wrangle_act

April 25, 2021

In [208]:
```
# Downloading and importing all the necessary libraries to complete the project.
import tweepy
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
import requests
import json
import os
import re
import warnings
warnings.simplefilter('ignore')
```

In [209]:
```
from PIL import Image
from wordcloud import WordCloud, STOPWORDS, ImageColorGenerator
```

In [210]:
```
pd.set_option('display.max_colwidth', -1)
```

1 Gather

First table archive downloaded from the internet manually and programmatically opened into a pandas DataFrame.

In [211]:
```
archive = pd.read_csv('twitter-archive-enhanced.csv')
```

Second table tweet_count downloaded programmatically from twitter's API using tweepy, then saved to a JSON file, stored in a dictionary, then loaded into a pandas DataFrame.

In [212]:
```
from tweepy import OAuthHandler
from timeit import default_timer as timer
```

In [213]:
```
consumer_key = 'HIDDEN'
consumer_secret = 'HIDDEN'
access_token = 'HIDDEN'
access_secret = 'HIDDEN'
```

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

api = tweepy.API(auth, wait_on_rate_limit=True)

tweet_ids = archive.tweet_id.values
len(tweet_ids)
```

```
Out[213]: 2356
```

```
In [214]: # set a function for tweet extraction
          # file already created so no need to execute to continue the notebook
          def tweet_extraction():
              count = 0
              fails_dict = {}
              start = timer()
              with open('tweet_json.txt', 'w') as outfile:
                  for tweet_id in tweet_ids:
                      count += 1
                      print(str(count) + ": " + str(tweet_id))
                      try:
                          tweet = api.get_status(tweet_id, tweet_mode='extended')
                          print("Success")
                          json.dump(tweet._json, outfile)
                          outfile.write('\n')
                      except tweepy.TweepError as e:
                          print("Fail")
                          fails_dict[tweet_id] = e
                          pass
              end = timer()
              print(end - start)
              print(fails_dict)
```

```
In [229]: df_list = []
          with open('tweet_json.txt') as file:
              for line in file:
                  data = json.loads(line)
                  keys = data.keys()
                  user = data.get('user')
                  id_str = data.get('id_str')
                  retweet_count = data.get('retweet_count')
                  favorite_count = data.get('favorite_count')
                  df_list.append({'id_str': id_str,
                                  'retweet_count': retweet_count,
                                  'favorite_count': favorite_count})
```

```
[]
```

```
In [220]: tweet_count = pd.DataFrame(df_list, columns = ['id_str', 'retweet_count', 'favorite_co

Out[220]: []

In [219]: tweet_count.head()

Out[219]: Empty DataFrame
          Columns: []
          Index: []
```

Third table, programmatically downloaded from the Udacity servers and stored in a folder image_pred, then written to local computer and loaded into a pandas DataFrame.

```
In [218]: # Downloading the image predictions from the internet
          folder_name = 'image_pred'
          if not os.path.exists(folder_name):
              os.makedirs(folder_name)

          url = 'https://d17h27t6h515a5.cloudfront.net/topher/2017/August/599fd2ad_image-predict
          response = requests.get(url)

In [ ]: with open(os.path.join(folder_name, url.split('/')[-1]), mode='wb') as file:
            file.write(response.content)

In [ ]: image_pred = pd.read_csv('image_predictions.tsv', sep='\t')
```

2 Assessing Data

2.0.1 Visual

```
In [ ]: archive.head()

In [155]: archive.text.sample(20)

Out[155]: 1001    Pls don't send more sherks. I don't care how seemingly floofy they are. It doe
          2020    This is Tuco. That's the toast that killed his father. 9/10 https://t.co/ujnWy
          1407    Meet Reagan. He's a Persnicketus Derpson. Great with kids. Permanently caught
          1124    This is Ziva. She doesn't know how her collar works. 11/10 would totally fix f
          383     This is Charlie. He wins every game of chess he plays. Won't let opponent pet
          578     Say hello to Mauve and Murphy. They're rather h*ckin filthy. Preferred nap ove
          1667    Meet Joey and Izzy. Joey only has one ear that works and Izzy wants 2015 to be
          2087    This is Trigger. He was minding his own business on stair when he overheard sc
          2325    This is Walter. He is an Alaskan Terrapin. Loves outdated bandanas. One ear st
          539     Hooman catch successful. Massive hit by dog. Fumble ensued. Possession to dog.
          773     RT @dog_rates: We only rate dogs. Pls stop sending in non-canines like this Mc
          783     This is Maximus. A little rain won't stop him. He will persevere. 12/10 innova
          2016    This is Bradley. That is his sandwich. He carries it everywhere. 10/10 https:/
          1139    This is Rueben. He has reached ultimate pupper zen state. 11/10 tranquil af ht
```

```
1069    When the photographer forgets to tell you where to look... 10/10 https://t.co/
1528    This is Oddie. He's trying to communicate. 12/10 very solid effort (vid by @ka
1405    "I'm bathing the children what do you want?"  ...both 10/10 https://t.co/Rizm1
1752    Meet Lola. She's a Metamorphic Chartreuse. Plays with her food. Insubordinate
350     This is Dutch. He dressed up as his favorite emoji for Valentine's Day. I've g
53      This is Rey. He's a Benebop Cumberfloof. 12/10 dangerously pettable https://t.
Name: text, dtype: object
```

