    1
0.942911    1
0.537652    1
0.672791    1
0.952258    1
0.855959    1
0.665578    1
0.841265    1
0.668164    1
0.946828    1
0.714719    1
0.352946    1
0.713102    1
0.765266    1
0.491022    1
0.905334    1
1.000000    1
Name: p1_conf, Length: 2006, dtype: int64


True     1532
False     543
Name: p1_dog, dtype: int64


Labrador_retriever                104
golden_retriever                   92
Cardigan                           73
Chihuahua                          44
Pomeranian                         42
```

```
Chesapeake_Bay_retriever          41
French_bulldog                    41
toy_poodle                        37
cocker_spaniel                    34
miniature_poodle                  33
Siberian_husky                    33
beagle                            28
Pembroke                          27
collie                            27
Eskimo_dog                        27
kuvasz                            26
Italian_greyhound                 22
Pekinese                          21
American_Staffordshire_terrier    21
chow                              20
Samoyed                           20
malinois                          20
toy_terrier                       20
miniature_pinscher                20
Norwegian_elkhound                19
Boston_bull                       19
Staffordshire_bullterrier         18
pug                               17
Irish_terrier                     17
Shih-Tzu                          16
                                 ...
cowboy_boot                        1
nail                               1
bannister                          1
printer                            1
horse_cart                         1
patio                              1
handkerchief                       1
breastplate                        1
mailbox                            1
triceratops                        1
sarong                             1
European_gallinule                 1
volcano                            1
pickup                             1
cab                                1
Bernese_mountain_dog               1
armadillo                          1
leafhopper                         1
cowboy_hat                         1
solar_dish                         1
hay                                1
timber_wolf                        1
```

```
wallaby                              1
laptop                               1
tray                                 1
quail                                1
dock                                 1
cockroach                            1
seashore                             1
platypus                             1
Name: p2, Length: 405, dtype: int64


0.069362    3
0.027907    2
0.193654    2
0.271929    2
0.003143    2
0.197021    2
0.347609    2
0.151047    2
0.052724    2
0.153126    2
0.119256    2
0.227150    2
0.057091    2
0.149950    2
0.025119    2
0.165930    2
0.190503    2
0.012763    2
0.181351    2
0.325106    2
0.020089    2
0.172844    2
0.142204    2
0.152445    2
0.052956    2
0.099984    2
0.093940    2
0.252706    2
0.140798    2
0.130611    2
           ..
0.083513    1
0.100988    1
0.038062    1
0.317368    1
0.256433    1
0.057883    1
```

```
0.098354    1
0.250014    1
0.088474    1
0.178088    1
0.053008    1
0.052396    1
0.165655    1
0.182538    1
0.074962    1
0.120530    1
0.169758    1
0.119745    1
0.090938    1
0.071665    1
0.139346    1
0.036575    1
0.073101    1
0.118181    1
0.000077    1
0.138331    1
0.254884    1
0.090644    1
0.219323    1
0.016301    1
Name: p2_conf, Length: 2004, dtype: int64


True     1553
False     522
Name: p2_dog, dtype: int64


Labrador_retriever                79
Chihuahua                         58
golden_retriever                  48
Eskimo_dog                        38
kelpie                            35
kuvasz                            34
chow                              32
Staffordshire_bullterrier         32
cocker_spaniel                    31
beagle                            31
Pekinese                          29
Pomeranian                        29
toy_poodle                        29
Chesapeake_Bay_retriever          27
Great_Pyrenees                    27
Pembroke                          27
```

```
malamute                          26
French_bulldog                    26
American_Staffordshire_terrier    24
Cardigan                          23
pug                               23
basenji                           21
toy_terrier                       20
bull_mastiff                      20
Siberian_husky                    19
Shetland_sheepdog                 17
Boston_bull                       17
doormat                           16
Lakeland_terrier                  16
boxer                             16
                                  ..
hand_blower                        1
green_lizard                       1
bannister                          1
African_chameleon                  1
cup                                1
mitten                             1
cloak                              1
goldfish                           1
pool_table                         1
rotisserie                         1
triceratops                        1
chimpanzee                         1
stinkhorn                          1
plastic_bag                        1
maze                               1
pickup                             1
traffic_light                      1
red_wolf                           1
cab                                1
passenger_car                      1
prairie_chicken                    1
mountain_tent                      1
golfcart                           1
European_fire_salamander           1
bonnet                             1
rock_crab                          1
paintbrush                         1
wallet                             1
seashore                           1
partridge                          1
Name: p3, Length: 408, dtype: int64
```

0.094759	2
0.035711	2
0.000428	2
0.044660	2
0.162084	2
0.077130	2
0.116806	2
0.146427	2
0.003956	2
0.039012	2
0.137186	2
0.109677	2
0.096435	2
0.157028	2
0.157524	2
0.118199	2
0.016497	2
0.100842	2
0.151024	2
0.005410	2
0.026364	2
0.186789	2
0.223263	2
0.087355	2
0.121523	2
0.041476	2
0.071436	2
0.046403	2
0.014858	2
0.003330	2
	..
0.098207	1
0.013206	1
0.106014	1
0.044002	1
0.127037	1
0.001404	1
0.001274	1
0.008451	1
0.039808	1
0.056548	1
0.001310	1
0.016663	1
0.079883	1
0.122701	1
0.047397	1
0.068297	1
0.031673	1

```
0.047601    1
0.019516    1
0.051835    1
0.078720    1
0.143328    1
0.000436    1
0.003383    1
0.109454    1
0.024007    1
0.132820    1
0.002099    1
0.083643    1
0.033835    1
Name: p3_conf, Length: 2006, dtype: int64


