Data Wrangling Analysis on WeRateDog

Real-world data rarely comes clean. I made use of Python and its libraries for this project, I gathered data from a variety of sources and in a variety of formats, assess its quality and tidiness, then clean it. I also document my wrangling efforts in a Jupyter Notebook, plus showcase them through analyses and visualizations. The goal for this project is to create interesting and trustworthy analyses and visualizations and it's include accessing data from different sources and in different format.

Let's know what WeRateDog is all about

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

A golden retriever named Stuart

Example of Tweet from WeRateDogs

Project Context

In this Project, I made use of 3 Dataset. 2 out of the 2 dataset was made available for me by my Instructors(Udacity) and I was to generate the other Dataset using Twitter API. Unfortunately I couldn't have access to a twitter developer account so I made use of the other option of getting the API provided by my instructors.

- The first Dataset is **twitter_archive_enhanced.csv** which was provided by udacity Instructors filtered for tweets with ratings only. It contain 2356 tweets which was downloaded programmatically by Udacity Instructors. I manually downloaded and read the Dataset into my working environment (Jupyter Notebook).

A view of the First Dataset Below:

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

- The second Data set was hosted on udacity server. It was produced by running every image in the WeRateDogs Twitter archive through a neural network that can classify breeds of dogs. The results was a table full of image predictions (the top three only) alongside each tweet ID, image URL, and the image number that corresponded to the most confident prediction (numbered 1 to 4 since tweets can have up to four images). I downloaded the file programmatically using the Requests library and the URL provided and I saved it into a tsv file name **image_prediction.tsv**.

A view of the second Dataset Below:

	tweet_id	jpg_url	img_num	p1	p1_conf	p1_dog	p2	p2_conf	p2_dog	
0	666020888022790149	https://pbs.twimg.com/media/CT4udn0WwAA0aMy.jpg	1	Welsh_springer_spaniel	0.465074	True	collie	0.156665	True	Shetland_sheep
1	666029285002620928	https://pbs.twimg.com/media/CT42GRgUYAA5iDo.jpg	1	redbone	0.506826	True	miniature_pinscher	0.074192	True	Rhodesian_ridgeb
2	666033412701032449	https://pbs.twimg.com/media/CT4521TWwAEvMyu.jpg	1	German_shepherd	0.596461	True	malinois	0.138584	True	bloodho
3	666044226329800704	https://pbs.twimg.com/media/CT5Dr8HUEAA-lEu.jpg	1	Rhodesian_ridgeback	0.408143	True	redbone	0.360687	True	miniature_pinsc
4	666049248165822465	https://pbs.twimg.com/media/CT5IQmsXIAAKY4A.jpg	1	miniature_pinscher	0.560311	True	Rottweiler	0.243682	True	Doberr
...	...	...	...	...	...	...	...	...	...	
2070	891327558926688256	https://pbs.twimg.com/media/DF6hr6BUMAAzZgT.jpg	2	basset	0.555712	True	English_springer	0.225770	True	German_sh haired_poi
2071	891689557279858688	https://pbs.twimg.com/media/DF_q7IAWsAEuuN8.jpg	1	paper_towel	0.170278	False	Labrador_retriever	0.168086	True	spa
2072	891815181378084864	https://pbs.twimg.com/media/DGBdLU1WsAANxJ9.jpg	1	Chihuahua	0.716012	True	malamute	0.078253	True	ke
2073	892177421306343426	https://pbs.twimg.com/media/DGGmoV4XsAAUL6n.jpg	1	Chihuahua	0.323581	True	Pekinese	0.090647	True	papi

- The third Dataset is to be scrapped from Twitter API, I did all the necessary application to apply for a twitter developer account which will enable me to scrap the data from the twitter API but unfortunately I wasn't granted access to the account so I had to use the other option which was made available to me by my instructor. I copied the code made available by my instructor into jupyter notebook and I downloaded the Twitter API data made available, read it into a dataframe and saved it as a .csv file.

	id	retweet_count	favorite_count
0	892420643555336193	8853	39467
1	892177421306343426	6514	33819
2	891815181378084864	4328	25461
3	891689557279858688	8964	42908
4	891327558926688256	9774	41048
...	...	...	...
2349	666049248165822465	41	111
2350	666044226329800704	147	311
2351	666033412701032449	47	128
2352	666029285002620928	48	132
2353	666020888022790149	532	2535

2354 rows × 3 columns

A view of the third Dataset Above:

The third Dataset involves using the tweet IDs in the WeRateDogs twitter archive to query the Twitter API for each tweets JSON data using Python's [Tweepy] library and gather **each tweet's retweet count** and **favorite ("like") count** at the minimum and any additional data you find interesting. After which I will store each tweet's entire set of JSON data in a file called tweet_json.txt file.

Each tweet's JSON data was written to its own line. Then read this .txt file line by line into a pandas DataFrame with (at minimum) **tweet ID, retweet count, and favorite count**.

Accessing the Dataset

I Accessed the 3 Dataset visually and programmatically and I was able to find 9 Quality Issues and 2 Tidiness Issue.