```
In [230]: image_pred.head()
```

Out[230]:

	tweet_id	jpg_url	img_num	p1	p1_conf	p1_dog	p2	p2_conf	p2_dog	p3	p3_conf	p3_dog
0	666020888022790149	https://pbs.twimg.com/media/CT4udn0WwAA0aMy.jpg	1	Welsh_springer_spaniel	0.465074	True	collie	0.156665	True	Shetland_sheepdog	0.061428	True
1	666029285002620928	https://pbs.twimg.com/media/CT42GRgUYAA5iDo.jpg	1	redbone	0.506826	True	miniature_pinscher	0.074192	True	Rhodesian_ridgeback	0.072010	True
2	666033412701032449	https://pbs.twimg.com/media/CT4521TWwAEvMyu.jpg	1	German_shepherd	0.596461	True	malinois	0.138584	True	bloodhound	0.116197	True
3	666044226329800704	https://pbs.twimg.com/media/CT5Dr8HUEAA-lEu.jpg	1	Rhodesian_ridgeback	0.408143	True	redbone	0.360687	True	miniature_pinscher	0.222752	True
4	666049248165822465	https://pbs.twimg.com/media/CT5IQmsXIAAKY4A.jpg	1	miniature_pinscher	0.560311	True	Rottweiler	0.243682	True	Doberman	0.154629	True

2.0.2 Quality

archive table

Missing values in columns: in_reply_to_status_id, in_reply_to_user_id, retweeted_status_id, retweeted_status_user_id, retweeted_status_timestamp, and expanded_urls.

Column name floofer should be spelled 'floof'

image-pred Table

The types of dogs in columns p1, p2, and p3 had some uppercase and lowercase letters.

Tidiness

The column text had multiple variables like a url link, rating, and some tweets represented two dogs.

The tweet_count and archive table should be merged as this is related data.

Programmatic

```
In [231]: 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
```

```
<li>tweet_id has dtype int64 and should be object</li>
<li><b>timestamp</b> should be a datetime64 dtype type as well</li>
```

```
In [232]: doggo = archive.doggo.value_counts()
          floofer = archive.floofer.value_counts()
          pupper = archive.pupper.value_counts()
          puppo = archive.puppo.value_counts()
          print(doggo);
          print(floofer);
          print(pupper);
          print(puppo)
```

```
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
```

Misssing information for the dog stages

```
In [233]: archive.name.value_counts().head(20)
```

```
Out[233]: None       745
          a           55
          Charlie     12
          Cooper      11
          Oliver      11
          Lucy        11
          Lola        10
          Tucker      10
          Penny       10
          Bo           9
          Winston      9
          the          8
          Sadie        8
          Buddy        7
          Daisy        7
          Toby         7
          an           7
          Bailey       7
          Bella        6
          Rusty        6
          Name: name, dtype: int64
```

Many missing names from the list under 'None', and random names like 'a' and 'an' might be parts of strings that got taken out of context.

```
In [234]: archive.source.value_counts()
```

```
Out[234]: <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
```

The source column looks messy and clutters the table

```
In [235]: tweet_count.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 0 entries
Data columns (total 3 columns):
id_str            0 non-null object
retweet_count     0 non-null object
favorite_count    0 non-null object
dtypes: object(3)
memory usage: 0.0+ bytes
```

```
In [236]: image_pred.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
```

The tweet_id column should be dtype object instead of int64.

Assessment Summary
Quality
Archive Table

```
<li>Missing values in columns: in_reply_to_status_id, in_reply_to_user_id, retweeted_status_id,
<li>Column name floofer should be spelled 'floof'</li>
<li>tweet_id has dtype int64 and should be object</li>
<li>timestamp should be a datetime64 dtype type as well</li>
<li>Missing information for dog stages.</li>
<li>Many missing names from the list under 'None', and random names like 'a' and 'an' might be p
<li>Remove from table retweets and replies keepng only original tweets.</li>
<li>Some tweets had "\&amp" combined with ";" which is the html code to display just the ampersa
```

tweet_count Table

```
<li>The column id_str should be changed to tweet_id so merging tables will be smoother.</li>
```

image_pred Table

```
<li>The types of dogs in columns p1, p2, and p3 had some uppercase and lowercase letters.</li>
<li>The tweet_id column should be dtype object instead of int64.</li>
```

Tidiness

```
<li>The tweet_count and archive table should be merged as this is added data to the other table.
<li>The source column in archive table looks messy and clutters the table.</li>
<li>All three tables will eventually be merged into one.</li>
```

3 Cleaning Data

Make copies of each table first before cleaning, as to help reduce catastrophe

```
In [237]: # trimmed the names in order to make less wordy when coding
          archive_clean = archive.copy()
          image_clean = image_pred.copy()
          tweet_clean = tweet_count.copy()
```

1 archive: remove from table retweets and replies keeping only original tweets

Define

Find the retweets and replies using the retweeted_status_id and in_reply_to_status_id columns and remove from the DataFrame

Code

```
In [238]: drop_retweet = archive_clean[pd.notnull(archive_clean['retweeted_status_id'])].index
          drop_reply = archive_clean[pd.notnull(archive_clean['in_reply_to_status_id'])].index

In [239]: archive_clean.drop(index=drop_retweet, inplace=True)
          archive_clean.drop(index=drop_reply, inplace=True)
```

Test

```
In [240]: archive_clean.info()

<class 'pandas.core.frame.DataFrame'>
Int64Index: 2097 entries, 0 to 2355
Data columns (total 17 columns):
tweet_id                      2097 non-null int64
in_reply_to_status_id         0 non-null float64
in_reply_to_user_id           0 non-null float64
timestamp                     2097 non-null object
source                        2097 non-null object
text                          2097 non-null object
retweeted_status_id           0 non-null float64
retweeted_status_user_id      0 non-null float64
retweeted_status_timestamp    0 non-null object
expanded_urls                 2094 non-null object
rating_numerator              2097 non-null int64
rating_denominator            2097 non-null int64
name                          2097 non-null object
doggo                         2097 non-null object
floofer                       2097 non-null object
pupper                        2097 non-null object
puppo                         2097 non-null object
dtypes: float64(4), int64(3), object(10)
memory usage: 294.9+ KB
```

4 2 archive: Missing values in columns and unnecessary columns

Define

Remove columns with missing values using dropna() method. Also, use the drop() method to drop source column from table as well

Code