True     1499
False     576
Name: p3_dog, dtype: int64
```

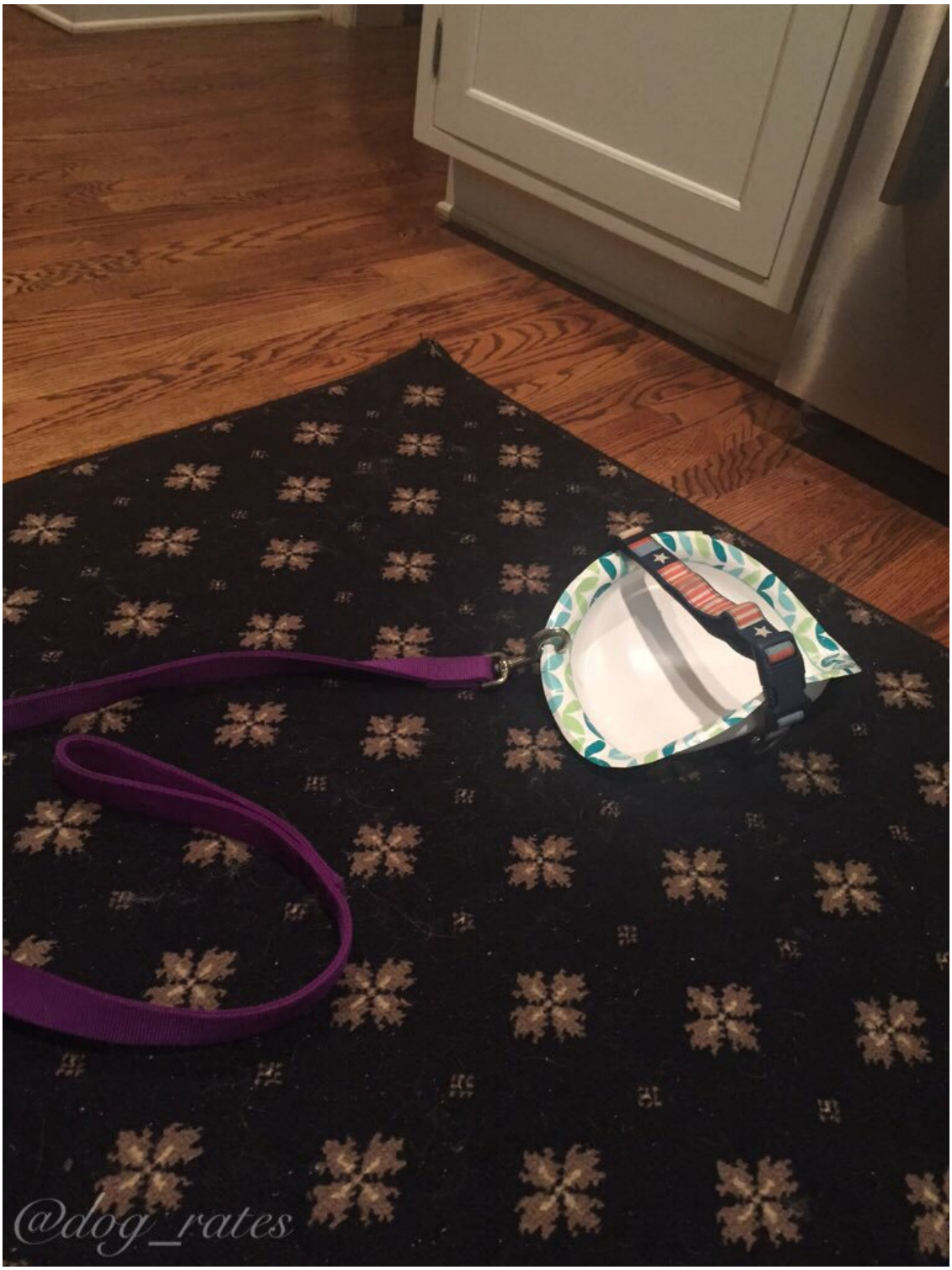
In [483]:
```
# Checking the false values in p_dog columns.
# Assumption: when all three values are false the might not be any dog on the picture
df_p_dog_test = df_image_predictions[(df_image_predictions['p1_dog'] == False) &
                                     (df_image_predictions['p2_dog'] == False) &
                                     (df_image_predictions['p3_dog'] == False)]

df_p_dog_test.sample(5)
```

Out[483]:
```
                 tweet_id                                          jpg_url  \
1444  775733305207554048  https://pbs.twimg.com/media/CsP1UvaW8AExVSA.jpg
1946  862457590147678208  https://pbs.twimg.com/media/C_gQmaTUMAAPYSS.jpg
471   675135153782571009  https://pbs.twimg.com/media/CV6P1lnWIAAUQHk.jpg
1500  783839966405230592  https://pbs.twimg.com/media/CuDCSM-XEAAJw1W.jpg
1207  742161199639494656  https://pbs.twimg.com/media/CkyvqnNWYAQxQY1.jpg

      img_num                 p1   p1_conf  p1_dog            p2   p2_conf  \
1444        1  long-horned_beetle  0.613852   False            ox  0.029473
1946        1        home_theater  0.496348   False  studio_couch  0.167256
471         1               stove  0.587507   False    rotisserie  0.051713
1500        1               quilt  0.333739   False   Siamese_cat  0.136245
1207        1             balloon  0.990736   False  punching_bag  0.004754

      p2_dog                 p3   p3_conf  p3_dog
1444   False  rhinoceros_beetle  0.027806   False
1946   False       barber_chair  0.052625   False
```

```
471    False          microwave  0.020725   False
1500   False   three-toed_sloth  0.117464   False
1207   False          parachute  0.000436   False
```

In [484]:
```
#Checking some random image URLs for the contents:
Image("https://pbs.twimg.com/media/CVKVM3NW4AAdi1e.jpg")
```

Out[484]:




```
In [485]: Image("https://pbs.twimg.com/media/CiIuBwCUgAAAGbz.jpg")

Out[485]:
```

```
In [486]: # Assumption: when at least one values is true, chances are high, there is a dog shown
          df_p_dog_test_2 = df_image_predictions[(df_image_predictions['p1_dog'] == True) |
                                                 (df_image_predictions['p2_dog'] == True) |
```

```
                                        (df_image_predictions['p3_dog'] == True)]

df_p_dog_test_2.sample(5)
```

Out[486]:

```
                tweet_id                                          jpg_url  \
158   668872652652679168  https://pbs.twimg.com/media/CUhQIAhXAAA2j7u.jpg
399   673686845050527744  https://pbs.twimg.com/media/CVlqi_AXIAASlcD.jpg
2061  889638837579907072  https://pbs.twimg.com/media/DFihzFfXsAYGDPR.jpg
1783  829011960981237760  https://pbs.twimg.com/media/C4E99ygWcAAQpPs.jpg
218   670073503555706880  https://pbs.twimg.com/media/CUyUSuWXIAAZKYF.jpg

      img_num              p1   p1_conf  p1_dog              p2   p2_conf  \
158         1           teddy  0.413379   False          pillow  0.325623
399         1        Pekinese  0.185903    True      guinea_pig  0.172951
2061        1  French_bulldog  0.991650    True           boxer  0.002129
1783        2           boxer  0.312221    True       dalmatian  0.244040
218         1        malamute  0.601886    True  Siberian_husky  0.340106

      p2_dog                        p3   p3_conf  p3_dog
158    False        miniature_schnauzer  0.035537    True
399    False                        pug  0.166183    True
2061    True  Staffordshire_bullterrier  0.001498    True
1783    True                      conch  0.130273   False
218     True                 Eskimo_dog  0.050041    True
```

In [487]:

```
Image("https://pbs.twimg.com/media/C5d0QtvXMAI_7uz.jpg")
```

Out[487]:

```
In [488]: Image("https://pbs.twimg.com/media/CuV8yfxXEAAUlye.jpg")
```

Out[488]:

Please note: I personally checked many more URLs per case. These four just demonstrate the manual assessment.

```
In [489]: # Mixed capitalization and missused separators among the prediction values.
```

```
df_image_predictions.p1.sample(10)
```

Out[489]:
```
784                 badger
1605             mousetrap
82                  borzoi
2054        French_bulldog
878                    pug
835                 teddy
1260           tiger_shark
1670              Doberman
1673    Norwegian_elkhound
1407      golden_retriever
Name: p1, dtype: object
```

In [490]:
```
# The Values in column image_counts do not imply clearly what they are meant for. If k
# However, this is not clear to me. I therefore categorize it as a quality issue. The
df_image_predictions.img_num.value_counts()
```

Out[490]:
```
1    1780
2     198
3      66
4      31
Name: img_num, dtype: int64
```

Quality Issues

```
<li>Duplicated image URLs: assessment revealed there are 66 duplicates entries present, however,
<li>The false values of the columns p1 to p4_dog imply not all images are showing dogs. Manual s
<li>The columns p1, p2, p3 have mixed capitalization. Each first letter of a word will be capita
    <li>In the columns p1, p2, p3 single words are separated by an underscore. Underscores will
    <li>The column img_num will be dropped. The reason, why this is an issue of quality: the val
```

Tidiness Issues

```
<li>The columns "p1, p1_conf, p1_dog, etc." have parts of their values in the column titles. The
     <li>The fact that the data of the same observational unit (i.e. the Tweet iself) is stored
</ol>
```

1.4 Cleaning Data

1.4.1 Table 1: Twitter Archive

General preparations and handling the observations

In [491]:
```
# Creating a copy of the dataset.
df_twitter_archive_cleaned = df_twitter_archive.copy()
```

Handling Observation 1 Define

Delete tweeds beyond 2018-08-01. According to the instructions, no image predictions are avaiblable beyond that date.

Code

```
In [492]: df_twitter_archive_cleaned = df_twitter_archive_cleaned[df_twitter_archive_cleaned['ti
```

Test

```
In [493]: df_twitter_archive_cleaned.shape # Check
```

```
Out[493]: (2354, 17)
```

Handling Observation 2 to 4 Define

Deleting rows with retweets and responses. For structual and logical reasons, they will not be part of analysis. Once the values are dropped, the columns can also be dropped.

Code

```
In [494]: df_twitter_archive_cleaned = df_twitter_archive_cleaned[df_twitter_archive_cleaned['in
          df_twitter_archive_cleaned = df_twitter_archive_cleaned[df_twitter_archive_cleaned['re

          columns_to_be_dropped = ["in_reply_to_status_id", "in_reply_to_user_id", "retweeted_st
          df_twitter_archive_cleaned.drop(columns_to_be_dropped, axis = 1, inplace=True)
          columns_to_be_dropped = []
```

Test

```
In [495]: print(df_twitter_archive_cleaned.info()) #Check
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2095 entries, 2 to 2355
Data columns (total 12 columns):
tweet_id              2095 non-null int64
timestamp             2095 non-null object
source                2095 non-null object
text                  2095 non-null object
expanded_urls         2092 non-null object
rating_numerator      2095 non-null int64
rating_denominator    2095 non-null int64
name                  2095 non-null object
doggo                 2095 non-null object
floofer               2095 non-null object
pupper                2095 non-null object
puppo                 2095 non-null object
dtypes: int64(3), object(9)
memory usage: 212.8+ KB
None
```

Fixing Quality Issues

Fixing Quality Issue 1 Define
Deleting rows with empty values in column expanded_urls.
Code

```
In [496]: df_twitter_archive_cleaned['expanded_urls'].dropna(how = "all", inplace=True)
```

Test

```
In [497]: print(df_twitter_archive_cleaned.info())
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2095 entries, 2 to 2355
Data columns (total 12 columns):
tweet_id              2095 non-null int64
timestamp             2095 non-null object
source                2095 non-null object
text                  2095 non-null object
expanded_urls         2092 non-null object
rating_numerator      2095 non-null int64
rating_denominator    2095 non-null int64
name                  2095 non-null object
doggo                 2095 non-null object
floofer               2095 non-null object
pupper                2095 non-null object
puppo                 2095 non-null object
dtypes: int64(3), object(9)
memory usage: 212.8+ KB
None
```

There are still only 2092 instead of 2095 values counted in that column. Maybe pandas got the datatype wrong, while loading from the CSV-File.
Detailed check:

```
In [498]: sum(df_twitter_archive_cleaned.expanded_urls == "")
```

```
Out[498]: 0
```

This appears odd. I therefore like ensure, the column is assigned the string datatype.

```
In [499]: df_twitter_archive_cleaned['expanded_urls'] = df_twitter_archive_cleaned['expanded_url
```

Reverse Test

```
In [500]: sum(df_twitter_archive_cleaned.expanded_urls != "")
```

```
Out[500]: 2095
```

Final test

```
In [501]: print(df_twitter_archive_cleaned.info()) #Check
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2095 entries, 2 to 2355
Data columns (total 12 columns):
tweet_id              2095 non-null int64
timestamp             2095 non-null object
source                2095 non-null object
text                  2095 non-null object
expanded_urls         2092 non-null object
rating_numerator      2095 non-null int64
rating_denominator    2095 non-null int64
name                  2095 non-null object
doggo                 2095 non-null object
floofer               2095 non-null object
pupper                2095 non-null object
puppo                 2095 non-null object
dtypes: int64(3), object(9)
memory usage: 212.8+ KB
None
```

Fixing Quality Issue 2 Define

Deleting redundant information from the column expanded_urls. Some rows contain the value twice, separated by a comma.

Code

```
In [502]: df_twitter_archive_cleaned.expanded_urls.value_counts()
```

```
Out[502]: https://vine.co/v/ea0OwvPTx9l
          https://twitter.com/dog_rates/status/748324050481647620/photo/1,https://twitter.com/do
          https://twitter.com/dog_rates/status/766008592277377025/photo/1
          https://twitter.com/dog_rates/status/666044226329800704/photo/1
          https://twitter.com/dog_rates/status/684940049151070208/photo/1,https://twitter.com/do
          https://twitter.com/dog_rates/status/693647888581312512/photo/1
          https://twitter.com/dog_rates/status/754120377874386944/photo/1
          https://twitter.com/dog_rates/status/670465786746662913/photo/1
          https://twitter.com/dog_rates/status/739606147276148736/photo/1,https://twitter.com/do
          https://twitter.com/dog_rates/status/699088579889332224/photo/1
          https://twitter.com/dog_rates/status/756303284449767430/photo/1
          https://twitter.com/dog_rates/status/742423170473463808/photo/1
          https://twitter.com/dog_rates/status/666099513787052032/photo/1
          https://twitter.com/dog_rates/status/673656262056419329/photo/1
          https://twitter.com/dog_rates/status/693262851218264065/photo/1
          https://twitter.com/dog_rates/status/671497587707535361/photo/1
          https://twitter.com/dog_rates/status/774757898236878852/photo/1
```

```
https://twitter.com/dog_rates/status/771014301343748096/photo/1
https://twitter.com/dog_rates/status/715009755312439296/photo/1
https://twitter.com/dog_rates/status/680130881361686529/photo/1
https://twitter.com/dog_rates/status/698549713696649216/photo/1
https://twitter.com/dog_rates/status/707411934438625280/photo/1
https://twitter.com/dog_rates/status/667902449697558528/photo/1
https://twitter.com/dog_rates/status/676089483918516224/photo/1
https://twitter.com/dog_rates/status/677716515794329600/photo/1
https://twitter.com/dog_rates/status/813142292504645637/photo/1,https://twitter.com/do
https://twitter.com/dog_rates/status/672205392827572224/photo/1
https://twitter.com/dog_rates/status/691675652215414786/photo/1
https://twitter.com/dog_rates/status/692187005137076224/photo/1,https://twitter.com/do
https://twitter.com/dog_rates/status/686386521809772549/photo/1