```
In [241]: archive_clean.dropna(axis='columns',how='any', inplace=True)

In [242]: archive_clean.drop(columns='source', inplace=True)
```

Test

```
In [243]: archive_clean.head()
```

Out[243]:

	tweet_id	timestamp	\
0	892420643555336193	2017-08-01 16:23:56 +0000	
1	892177421306343426	2017-08-01 00:17:27 +0000	
2	891815181378084864	2017-07-31 00:18:03 +0000	
3	891689557279858688	2017-07-30 15:58:51 +0000	
4	891327558926688256	2017-07-29 16:00:24 +0000	

0	This is Phineas. He's a mystical boy. Only ever appears in the hole of a donut. 13/
1	This is Tilly. She's just checking pup on you. Hopes you're doing ok. If not, she's
2	This is Archie. He is a rare Norwegian Pouncing Corgo. Lives in the tall grass. You
3	This is Darla. She commenced a snooze mid meal. 13/10 happens to the best of us htt
4	This is Franklin. He would like you to stop calling him "cute." He is a very fierce

	rating_numerator	rating_denominator	name	doggo	floofer	pupper	puppo
0	13	10	Phineas	None	None	None	None
1	13	10	Tilly	None	None	None	None
2	12	10	Archie	None	None	None	None
3	13	10	Darla	None	None	None	None
4	12	10	Franklin	None	None	None	None

5 3 Fixing column names

tweet_clean: unifying column names

archive_clean: column names

Define

```
<li>In the tweet_clean table the column name id_str changed to tweet_id using the rename() funct
<li>In the archive_clean table, column name floofer should be "floof" to match the dog stage ass
```

Code

```
In [244]: tweet_clean.rename(index=str, columns={"id_str": "tweet_id"}, inplace=True)
          archive_clean.rename(columns={"floofer": "floof",
                                        "rating_numerator": "rate_num",
                                        "rating_denominator": "rate_denom"}, inplace=
```

Test

```
In [245]: tweet_clean.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 0 entries
Data columns (total 3 columns):
tweet_id          0 non-null object
retweet_count     0 non-null object
favorite_count    0 non-null object
dtypes: object(3)
memory usage: 0.0+ bytes
```

6 Fixing Datatypes

image_clean: tweet_id dtype "string"

archive_clean: timestamp dtype "datetime"

archive_clean: tweet_id dtype "string"

Define

```
<li>In the image_clean table, change the dtype of column tweet_id from int64 to object using the
<li>In the archive_clean table, change the dtype of column timestamp from object to datetime usi
<li>In the archive_clean table, change the dtype of column tweet_id from int64 to object using t
```

code

```
In [246]: image_clean['tweet_id'] = image_clean['tweet_id'].astype('str')
          archive_clean['timestamp'] = pd.to_datetime(archive_clean['timestamp'])
          archive_clean['tweet_id'] = archive_clean['tweet_id'].astype('str')
```

Test

```
In [247]: image_clean.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2075 entries, 0 to 2074
Data columns (total 12 columns):
tweet_id    2075 non-null object
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(1), object(5)
memory usage: 152.1+ KB
```

```
In [248]: archive_clean.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2097 entries, 0 to 2355
Data columns (total 10 columns):
tweet_id      2097 non-null object
timestamp     2097 non-null datetime64[ns]
text          2097 non-null object
rate_num      2097 non-null int64
rate_denom    2097 non-null int64
name          2097 non-null object
doggo         2097 non-null object
floof         2097 non-null object
pupper        2097 non-null object
puppo         2097 non-null object
dtypes: datetime64[ns](1), int64(2), object(7)
memory usage: 180.2+ KB
```

7 5 image_clean: dog breeds uniformity

Define

In the image_clean table, the dogbreeds in the p1, p2, and p3 are converting all the names to lowercase letters.

Code

```
In [249]: image_clean['p1'] = image_clean['p1'].str.lower()
          image_clean['p2'] = image_clean['p2'].str.lower()
          image_clean['p3'] = image_clean['p3'].str.lower()
```

Test

```
In [250]: image_clean.p1.head()
```

```
Out[250]: 0    welsh_springer_spaniel
          1    redbone
```

```
2    german_shepherd
3    rhodesian_ridgeback
4    miniature_pinscher
Name: p1, dtype: object
```

```
In [251]: image_clean.p2.head()
```

```
Out[251]: 0    collie
          1    miniature_pinscher
          2    malinois
          3    redbone
          4    rottweiler
          Name: p2, dtype: object
```

```
In [252]: image_clean.p3.head()
```

```
Out[252]: 0    shetland_sheepdog
          1    rhodesian_ridgeback
          2    bloodhound
          3    miniature_pinscher
          4    doberman
          Name: p3, dtype: object
```

8 6 archive_clean: clean up text column

Define

```
<li>In the archive_clean table, change the html ampersand code from "&amp ;" to "&" in the text
<li>Remove the "/n " the newline symbol</li>
<li>Remove ending url link.</li>
```

Code

```
In [253]: archive_clean['text'] = archive_clean.text.str.replace("&amp;", "&")
          archive_clean['text'] = archive_clean.text.str.replace("\n", " ")
          archive_clean['text'] = archive_clean.text.str.replace(r"http\S+", "")
          archive_clean['text'] = archive_clean.text.str.strip()
```

Test

```
In [254]: archive_clean.query("text == '&amp;'")
```

```
Out[254]: Empty DataFrame
          Columns: [tweet_id, timestamp, text, rate_num, rate_denom, name, doggo, floof, pupper,
          Index: []
```