https://twitter.com/dog_rates/status/667160273090932737/photo/1
https://twitter.com/dog_rates/status/703769065844768768/photo/1,https://twitter.com/do
https://twitter.com/dog_rates/status/674291837063053312/photo/1
https://twitter.com/dog_rates/status/749996283729883136/photo/1
https://twitter.com/dog_rates/status/775085132600442880/photo/1
https://twitter.com/dog_rates/status/709556954897764353/photo/1,https://twitter.com/do
https://twitter.com/dog_rates/status/749064354620928000/photo/1,https://twitter.com/do
https://twitter.com/dog_rates/status/837110210464448512/photo/1
https://twitter.com/dog_rates/status/852226086759018497/video/1
https://twitter.com/dog_rates/status/671866342182637568/photo/1
https://twitter.com/dog_rates/status/846874817362120707/photo/1,https://twitter.com/do
https://twitter.com/dog_rates/status/751132876104687617/photo/1,https://twitter.com/do
https://twitter.com/dog_rates/status/882627270321602560/photo/1
https://twitter.com/dog_rates/status/673352124999274496/photo/1
https://twitter.com/dog_rates/status/667911425562669056/photo/1
https://twitter.com/dog_rates/status/867421006826221569/photo/1
https://twitter.com/dog_rates/status/744334592493166593/photo/1
https://twitter.com/dog_rates/status/667453023279554560/photo/1
https://twitter.com/dog_rates/status/670792680469889025/photo/1
https://twitter.com/dog_rates/status/680970795137544192/photo/1
https://twitter.com/dog_rates/status/671744970634719232/photo/1
https://twitter.com/dog_rates/status/686007916130873345/photo/1
https://twitter.com/dog_rates/status/830583320585068544/photo/1,https://twitter.com/do
https://twitter.com/dog_rates/status/779377524342161408/video/1
https://twitter.com/dog_rates/status/793150605191548928/photo/1
https://twitter.com/dog_rates/status/743510151680958465/video/1
https://twitter.com/dog_rates/status/834167344700198914/photo/1
https://twitter.com/dog_rates/status/669006782128353280/photo/1
https://twitter.com/dog_rates/status/679877062409191424/photo/1
https://twitter.com/dog_rates/status/756526248105566208/photo/1
Name: expanded_urls, Length: 2091, dtype: int64
```

In [503]:
```
df_twitter_archive_cleaned['expanded_urls_cleaned'] = df_twitter_archive_cleaned.expan
df_twitter_archive_cleaned.drop('expanded_urls', axis = 1, inplace=True)
```

```
df_twitter_archive_cleaned.rename(columns={'expanded_urls_cleaned':'expanded_url'}, in
```

Test

```
In [504]: df_twitter_archive_cleaned.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2095 entries, 2 to 2355
Data columns (total 12 columns):
tweet_id              2095 non-null int64
timestamp             2095 non-null object
source                2095 non-null object
text                  2095 non-null object
rating_numerator      2095 non-null int64
rating_denominator    2095 non-null int64
name                  2095 non-null object
doggo                 2095 non-null object
floofer               2095 non-null object
pupper                2095 non-null object
puppo                 2095 non-null object
expanded_url          2092 non-null object
dtypes: int64(3), object(9)
memory usage: 212.8+ KB
```

Fixing Quality Issue 3 Define
Delete rows with duplicate expanded_urls, in case there are still some left.
Code

```
In [505]: df_twitter_archive_cleaned.drop_duplicates(subset="expanded_url", inplace=True)
```

Test

```
In [506]: sum(df_twitter_archive_cleaned.duplicated(subset="expanded_url"))

Out[506]: 0
```

Fixing Quality Issue 4 Define
Deleting rows with illogical values in the columns rating_numerator and rating_denominator.
Code

```
In [507]: df_twitter_archive_cleaned  = df_twitter_archive_cleaned[(df_twitter_archive_cleaned.r
                                                                    & (df_twitter_archive_cleaned

In [508]: # df_numerator_check serves for later checking
          df_numerator_check = df_twitter_archive_cleaned[(df_twitter_archive_cleaned.rating_num
                                                          (df_twitter_archive_cleaned.rating_num

In [509]: df_twitter_archive_cleaned  = df_twitter_archive_cleaned[(df_twitter_archive_cleaned.r
                                                                    & (df_twitter_archive_cleaned
```

```
In [510]: # df_nominator_check serves for later checking
          df_denominator_check = df_twitter_archive_cleaned[(df_twitter_archive_cleaned.rating_d
                                                            | (df_twitter_archive_cleaned.rating
```

Test

```
In [511]: df_numerator_check.shape # Must be zero rows now.

Out[511]: (0, 12)

In [512]: df_numerator_check.shape # Must be zero rows now.

Out[512]: (0, 12)

In [513]: df_twitter_archive_cleaned.shape # Check - must be less rows in total now.

Out[513]: (2067, 12)
```

Fixing Quality Issue 5 Define
Merging columns rating_numerator and rating_denominator.
Code

```
In [514]: df_twitter_archive_cleaned['rating'] = df_twitter_archive_cleaned.rating_numerator / d

          #Dropping factor columns
          columns_to_be_dropped = []
          columns_to_be_dropped = ["rating_numerator", "rating_denominator"]
          df_twitter_archive_cleaned.drop(columns_to_be_dropped, axis = 1, inplace=True)
          columns_to_be_dropped = []
```

Test

```
In [515]: df_twitter_archive_cleaned.rating.value_counts() # Check if calculation is correct

Out[515]: 1.200000    486
          1.000000    434
          1.100000    413
          1.300000    284
          0.900000    152
          0.800000     98
          0.700000     51
          1.400000     38
          0.500000     34
          0.600000     32
          0.300000     19
          0.400000     15
          0.200000      9
          0.818182      1
          0.636364      1
          Name: rating, dtype: int64
```

```
In [516]: df_twitter_archive_cleaned.info() # Check if column has been inserted properly.
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2067 entries, 2 to 2355
Data columns (total 11 columns):
tweet_id        2067 non-null int64
timestamp       2067 non-null object
source          2067 non-null object
text            2067 non-null object
name            2067 non-null object
doggo           2067 non-null object
floofer         2067 non-null object
pupper          2067 non-null object
puppo           2067 non-null object
expanded_url    2066 non-null object
rating          2067 non-null float64
dtypes: float64(1), int64(1), object(9)
memory usage: 193.8+ KB
```

Fixing Quality Issue 6 Define
Dropping column source and name.
Code

```
In [517]: columns_to_be_dropped = ["source", "name"]
          df_twitter_archive_cleaned.drop(columns_to_be_dropped, axis = 1, inplace=True)
          columns_to_be_dropped = []
```

Test

```
In [518]: print(df_twitter_archive_cleaned.info()) #Check
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2067 entries, 2 to 2355
Data columns (total 9 columns):
tweet_id        2067 non-null int64
timestamp       2067 non-null object
text            2067 non-null object
doggo           2067 non-null object
floofer         2067 non-null object
pupper          2067 non-null object
puppo           2067 non-null object
expanded_url    2066 non-null object
rating          2067 non-null float64
dtypes: float64(1), int64(1), object(7)
memory usage: 161.5+ KB
None
```

Tidiness Issues

Fixing Tidiness Issue 1 (optional) No need to separate Tweets of replies and retweets from the archive table. They have been deleted, as they will not be part of the analysis by definition.

Fixing Tidiness Issue 2 (optional) Define
Dropping columns with special classifications. They are not needed for this particular (my) analysis.
Code

```
In [519]: columns_to_be_dropped = []
          columns_to_be_dropped = ["doggo", "floofer", "pupper", "puppo"]
          df_twitter_archive_cleaned.drop(columns_to_be_dropped, axis = 1, inplace=True)
          columns_to_be_dropped = []
```

Test

```
In [520]: df_twitter_archive_cleaned.info() # Check
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2067 entries, 2 to 2355
Data columns (total 5 columns):
tweet_id        2067 non-null int64
timestamp       2067 non-null object
text            2067 non-null object
expanded_url    2066 non-null object
rating          2067 non-null float64
dtypes: float64(1), int64(1), object(3)
memory usage: 96.9+ KB
```

1.4.2 Table 2: Image Predictions

General preparations

```
In [521]: # Creating a copy of the dataset.
          df_image_predictions_cleaned = df_image_predictions.copy()
```

```
In [522]: df_image_predictions_cleaned.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
```

Fixing Quality Issues

Fixing Quality Issue 1 Define
Duplicated img_urls: fixing not neccessary. Refer to prior explanation.

Fixing Quality Issue 2 Define
The false values inside the columns p1 to p3_dog imply not all images are showing dogs. Manual sampling revealed chances are high an image contains a dog, when at least one of the values per is row is true. I, therefore, drop rows, which where the values in the columns p1_dog, p2_dog, p3_dog are all false.

```
In [523]: df_image_predictions_cleaned = df_image_predictions_cleaned[(df_image_predictions_clea
                                          (df_image_predictions_cleaned['p2_dog'] == True) |
                                          (df_image_predictions_cleaned['p3_dog'] == True)]
```

Test

```
In [524]: df_image_predictions_cleaned.shape # Number of rows should be much higher now.

Out[524]: (1751, 12)
```

Fixing Quality Issue 3 and 4 Define
The columns p1, p2, p3 have mixed capitalization and single words are separated by an underscore. Each first letter of a word will be capitalized and underscores will be replaced by spaces.

```
In [525]: columns_to_be_corrected =  ["p1", "p2", "p3"]
          for column in columns_to_be_corrected:
              df_image_predictions_cleaned[column].replace('_', ' ', inplace=True, regex=True)
              df_image_predictions_cleaned[column] = df_image_predictions_cleaned[column].str.ti
```

Test

```
In [526]: df_image_predictions_cleaned.sample(10) # Visual check

Out[526]:                 tweet_id                                       jpg_url  \
          1648  808838249661788160  https://pbs.twimg.com/media/CzmSFlKUAAAQOjP.jpg
          565   678278586130948096  https://pbs.twimg.com/media/CWm6xySUEAAqfFU.jpg
          355   672594978741354496  https://pbs.twimg.com/media/CVWJkJXWsAInlZl.jpg
```

```
1627  804413760345620481  https://pbs.twimg.com/media/CuRDF-XWcAIZSer.jpg
644   681579835668455424  https://pbs.twimg.com/media/CXV1Ot_W8AEpkQO.jpg
451   674743008475090944  https://pbs.twimg.com/media/CVOrL7RWEAAbhqm.jpg
1861  842535590457499648  https://pbs.twimg.com/media/C7FJpgVW4AIDzi6.jpg
279   671109016219725825  https://pbs.twimg.com/media/CVBCFkyU4AE2Wcr.jpg
1389  766423258543644672  https://pbs.twimg.com/media/CqLh4yJWcAAHomv.jpg
907   700462010979500032  https://pbs.twimg.com/media/CbiKe7-WOAIVNNr.jpg

      img_num                    p1   p1_conf  p1_dog                    p2  \
1648        1            Rottweiler  0.369530    True    Miniature Pinscher
565         1           Maltese Dog  0.897841    True                 Lhasa
355         1        Great Pyrenees  0.755945    True  Old English Sheepdog
1627        1                  Chow  0.090341    True            Binoculars
644         1            Rottweiler  0.760671    True    Labrador Retriever
451         1  Bernese Mountain Dog  0.583054    True     Shetland Sheepdog
1861        1              Pembroke  0.685084    True              Cardigan
279         1               Basenji  0.855959    True                Beagle
1389        2              Keeshond  0.995823    True            Pomeranian
907         1               Hamster  0.678651   False            Pomeranian

       p2_conf  p2_dog                         p3   p3_conf  p3_dog
1648  0.194867    True                     Kelpie  0.160104    True
565   0.035717    True             Tibetan Terrier  0.017107    True
355   0.082337    True                Afghan Hound  0.027037    True
1627  0.083499   False                Irish Setter  0.077456    True
644   0.096585    True   Staffordshire Bullterrier  0.040333    True
451   0.065990    True  Greater Swiss Mountain Dog  0.065236    True
1861  0.314608    True                     Basenji  0.000160    True
279   0.036723    True                 Toy Terrier  0.029258    True
1389  0.003897    True          Norwegian Elkhound  0.000253    True
907   0.110268    True                      Angora  0.104139   False
```

In [527]:
```
# The last step reduced ambiguity in the writings, there may be duplicates now.
# Check for duplicates after correcting capitalization and separation.
sum(df_image_predictions_cleaned.duplicated())
```

Out[527]: 0

Fixing Quality Issue 5 Define

The column img_num does not have a clear purpose and does not add value. It will therefore be droppped.

In [528]:
```
df_image_predictions_cleaned.drop("img_num", axis = 1, inplace=True)
```

Test

In [529]:
```
df_image_predictions_cleaned.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1751 entries, 0 to 2073
Data columns (total 11 columns):
tweet_id    1751 non-null int64
jpg_url     1751 non-null object
p1          1751 non-null object
p1_conf     1751 non-null float64
p1_dog      1751 non-null bool
p2          1751 non-null object
p2_conf     1751 non-null float64
p2_dog      1751 non-null bool
p3          1751 non-null object
p3_conf     1751 non-null float64
p3_dog      1751 non-null bool
dtypes: bool(3), float64(3), int64(1), object(4)
memory usage: 128.2+ KB
```

Fixing Tidiness Issues

Fixing Tidiness Issues 1 Define
Joining the prediction value columns p1 to p3 to a combined set of columns.
Code

```
In [530]: #Splitting the prediction tries into separate dataframes and dropping the other tries.
          df_q2_1_temp = df_image_predictions_cleaned.copy()
          df_q2_1_temp.rename(columns={'p1':'p','p1_conf':'p_conf', 'p1_dog':'p_dog'}, inplace=T
          columns_to_be_dropped = []
          columns_to_be_dropped = ["p2", "p2_conf", "p2_dog", "p3", "p3_conf", "p3_dog"]
          df_q2_1_temp.drop(columns_to_be_dropped, axis = 1, inplace=True)
          columns_to_be_dropped = []

          df_q2_2_temp = df_image_predictions_cleaned.copy()
          df_q2_2_temp.rename(columns={'p2':'p','p2_conf':'p_conf', 'p2_dog':'p_dog'}, inplace=T
          columns_to_be_dropped = []
          columns_to_be_dropped = ["p1", "p1_conf", "p1_dog", "p3", "p3_conf", "p3_dog"]
          df_q2_2_temp.drop(columns_to_be_dropped, axis = 1, inplace=True)
          columns_to_be_dropped = []

          df_q2_3_temp = df_image_predictions_cleaned.copy()
          df_q2_3_temp.rename(columns={'p3':'p','p3_conf':'p_conf', 'p3_dog':'p_dog'}, inplace=T
          columns_to_be_dropped = []
          columns_to_be_dropped = ["p1", "p1_conf", "p1_dog", "p2", "p2_conf", "p2_dog"]
          df_q2_3_temp.drop(columns_to_be_dropped, axis = 1, inplace=True)
          columns_to_be_dropped = []

          print(str(df_q2_1_temp.shape)) # Quick check on the validity. --> Looks good.
```