```
In [255]: archive_clean.iloc[[588, 797, 853, 948, 985, 1005, 1136, 1234, 1239, 1278,
                              1294, 1307, 1426, 1556, 1592, 1649, 1653, 1719, 1759,
                              1811, 1860, 1922, 1960, 2005, 2014, 2047, 2076], [2,3,4,5]]
```

Out[255]:

766	"Yep... just as I suspected. You're not flossing." 12/10 and 11/10 for the pup n
1007	This is Bookstore and Seaweed. Bookstore is tired and Seaweed is an asshole. 10/
1068	After so many requests, this is Bretagne. She was the last surviving 9/11 search
1165	Happy 4/20 from the squad! 13/10 for all
1202	This is Bluebert. He just saw that both #FinalFur match ups are split 50/50. Ama
1222	Meet Travis and Flurp. Travis is pretty chill but Flurp can't lie down properly.
1359	This is Socks. That water pup w the super legs just splashed him. Socks did not
1459	This may be the greatest video I've ever been sent. 4/10 for Charles the puppy,
1465	Meet Oliviér. He takes killer selfies. Has a dog of his own. It leaps at random
1508	When bae says they can't go out but you see them with someone else that same nig
1525	This is Eriq. His friend just reminded him of last year's super bowl. Not cool f
1538	Meet Fynn & Taco. Fynn is an all-powerful leaf lord and Taco is in the wrong pla
1662	This is Darrel. He just robbed a 7/11 and is in a high speed police chase. Was j
1795	Meet Tassy & Bee. Tassy is pretty chill, but Bee is convinced the Ruffles are ha
1832	These two pups just met and have instantly bonded. Spectacular scene. Mesmerizin
1897	Meet Rufio. He is unaware of the pink legless pupper wrapped around him. Might w
1901	Two gorgeous dogs here. Little waddling dog is a rebel. Refuses to look at camer
1970	Meet Eve. She's a raging alcoholic 8/10 (would b 11/10 but pupper alcoholism is
2010	10/10 for dog. 7/10 for cat. 12/10 for human. Much skill. Would pet all
2064	Meet Holly. She's trying to teach small human-like pup about blocks but he's not
2113	Meet Hank and Sully. Hank is very proud of the pumpkin they found and Sully does
2177	Here we have Pancho and Peaches. Pancho is a Condoleezza Gryffindor, and Peaches
2216	This is Spark. He's nervous. Other dog hasn't moved in a while. Won't come when
2263	This is Kial. Kial is either wearing a cape, which would be rad, or flashing us,
2272	Two dogs in this one. Both are rare Jujitsu Pythagoreans. One slightly whiter th
2306	These are Peruvian Feldspars. Their names are Cupit and Prencer. Both resemble R
2335	This is an Albanian 3 1/2 legged Episcopalian. Loves well-polished hardwood flo

	rate_num	rate_denom	name
766	12	10	None
1007	10	10	Bookstore
1068	9	11	None
1165	4	20	None
1202	50	50	Bluebert
1222	10	10	Travis
1359	9	10	Socks
1459	4	10	None
1465	10	10	Oliviér
1508	5	10	None
1525	10	10	Eriq
1538	11	10	Fynn
1662	7	11	Darrel
1795	10	10	Tassy
1832	10	10	None
1897	10	10	Rufio
1901	5	10	None
1970	8	10	Eve

```
2010  10         10          None
2064  11         10          Holly
2113  11         10          Hank
2177  10         10          None
2216  8          10          Spark
2263  10         10          Kial
2272  7          10          None
2306  10         10          None
2335  1          2           an
```

9 7 archive_clean: fix some of the ratings columns

Define

In the archive_clean table, use several methods such as extractall(), query(), contains(), etc to check for misextraction of the ratings.

Code

```
In [256]: archive_clean.reset_index(inplace=True, drop=True)

In [257]: archive_clean[archive_clean.text.str.contains(r"(\d+\.\d*\/\d+)")][['text', 'rate_num'

Out[257]:
          41    This is Bella. She hopes her smile made you smile. If not, she is also offering
          528   This is Logan, the Chow who lived. He solemnly swears he's up to lots of good. H
          586   This is Sophie. She's a Jubilant Bush Pupper. Super h*ckin rare. Appears at rand
          1474  Here we have uncovered an entire battalion of holiday puppers. Average of 11.26/

                rate_num
          41    5
          528   75
          586   27
          1474  26

In [258]: hyphen_table = archive_clean.text.str.extractall(r"(\d+\d*\/\d+)")
          hyphen_table.head(10)