```
print(str(df_q2_2_temp.shape))
print(str(df_q2_3_temp.shape))
```

```
(1751, 5)
(1751, 5)
(1751, 5)
```

In [531]:
```
# Now adding to each temporary data frame a column which states which number of try it
df_q2_1_temp['p_try_number'] = 1
df_q2_2_temp['p_try_number'] = 2
df_q2_3_temp['p_try_number'] = 3
```

In [532]:
```
# Quick check
df_q2_1_temp.sample(3)
```

Out[532]:
```
               tweet_id                                          jpg_url  \
789  690597161306841088  https://pbs.twimg.com/media/CZV-c9NVIAEWtiU.jpg
564  678255464182861824  https://pbs.twimg.com/media/CWmlvxJU4AEAqaN.jpg
523  676588346097852417  https://pbs.twimg.com/media/CWO5gmCUYAAX4WA.jpg

                p    p_conf  p_dog  p_try_number
789        Lhasa  0.097500   True             1
564    Chihuahua  0.613819   True             1
523  Boston Bull  0.976577   True             1
```

In [533]:
```
df_q2_2_temp.sample(3) # Quick check
```

Out[533]:
```
                tweet_id                                          jpg_url  \
1618  802572683846291456  https://pbs.twimg.com/media/CyNPmJgXcAECPuB.jpg
1607  800513324630806528  https://pbs.twimg.com/media/Cxv-nkJUoAAhzMt.jpg
1422  772114945936949249  https://pbs.twimg.com/media/Crcacf9WgAEcrMh.jpg

                      p    p_conf  p_dog  p_try_number
1618  Labrador Retriever  0.173252   True             2
1607            Cardigan  0.167373   True             2
1422         Toy Terrier  0.052980   True             2
```

In [534]:
```
df_q2_3_temp.sample(3) # Quick check
```

Out[534]:
```
                tweet_id                                          jpg_url  \
909   700518061187723268  https://pbs.twimg.com/media/Cbi9dI_UYAAgkyC.jpg
947   704761120771465216  https://pbs.twimg.com/media/CcfQgHVWoAAxauy.jpg
1345  759159934323924993  https://pbs.twimg.com/media/CU1zsMSUAAASOqW.jpg

                                p    p_conf  p_dog  p_try_number
909                     Chihuahua  0.121839   True             3
947                       Basenji  0.072097   True             3
1345  Soft-Coated Wheaten Terrier  0.223263   True             3
```

```
In [535]: # Append the three outcomes/options to one single data frame
          df_q2 = df_q2_1_temp.append(df_q2_2_temp)
          df_q2 = df_q2.append(df_q2_3_temp)
          print(str(df_q2.shape)) # Quick check on the validity. --> Looks good.
          print(str(sum(df_q2.duplicated()))) # No duplicates
```

```
(5253, 6)
0
```

```
In [536]: df_q2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 5253 entries, 0 to 2073
Data columns (total 6 columns):
tweet_id        5253 non-null int64
jpg_url         5253 non-null object
p               5253 non-null object
p_conf          5253 non-null float64
p_dog           5253 non-null bool
p_try_number    5253 non-null int64
dtypes: bool(1), float64(1), int64(2), object(2)
memory usage: 251.4+ KB
```

```
In [537]: # Handing the temporary, unpivoted, cleaned image prediction table over to the origina
          df_image_predictions_cleaned = []
          df_image_predictions_cleaned = df_q2.copy()
```

Test

```
In [538]: df_image_predictions_cleaned.info() # Check if the unpivot action has been handed over
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 5253 entries, 0 to 2073
Data columns (total 6 columns):
tweet_id        5253 non-null int64
jpg_url         5253 non-null object
p               5253 non-null object
p_conf          5253 non-null float64
p_dog           5253 non-null bool
p_try_number    5253 non-null int64
dtypes: bool(1), float64(1), int64(2), object(2)
memory usage: 251.4+ KB
```

Fixing Tidiness Issues 2 Define
Creating one combined table, i.e. merging the three tables by twitter_id.
Code

```
In [539]: df_twitter_combined = 0
          df_twitter_combined = df_twitter_archive_cleaned.copy()
```

```
In [540]: # Merging performance data in.
          df_twitter_combined = df_twitter_combined.merge(df_tweet_performance, on='tweet_id', h
```

```
In [541]: print("Shape df_twitter_archive_cleaned: " + str(df_twitter_archive_cleaned.shape))
          print("Shape df_tweet_performance: " + str(df_tweet_performance.shape))
          print("Shape df_twitter_combined: " + str(df_twitter_combined.shape) + " after joining
```

```
Shape df_twitter_archive_cleaned: (2067, 5)
Shape df_tweet_performance: (2340, 3)
Shape df_twitter_combined: (2065, 7) after joining df_twitter_archive_cleaned and df_tweet_perfo
```

```
In [542]: # Merging image prediction data in.
          df_twitter_combined = df_twitter_combined.merge(df_image_predictions_cleaned, on='twee
```

Test

```
In [543]: print("Shape df_image_predictions_cleaned: " + str(df_image_predictions_cleaned.shape)
          print("Shape df_twitter_combined " + str(df_twitter_combined.shape) + " after joining
```

```
Shape df_image_predictions_cleaned: (5253, 6)
Shape df_twitter_combined (4935, 12) after joining df_twitter_archive_cleaned, df_tweet_performa
```

The amounts of rows look plausible. The combined dataset should have less rows than before after each of the two merges.

```
In [544]: # Checking the overall shape of the triple-merge.
          df_twitter_combined.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 4935 entries, 0 to 4934
Data columns (total 12 columns):
tweet_id          4935 non-null int64
timestamp         4935 non-null object
text              4935 non-null object
expanded_url      4935 non-null object
rating            4935 non-null float64
retweet_count     4935 non-null int64
favorite_count    4935 non-null int64
jpg_url           4935 non-null object
p                 4935 non-null object
p_conf            4935 non-null float64
p_dog             4935 non-null bool
p_try_number      4935 non-null int64
dtypes: bool(1), float64(2), int64(4), object(5)
memory usage: 467.5+ KB
```

The amound of rows and the structure looks good.

1.5 Storing Cleaned Data

In [545]:
```
# According to the instructions the cleaned data frame of the Twitter archive
# shall be stored to a file called "twitter_archive_master.csv".
df_twitter_archive_cleaned.to_csv("deliverables/twitter_archive_master.csv", header=Tr
```

In [546]:
```
# The data frame image_prediction_cleaned shall be stored to a separate CSV-file.
df_image_predictions_cleaned.to_csv("deliverables/twitter_image_predictions_master.csv
```

In [547]:
```
# The data frame tweet_performance shall be stored to a separate CSV-file.
df_tweet_performance.drop_duplicates(inplace=True)
df_tweet_performance.to_csv("deliverables/df_tweet_performance_master.csv", header=Tru
```

In [548]:
```
# Storing the combined dataset as well to a separate file
df_twitter_combined.to_csv("deliverables/df_twitter_combined_master.csv", header=True,
```

1.5.1 Storing Cleaned Data Into A SQLite Database

In [549]:
```
from sqlalchemy import create_engine
```

In [550]:
```
# Create SQLAlchemy engine
engine = create_engine('sqlite:///deliverables/twitter_analysis.db')
```

In [551]:
```
# Storing pandas data frames into the db-tables. In order to keep all processed data i
# versions, the following three subsets are going to be stored next to the final "comb
df_twitter_archive_cleaned.to_sql('twitter_archive_master', engine, index=False)
df_image_predictions_cleaned.to_sql('twitter_image_predictions_master', engine, index=
df_tweet_performance.to_sql('df_tweet_performance_master', engine, index=False)

# Storing the final master dataset into the database as well. This is the combined ver
# separate sources (joined).
df_twitter_combined.to_sql('df_twitter_combined_master', engine, index=False)
```

1.6 Analysis

Before starting the analysis, I drop all columns except those needed.

In [552]:
```
columns_to_be_dropped = []
columns_to_be_dropped = ["tweet_id", "timestamp", "text", "expanded_url", "jpg_url"]
df_twitter_combined.drop(columns_to_be_dropped, axis = 1, inplace=True)
columns_to_be_dropped = []
```

The dataset respectively the data frame df_twitter_combined is now ready for analysis.

Question 1: What were the 10 highest rated dog breeds on average, according to the prediction with the highest (p1_conf) confidence level?

In [553]:
```
df_q1 = df_twitter_combined[(df_twitter_combined['p_dog'] == True) &
                            (df_twitter_combined['p_try_number'] == 1)] # p1_conf --> p
highest_rated_breeds = (df_q1.groupby(['p']).mean()['rating'])
highest_rated_breeds.sort_values(ascending = False).head(10)
```

```
Out[553]: p
          Saluki                1.250000
          Briard                1.233333
          Tibetan Mastiff       1.225000
          Border Terrier        1.214286
          Silky Terrier         1.200000
          Standard Schnauzer    1.200000
          Eskimo Dog            1.177778
          Gordon Setter         1.175000
          Irish Setter          1.175000
          Samoyed               1.174359
          Name: rating, dtype: float64
```

Insight: The highest rated dog breed is the Saluki. Surprisingly, its average rating is 1.25 or 12.5/10. I personally would have expected it to be higher. Furthermore, the distribution of the average ratings per dog breed is close to each other. There is no breed which is rated far better than the rest.

Question 2: Which are the top 10 predictated dog breeds in terms of the confidence-level on average?

```
In [554]: df_question2 = df_twitter_combined[df_twitter_combined['p_dog'] == True]
```

```
In [555]: highest_confidence_breed_prediction = (df_question2.groupby(['p']).mean()['p_conf'])
          highest_confidence_breed_prediction.sort_values(ascending = False).head(10)
```

```
Out[555]: p
          Bernese Mountain Dog    0.651259
          Komondor                0.522381
          Samoyed                 0.520342
          Pembroke                0.495139
          Pug                     0.481903
          Blenheim Spaniel        0.475460
          Golden Retriever        0.446421
          Dalmatian               0.400302
          German Shepherd         0.369846
          Vizsla                  0.361907
          Name: p_conf, dtype: float64
```

Insight: The highest average confidence level in predicting the dog breed experienced the Bernese Mountain Dog. The second is the Komondor. Both breeds have a unique appearance, which probably made the prediction easier. The highest confidence level is 65% on average, which should be kept in mind. In my opinion, the level is too low, to really rely on these results. For deeper analysis, you should consider running the images through a model which is either dedicated to dogs only or which is trained better.

Question 3: Which dog breeds have the highest average retweet and favorite-count, according to the prediction with the highest (p1_conf) confidence level?

```
In [556]: df_q3_1 = df_twitter_combined[(df_twitter_combined['p_dog'] == True) &
                                        (df_twitter_combined['p_try_number'] == 1)]
```

```
In [557]: # Retweet count
          highest_retweet_count = (df_q3_1.groupby(['p']).mean()['retweet_count'])
          highest_retweet_count.sort_values(ascending = False).head(10)
```

```
Out[557]: p
          Standard Poodle          6284.285714
          English Springer         5658.333333
          Afghan Hound             5634.666667
          Eskimo Dog               5182.500000
          Giant Schnauzer          4845.000000
          Saluki                   4841.250000
          Great Pyrenees           4703.307692
          French Bulldog           4606.760000
          Lakeland Terrier         4576.866667
          Flat-Coated Retriever    4278.375000
          Name: retweet_count, dtype: float64
```

```
In [558]: # Favorite count
          highest_favorite_count = (df_q3_1.groupby(['p']).mean()['favorite_count'])
          highest_favorite_count.sort_values(ascending = False).head(10)
```

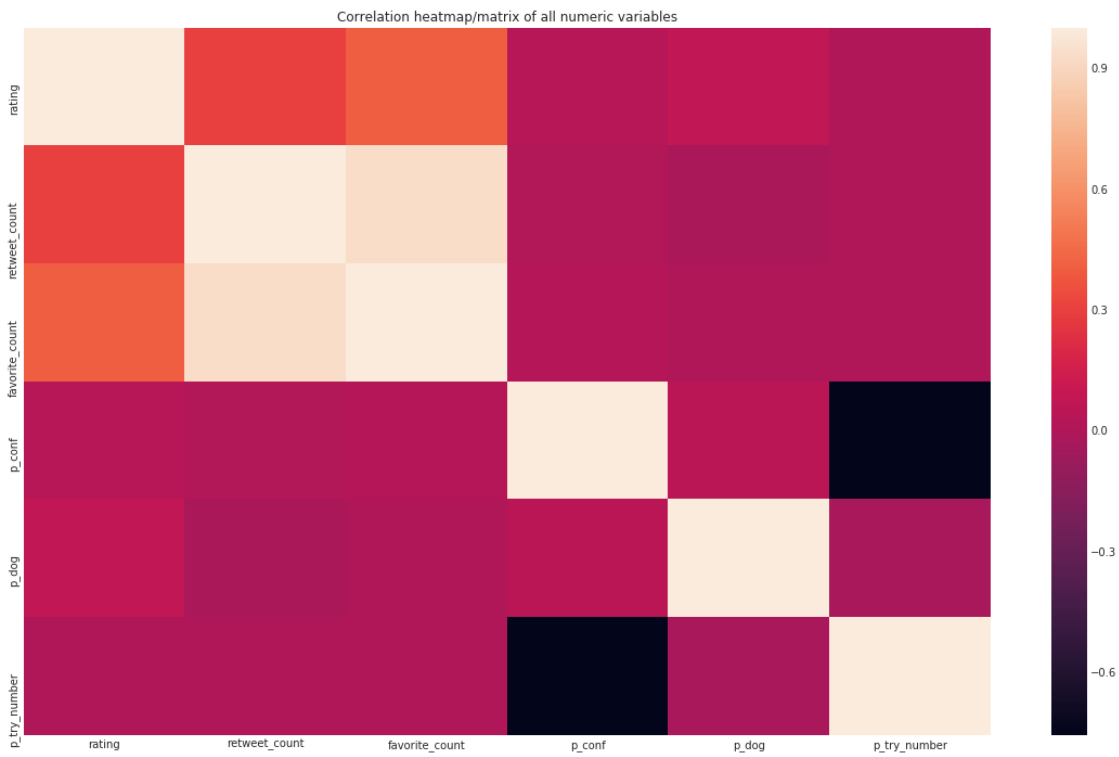
```
Out[558]: p
          Saluki                     23210.250000
          French Bulldog             18312.160000
          Giant Schnauzer            16554.000000
          Afghan Hound               16548.666667
          Black-And-Tan Coonhound    16360.000000
          Flat-Coated Retriever      16155.000000
          Irish Water Spaniel        15702.666667
          Standard Poodle            15182.142857
          English Springer           15065.333333
          Cardigan                   14673.764706
          Name: favorite_count, dtype: float64
```

Insight: On average the breed Standard Poodle appears to be retweeted most, whereas the Saluki appears to have earned the most favorites by far. However, you can see, that many breeds are in both top ten listings present. This might be a hint of a potential correlation between these variables.

Question 4: Does the rating correlate with the number of retweets or favorite-counts?