Out[258]:             0
            match
          0 0       13/10
          1 0       13/10
          2 0       12/10
          3 0       13/10
          4 0       12/10
          5 0       13/10
          6 0       13/10
          7 0       13/10
          8 0       13/10
          9 0       14/10
```

In [259]:
```
match_1 = hyphen_table.query("match == 1")
match_1.head()
```

Out[259]:

		0
	match	
588	1	11/10
797	1	7/10
853	1	14/10
948	1	13/10
985	1	11/10

In [260]:
```
match_1.index.labels
```

Out[260]:
```
FrozenList([[588, 797, 853, 948, 985, 1005, 1136, 1234, 1239, 1278, 1294, 1307, 1426,
```

In [261]:
```
# copied indices from above
archive_clean.iloc[[588, 797, 853, 948, 985, 1005, 1136, 1234, 1239, 1278,
                    1294, 1307, 1426, 1556, 1592, 1649, 1653, 1719, 1759,
                    1811, 1860, 1922, 1960, 2005, 2014, 2047, 2076], [2,3,4,5]]
```

Out[261]:

588	"Yep... just as I suspected. You're not flossing." 12/10 and 11/10 for the pup n			
797	This is Bookstore and Seaweed. Bookstore is tired and Seaweed is an asshole. 10/			
853	After so many requests, this is Bretagne. She was the last surviving 9/11 search			
948	Happy 4/20 from the squad! 13/10 for all			
985	This is Bluebert. He just saw that both #FinalFur match ups are split 50/50. Ama			
1005	Meet Travis and Flurp. Travis is pretty chill but Flurp can't lie down properly.			
1136	This is Socks. That water pup w the super legs just splashed him. Socks did not			
1234	This may be the greatest video I've ever been sent. 4/10 for Charles the puppy,			
1239	Meet Oliviér. He takes killer selfies. Has a dog of his own. It leaps at random			
1278	When bae says they can't go out but you see them with someone else that same nig			
1294	This is Eriq. His friend just reminded him of last year's super bowl. Not cool f			
1307	Meet Fynn & Taco. Fynn is an all-powerful leaf lord and Taco is in the wrong pla			
1426	This is Darrel. He just robbed a 7/11 and is in a high speed police chase. Was j			
1556	Meet Tassy & Bee. Tassy is pretty chill, but Bee is convinced the Ruffles are ha			
1592	These two pups just met and have instantly bonded. Spectacular scene. Mesmerizin			
1649	Meet Rufio. He is unaware of the pink legless pupper wrapped around him. Might w			
1653	Two gorgeous dogs here. Little waddling dog is a rebel. Refuses to look at camer			
1719	Meet Eve. She's a raging alcoholic 8/10 (would b 11/10 but pupper alcoholism is			
1759	10/10 for dog. 7/10 for cat. 12/10 for human. Much skill. Would pet all			
1811	Meet Holly. She's trying to teach small human-like pup about blocks but he's not			
1860	Meet Hank and Sully. Hank is very proud of the pumpkin they found and Sully does			
1922	Here we have Pancho and Peaches. Pancho is a Condoleezza Gryffindor, and Peaches			
1960	This is Spark. He's nervous. Other dog hasn't moved in a while. Won't come when			
2005	This is Kial. Kial is either wearing a cape, which would be rad, or flashing us,			
2014	Two dogs in this one. Both are rare Jujitsu Pythagoreans. One slightly whiter th			
2047	These are Peruvian Feldspars. Their names are Cupit and Prencer. Both resemble R			
2076	This is an Albanian 3 1/2 legged Episcopalian. Loves well-polished hardwood flo			

```
      rate_num  rate_denom       name
588   12        10           None
797   10        10           Bookstore
853   9         11           None
948   4         20           None
985   50        50           Bluebert
1005  10        10           Travis
1136  9         10           Socks
1234  4         10           None
1239  10        10           Oliviér
1278  5         10           None
1294  10        10           Eriq
1307  11        10           Fynn
1426  7         11           Darrel
1556  10        10           Tassy
1592  10        10           None
1649  10        10           Rufio
1653  5         10           None
1719  8         10           Eve
1759  10        10           None
1811  11        10           Holly
1860  11        10           Hank
1922  10        10           None
1960  8         10           Spark
2005  10        10           Kial
2014  7         10           None
2047  10        10           None
2076  1         2            an
```

In [262]:
```
#rating confused with 9/11(September 11th)
archive_clean.iloc[853, 3] = 14
archive_clean.iloc[853, 4] = 10

#rating confused with 4/20(Weed Day)
archive_clean.iloc[948, 3] = 13
archive_clean.iloc[948, 4] = 10

#rating confused with phrase 50/50 split
archive_clean.iloc[985, 3] = 11
archive_clean.iloc[985, 4] = 10

#rating confused with 7/11 which is name of convience store
archive_clean.iloc[1426, 3] = 10
archive_clean.iloc[1426, 4] = 10

#rating confused with 1/2 representing "half"
archive_clean.iloc[2076, 3] = 9
archive_clean.iloc[2076, 4] = 10
```

In [263]:
```
doubles_list = archive_clean.iloc[[588, 797, 1005, 1136, 1234, 1239, 1278,
                        1294, 1307, 1556, 1592, 1649, 1653, 1719, 1759,
                        1811, 1860, 1922, 1960, 2005, 2014, 2047]]
double_index = doubles_list.index
```

In [264]:
```
archive_clean.iloc[[41, 528, 586, 1474], [2,3,4]]
```

Out[264]:

```
41    This is Bella. She hopes her smile made you smile. If not, she is also offering
528   This is Logan, the Chow who lived. He solemnly swears he's up to lots of good. H
586   This is Sophie. She's a Jubilant Bush Pupper. Super h*ckin rare. Appears at rand
1474  Here we have uncovered an entire battalion of holiday puppers. Average of 11.26/

      rate_num  rate_denom
41    5         10
528   75        10
586   27        10
1474  26        10
```

In [265]:
```
archive_clean.iloc[41, 3] = 13.5
archive_clean.iloc[528, 3] = 9.75
archive_clean.iloc[586, 3] = 11.27
archive_clean.iloc[1474, 3] = 11.26
```

Test

In [266]:
```
archive_clean.iloc[[45, 528, 586, 1474], [2,3,4]]
```

Out[266]:

```
45    This is Gus. He's quite the cheeky pupper. Already perfected the disinterested w
528   This is Logan, the Chow who lived. He solemnly swears he's up to lots of good. H
586   This is Sophie. She's a Jubilant Bush Pupper. Super h*ckin rare. Appears at rand
1474  Here we have uncovered an entire battalion of holiday puppers. Average of 11.26/

      rate_num  rate_denom
45    12.00     10
528   9.75      10
586   11.27     10
1474  11.26     10
```

In [267]:
```
archive_clean.iloc[[853, 948, 985, 1426, 2076], [2,3,4,5]]
```

Out[267]:

```
853   After so many requests, this is Bretagne. She was the last surviving 9/11 search
948   Happy 4/20 from the squad! 13/10 for all
985   This is Bluebert. He just saw that both #FinalFur match ups are split 50/50. Ama
1426  This is Darrel. He just robbed a 7/11 and is in a high speed police chase. Was j
2076  This is an Albanian 3 1/2 legged  Episcopalian. Loves well-polished hardwood flo
```

```
      rate_num  rate_denom      name
853   14.0      10          None
948   13.0      10          None
985   11.0      10          Bluebert
1426  10.0      10          Darrel
2076  9.0       10          an
```

10 8 archive_clean: removing. doubles

Define

In the archive_clean table, there are some tweets with two dogs being rated, therefore those

Code