```
In [559]: # It makes sense, to create at heatmap showing the correlation factors of all variable
          # answer the question and to potentially discover more interesting relationships betwe

          #Plotting the results of calculations above.
          fig = plt.figure(figsize=(20,12))
```

```
sns.set_style('whitegrid')

corr = df_twitter_combined.corr()
sns.heatmap(corr,
            xticklabels=corr.columns.values,
            yticklabels=corr.columns.values);
plt.title('Correlation heatmap/matrix of all numeric variables');
```



In [560]:
```
df_twitter_combined.corr()
```

Out[560]:

	rating	retweet_count	favorite_count	p_conf	p_dog	\
rating	1.000000	0.303094	0.405390	0.032208	0.073449	
retweet_count	0.303094	1.000000	0.932059	0.012044	-0.013928	
favorite_count	0.405390	0.932059	1.000000	0.018496	0.002096	
p_conf	0.032208	0.012044	0.018496	1.000000	0.046688	
p_dog	0.073449	-0.013928	0.002096	0.046688	1.000000	
p_try_number	0.000000	0.000000	0.000000	-0.758019	-0.024727	

	p_try_number
rating	0.000000
retweet_count	0.000000
favorite_count	0.000000
p_conf	-0.758019

```
p_dog              -0.024727
p_try_number        1.000000
```

Insight: No, the variable rating does neither correlate strongly with the variable retweet_count (0.30), nor with the variable favorite_count (0.41). However, the variable retweet and favorite count correlate strongly with each other (0.93). It appears the viewers of the WeRateDogs Twitter account express their liking using both ways.

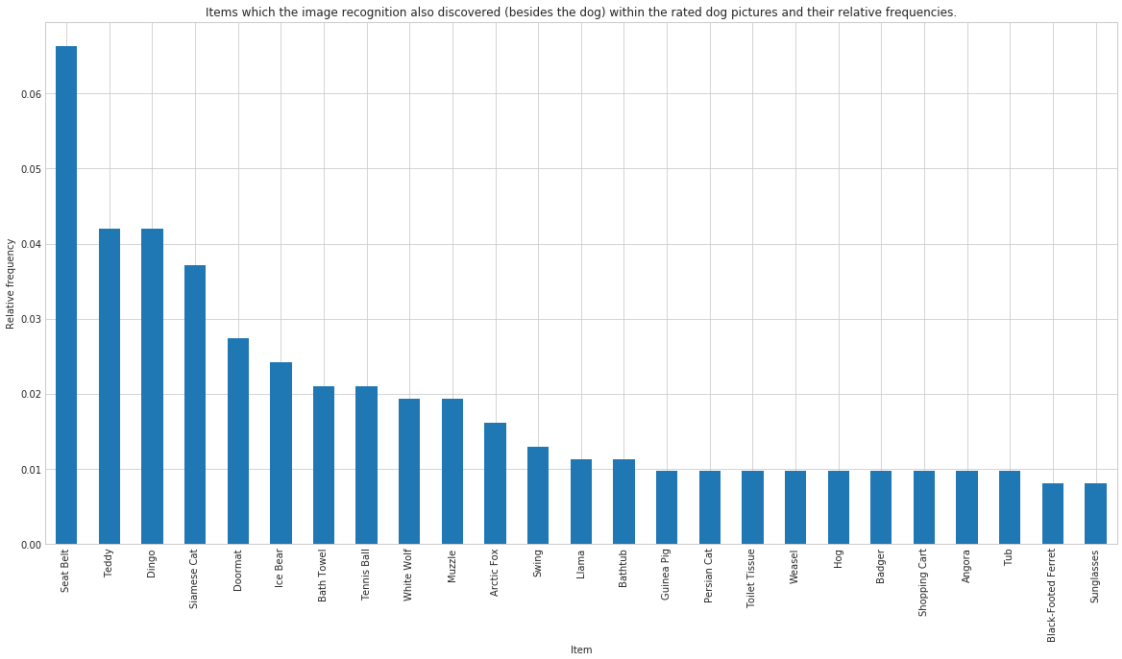
Question 5: Which items were also visible in the pictures and what are their frequencies?

In [561]:
```
df_q5 = df_twitter_combined[df_twitter_combined['p_dog'] == False]
df_q5 = (df_q5[['p']].copy())
```

In [562]:
```
df_q5.p.value_counts(sort=True, normalize=True).head(10) # Showing the relative freque
```

Out[562]:
```
Seat Belt      0.066236
Teddy          0.042003
Dingo          0.042003
Siamese Cat    0.037157
Doormat        0.027464
Ice Bear       0.024233
Bath Towel     0.021002
Tennis Ball    0.021002
White Wolf     0.019386
Muzzle         0.019386
Name: p, dtype: float64
```

In [563]:
```
#Plotting results of the calculations above.
fig = plt.figure(figsize=(20,10))
sns.set_style('whitegrid')

df_q5.p.value_counts(sort=True, normalize=True).head(25).plot.bar()
plt.title('Items which the image recognition also discovered (besides the dog) within
plt.xlabel('Item')
plt.ylabel('Relative frequency');
```



Insight: The most frequent item within the dog pictures appears to be a "seat belt", followed by teddies. The second item is classified as a Dingo, which actually might come down to a misinterpreted dog breed.

1.7 References

http://docs.tweepy.org/en/v3.5.0/api.html
https://stackoverflow.com/questions/12309269/how-do-i-write-json-data-to-a-file
https://stackabuse.com/reading-and-writing-json-to-a-file-in-python/
https://stackoverflow.com/questions/27900451/convert-tweepy-status-object-into-json
https://developer.twitter.com/en/docs/basics/rate-limiting
https://stackoverflow.com/questions/7370801/measure-time-elapsed-in-python
https://stackoverflow.com/questions/40705480/python-pandas-remove-everything-after-a-delimit
https://stackoverflow.com/questions/42462530/how-to-replace-the-white-space-in-a-string-in-a
https://stackoverflow.com/questions/39141856/capitalize-first-letter-of-each-word-in-the-col
https://stackoverflow.com/questions/34682828/extracting-specific-selected-columns-to-new-dat
https://stackoverflow.com/questions/29432629/correlation-matrix-using-pandas