```
In [268]: doubles_list = archive_clean.iloc[[588, 797, 1005, 1136, 1234, 1239, 1278,
                                1294, 1307, 1556, 1592, 1649, 1653, 1719, 1759,
                                1811, 1860, 1922, 1960, 2005, 2014, 2047]]
          double_index = doubles_list.index

In [269]: archive_clean.drop(axis='index', index=double_index, inplace=True)
```

Test

```
In [270]: archive_clean.info()

<class 'pandas.core.frame.DataFrame'>
Int64Index: 2075 entries, 0 to 2096
Data columns (total 10 columns):
tweet_id      2075 non-null object
timestamp     2075 non-null datetime64[ns]
text          2075 non-null object
rate_num      2075 non-null float64
rate_denom    2075 non-null int64
name          2075 non-null object
doggo         2075 non-null object
floof         2075 non-null object
pupper        2075 non-null object
puppo         2075 non-null object
dtypes: datetime64[ns](1), float64(1), int64(1), object(7)
memory usage: 178.3+ KB
```

11 9 MERGE

Define

Take both the archive_clean and tweet_clean tables and merge into one table using the join() method on the columns tweet_id.

Code

```
In [271]: df_merge1 = archive_clean.join(tweet_clean.set_index('tweet_id'), on='tweet_id')
```

Test

```
In [272]: df_merge1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2075 entries, 0 to 2096
Data columns (total 12 columns):
tweet_id          2075 non-null object
timestamp         2075 non-null datetime64[ns]
text              2075 non-null object
rate_num          2075 non-null float64
rate_denom        2075 non-null int64
name              2075 non-null object
doggo             2075 non-null object
floof             2075 non-null object
pupper            2075 non-null object
puppo             2075 non-null object
retweet_count     0 non-null object
favorite_count    0 non-null object
dtypes: datetime64[ns](1), float64(1), int64(1), object(9)
memory usage: 210.7+ KB
```

12 10 Final Merge

Define

Take the newly df_merge1 table and combine with the image_clean table using the same join() method on the tweet_id column.

Code

```
In [273]: master = df_merge1.join(image_clean.set_index('tweet_id'), on='tweet_id')
```

Test

```
In [274]: master.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2075 entries, 0 to 2096
Data columns (total 23 columns):
tweet_id          2075 non-null object
timestamp         2075 non-null datetime64[ns]
text              2075 non-null object
rate_num          2075 non-null float64
```

```
rate_denom          2075 non-null int64
name                2075 non-null object
doggo               2075 non-null object
floof               2075 non-null object
pupper              2075 non-null object
puppo               2075 non-null object
retweet_count       0 non-null object
favorite_count      0 non-null object
jpg_url             1949 non-null object
img_num             1949 non-null float64
p1                  1949 non-null object
p1_conf             1949 non-null float64
p1_dog              1949 non-null object
p2                  1949 non-null object
p2_conf             1949 non-null float64
p2_dog              1949 non-null object
p3                  1949 non-null object
p3_conf             1949 non-null float64
p3_dog              1949 non-null object
dtypes: datetime64[ns](1), float64(5), int64(1), object(16)
memory usage: 389.1+ KB
```

12.1 misssing data

Define

Removing the missing rows from the merged tables using the drop() method.

Code

```
In [275]: master_copy = master.copy()

In [276]: drop_index = master_copy[pd.isnull(master_copy['jpg_url'])].index
          drop_index2 = master_copy[pd.isnull(master_copy['retweet_count'])].index
          drop_index, drop_index2

Out[276]: (Int64Index([  32,   38,   65,   73,   78,   95,  113,  153,  155,  193,
                         ...
                         1505, 1512, 1522, 1537, 1552, 1568, 1579, 1594, 1666, 1956],
                        dtype='int64', length=126),
           Int64Index([   0,    1,    2,    3,    4,    5,    6,    7,    8,    9,
                         ...
                         2087, 2088, 2089, 2090, 2091, 2092, 2093, 2094, 2095, 2096],
                        dtype='int64', length=2075))

In [277]: master_copy.drop(index=drop_index, inplace=True)
          master_copy.drop(index=drop_index2, inplace=True)
```

```
KeyError                                  Traceback (most recent call last)

<ipython-input-277-53e321bc5005> in <module>()
  1 master_copy.drop(index=drop_index, inplace=True)
----> 2 master_copy.drop(index=drop_index2, inplace=True)


/opt/conda/lib/python3.6/site-packages/pandas/core/frame.py in drop(self, labels, axis,
3695                                            index=index, columns=columns,
3696                                            level=level, inplace=inplace,
-> 3697                                            errors=errors)
3698
3699     @rewrite_axis_style_signature('mapper', [('copy', True),


/opt/conda/lib/python3.6/site-packages/pandas/core/generic.py in drop(self, labels, axis
3109         for axis, labels in axes.items():
3110             if labels is not None:
-> 3111                 obj = obj._drop_axis(labels, axis, level=level, errors=errors)
3112
3113         if inplace:


/opt/conda/lib/python3.6/site-packages/pandas/core/generic.py in _drop_axis(self, labels
3141                 new_axis = axis.drop(labels, level=level, errors=errors)
3142             else:
-> 3143                 new_axis = axis.drop(labels, errors=errors)
3144             result = self.reindex(**{axis_name: new_axis})
3145


/opt/conda/lib/python3.6/site-packages/pandas/core/indexes/base.py in drop(self, labels,
4402             if errors != 'ignore':
4403                 raise KeyError(
-> 4404                     '{} not found in axis'.format(labels[mask]))
4405             indexer = indexer[~mask]
4406         return self.delete(indexer)


KeyError: '[  32   38   65   73   78   95  113  153  155  193  195  205  216  226  253\n
```

Test

In [279]: master_copy.info()

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1949 entries, 0 to 2096
```

```
Data columns (total 23 columns):
tweet_id          1949 non-null object
timestamp         1949 non-null datetime64[ns]
text              1949 non-null object
rate_num          1949 non-null float64
rate_denom        1949 non-null int64
name              1949 non-null object
doggo             1949 non-null object
floof             1949 non-null object
pupper            1949 non-null object
puppo             1949 non-null object
retweet_count     0 non-null object
favorite_count    0 non-null object
jpg_url           1949 non-null object
img_num           1949 non-null float64
p1                1949 non-null object
p1_conf           1949 non-null float64
p1_dog            1949 non-null object
p2                1949 non-null object
p2_conf           1949 non-null float64
p2_dog            1949 non-null object
p3                1949 non-null object
p3_conf           1949 non-null float64
p3_dog            1949 non-null object
dtypes: datetime64[ns](1), float64(5), int64(1), object(16)
memory usage: 445.4+ KB
```

```
In [280]: master_copy.to_csv(path_or_buf='master.csv', index=False)
```

```
In [281]: master_clean = master_copy[['timestamp','tweet_id', 'rate_num', 'rate_denom',
                                      'name', 'retweet_count', 'favorite_count', 'p1', 'p2', 'p3
```

13 Analyzing Data

Now we wanna take the final DataFrame and set the index to the timestamp column in order to gather some time series analysis. Twitter data is very time specific and would be nice visually to see changes over time.

```
In [282]: #setting the DataFrame to index the timestamp column
          master_copy = master_copy.set_index('timestamp')
```

```
In [283]: master_copy.sample(5)
```

```
Out[283]:                                tweet_id  \
          timestamp
          2016-03-19 01:54:56  711008018775851008
          2015-11-28 05:05:47  670468609693655041
```

```
2016-10-31 22:00:04  793210959003287553
2017-03-13 00:02:39  841077006473256960
2016-01-21 02:56:40  690005060500217858


timestamp
2016-03-19 01:54:56  This is Chuckles. He had a balloon but he accidentally let go of
2015-11-28 05:05:47  This is Edd. He's a Czechoslovakian Googolplex Merlot. Ready for
2016-10-31 22:00:04  This is Maude. She's the h*ckin happiest wasp you've ever seen. 1
2017-03-13 00:02:39  This is Dawn. She's just checking pup on you. Making sure you're
2016-01-21 02:56:40  "I'm the only one that ever does anything in this household" 10/1

                     rate_num  rate_denom      name doggo floof  pupper puppo  \
timestamp
2016-03-19 01:54:56  11.0      10           Chuckles  None  None  pupper  None
2015-11-28 05:05:47  10.0      10           Edd       None  None  None    None
2016-10-31 22:00:04  10.0      10           Maude     None  None  None    None
2017-03-13 00:02:39  12.0      10           Dawn      None  None  None    None
2016-01-21 02:56:40  10.0      10           None      None  None  None    None

                    retweet_count  ...   img_num               p1   p1_conf  \
timestamp                          ...
2016-03-19 01:54:56  NaN           ...    1.0     french_bulldog    0.731405
2015-11-28 05:05:47  NaN           ...    1.0     minivan           0.730152
2016-10-31 22:00:04  NaN           ...    1.0     doormat           0.874431
2017-03-13 00:02:39  NaN           ...    1.0     brittany_spaniel  0.962985
2016-01-21 02:56:40  NaN           ...    1.0     samoyed           0.270287

                    p1_dog                p2   p2_conf p2_dog           p3  \
timestamp
2016-03-19 01:54:56  True   boston_bull       0.150672  True   pug
2015-11-28 05:05:47  False  beach_wagon       0.078661  False  car_wheel
2016-10-31 22:00:04  False  french_bulldog    0.018759  True   boston_bull
2017-03-13 00:02:39  True   blenheim_spaniel  0.014820  True   clumber
2016-01-21 02:56:40  True   great_pyrenees    0.114027  True   teddy

                      p3_conf p3_dog
timestamp
2016-03-19 01:54:56  0.021811  True
2015-11-28 05:05:47  0.064346  False
2016-10-31 22:00:04  0.015134  True
2017-03-13 00:02:39  0.009557  True
2016-01-21 02:56:40  0.072475  False

[5 rows x 22 columns]
```

Basic Statistics

```
In [284]: # viewing some descriptive statistics with the quantitative measures in our analysis.
```

```
# used round(5) in order to visually see the last two columns without scientific notat
master_copy.describe().round(5)
```

Out[284]:

	rate_num	rate_denom	img_num	p1_conf	p2_conf	p3_conf
count	1949.00000	1949.00000	1949.00000	1949.00000	1949.00000	1949.00000
mean	12.20615	10.46024	1.20267	0.59426	0.13431	0.06021
std	41.82962	6.82715	0.56094	0.27237	0.10088	0.05094
min	0.00000	7.00000	1.00000	0.04433	0.00000	0.00000
25%	10.00000	10.00000	1.00000	0.36046	0.05296	0.01619
50%	11.00000	10.00000	1.00000	0.58776	0.11740	0.04947
75%	12.00000	10.00000	1.00000	0.84699	0.19540	0.09150
max	1776.00000	170.00000	4.00000	1.00000	0.48801	0.27342

As shown above, each variable initially seems to follow a logical pattern without abnormal c

Even, though some of the numerators in the rate_num column are extraordinarily high and abov
Also, for columns p1_conf, p2_conf, and p3_conf, the numbers are in the bounds from 0 to 1,

This suggests that the neural network developed to identify the breed of dog works effective

Correlation

In [285]:
```
# this method helps to see which variables correlate and helpful when doing a hypothes
master_copy.corr()
```

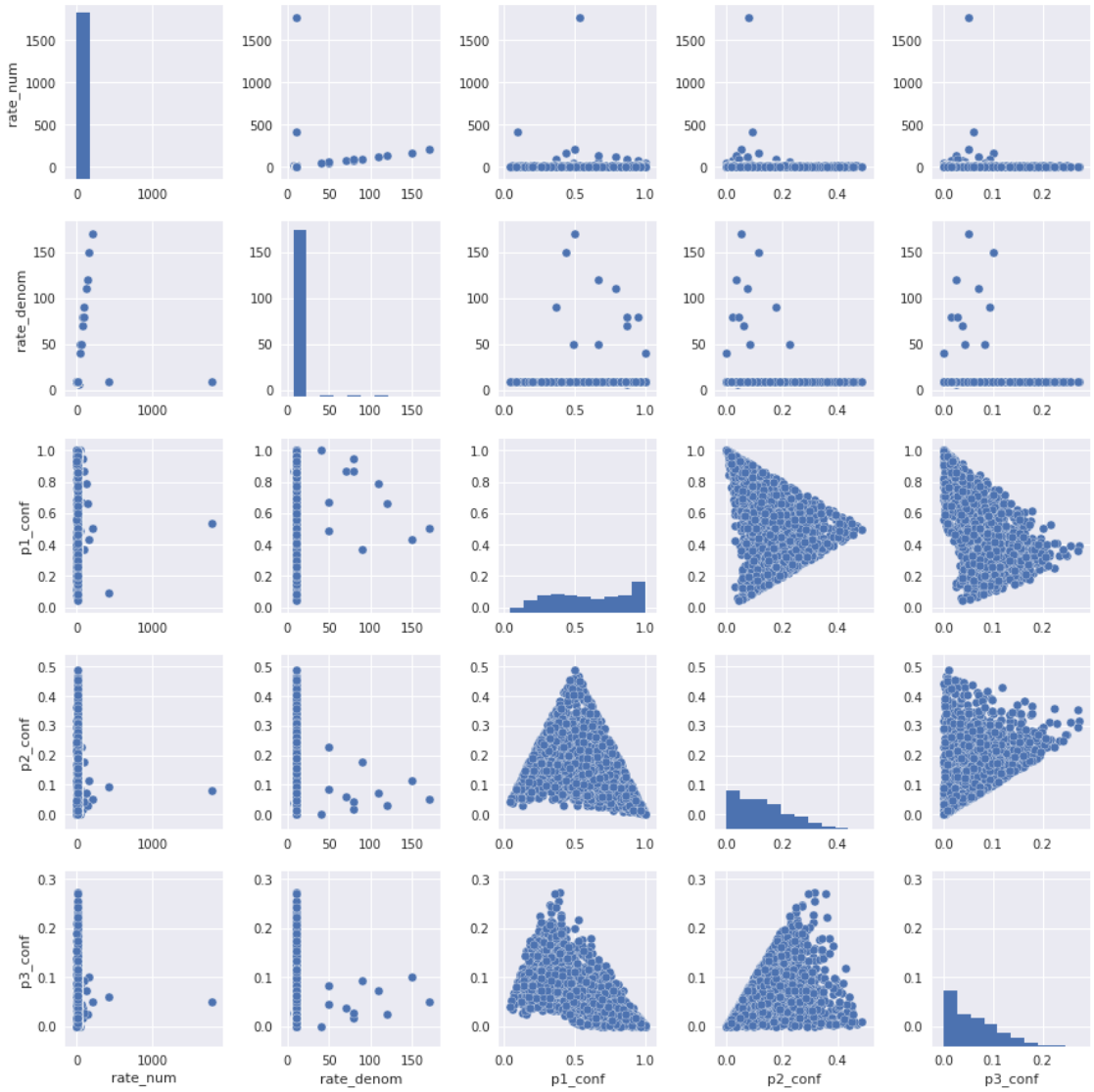
Out[285]:

	rate_num	rate_denom	img_num	p1_conf	p2_conf	p3_conf
rate_num	1.000000	0.184969	-0.002635	-0.006098	-0.020520	-0.007058
rate_denom	0.184969	1.000000	-0.016326	0.011969	-0.038065	-0.006542
img_num	-0.002635	-0.016326	1.000000	0.204818	-0.158799	-0.139215
p1_conf	-0.006098	0.011969	0.204818	1.000000	-0.509777	-0.707192
p2_conf	-0.020520	-0.038065	-0.158799	-0.509777	1.000000	0.480561
p3_conf	-0.007058	-0.006542	-0.139215	-0.707192	0.480561	1.000000

The correlation chart is useful for finding connections between variables, especially with h

The correlation coefficient beteen retweet_count and favorite_count is 0.929604, which is cl
The only other coefficient worth mentioning is the between p1_conf and p3_conf which is -0.7

In [287]:
```
sns.pairplot(master_copy, vars=["rate_num", "rate_denom", "p1_conf", "p2_conf", "p3_co
```



Word cloud with tweets

```
<li>A word cloud is a fun tool that lets user take the most frequently used words from a text, i
<li>The outline of a paw print was used for this word cloud in association with the @WeRateDogs
```

First we need to create a list with all the words that were tweeted in our DataFrame.

```
In [80]: tweets = np.array(master_copy.text)
         my_list = []
         for tweet in tweets:
             my_list.append(tweet.replace("\n",""))
```

Next, we downloaded an image of a paw print from the internet and used it in the function below to generate a word cloud with the tweets.

```
In [81]: mask = np.array(Image.open(requests.get('https://clipartix.com/wp-content/uploads/2016/
         text = my_list

In [82]: def gen_wc(text, mask):
             word_cloud = WordCloud(width = 500, height = 500, background_color='white', mask=ma
             plt.figure(figsize=(10,8),facecolor = 'white', edgecolor='red')
             plt.imshow(word_cloud)
             plt.axis('off')
             plt.tight_layout(pad=0)
             plt.show()
```

The code used above was modeled from this blog on how to generate a word cloud in python. ht

```
In [83]: gen_wc(text, mask)
```