Cleaning the Dataset

Before cleaning I made a copy of each of the data set and I cleaned the Issues I found using the Define, Code and Test method, after succesfull cleaning I merged the three Dataset into one dataset and I saved it as twitter_archive_master.csv

I obtain the following Insights from the merged Dataset

```
#looking at the description of our master dataset to draw insights from it
master.describe()
```

	tweet_id	rating_numerator	rating_denominator	img_num	p1_conf	p2_conf	p3_conf	retweet_count	favorite_count
count	1.994000e+03	1994.000000	1994.000000	1994.000000	1994.000000	1.994000e+03	1.994000e+03	1994.000000	1994.000000
mean	7.358508e+17	12.280843	10.532096	1.203109	0.593941	1.344195e-01	6.024848e-02	2766.753260	8895.725677
std	6.747816e+16	41.497718	7.320710	0.560777	0.271954	1.006807e-01	5.089067e-02	4674.698447	12213.193181
min	6.660209e+17	0.000000	2.000000	1.000000	0.044333	1.011300e-08	1.740170e-10	16.000000	81.000000
25%	6.758475e+17	10.000000	10.000000	1.000000	0.362857	5.393987e-02	1.619283e-02	624.750000	1982.000000
50%	7.084748e+17	11.000000	10.000000	1.000000	0.587635	1.174550e-01	4.950530e-02	1359.500000	4136.000000
75%	7.877873e+17	12.000000	10.000000	1.000000	0.846285	1.951377e-01	9.159438e-02	3220.000000	11308.000000
max	8.924206e+17	1776.000000	170.000000	4.000000	1.000000	4.880140e-01	2.734190e-01	79515.000000	132810.000000

```python
#Checking the percentage of the Dog names to obtain insights
master.name.value_counts(normalize =True) * 100
```

```
None          27.382146
a              2.758275
Charlie        0.551655
Lucy           0.501505
Cooper         0.501505
                 ...
Bookstore      0.050150
Shiloh         0.050150
Burt           0.050150
Gustav         0.050150
Christoper     0.050150
Name: name, Length: 936, dtype: float64
```

```python
#Checking the percentage of our dog stages to obtain insights
master.stage.value_counts(normalize =True) * 100
```

```
pupper            66.339869
doggo             20.588235
puppo              7.189542
doggo, pupper      2.941176
floofer            2.287582
doggo, puppo       0.326797
doggo, flooper     0.326797
Name: stage, dtype: float64
```

```python
#checking the percentage of our image number to obtain insights
master.img_num.value_counts(normalize=True)*100
```

```
1    85.807422
2     9.578736
3     3.109328
4     1.504514
Name: img_num, dtype: float64
```

Insights

- After studying and looking through master Data set, I found out that the dog stage "Pupper" is the most popular Dog stage with over 66% followed by "Doggo" with over 20% popularity and the least popular dog stages is "doggo, puppo" and "doggo, flooper".

- I also found out that the most dominant Image number according to neural network prediction is "1" with over 85% dominancy.

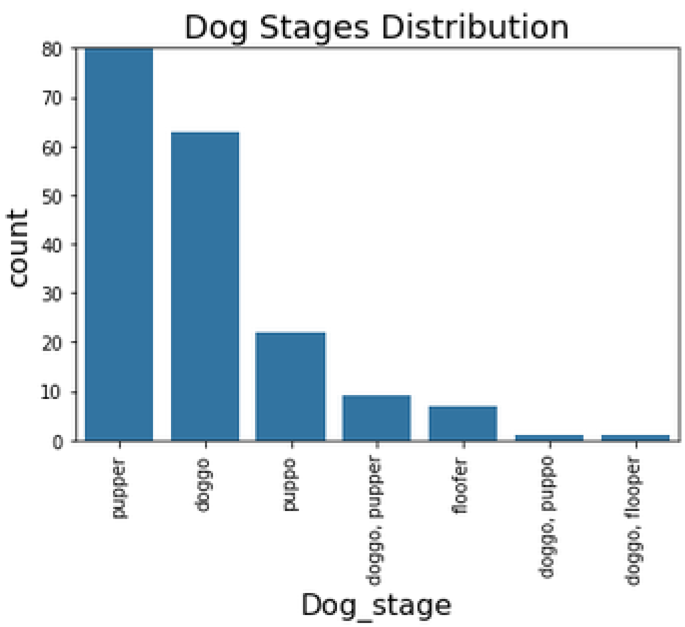
- After careful Analysis I found out that there is a linear relationship between retweet count and favorite count, favorite count tends to rise as retweet count rises.

- According to the statistical description of the Dataset, the minimum retweet count is 16 while the maximum retweet count is 79515.

Visualization

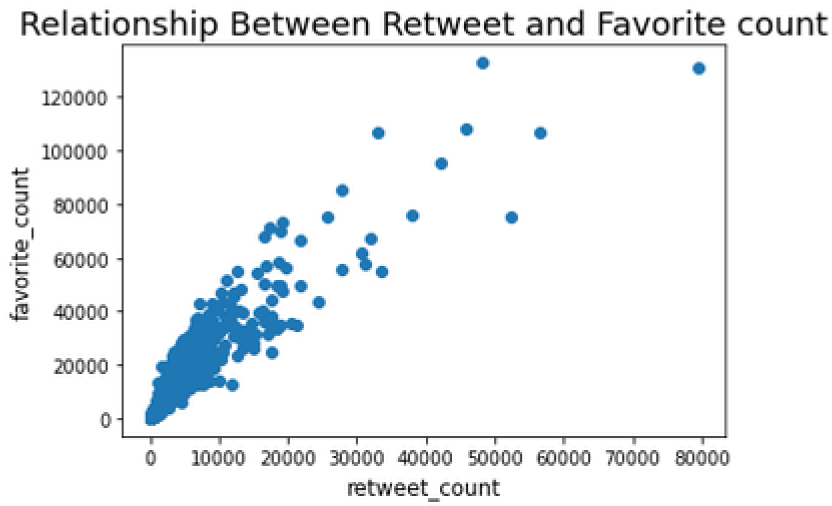
- **The most Popular Dog Stage is 'Pupper'**

The most popular dog stage according to their order or popularity as displayed by the charts below.



- **The Relationship between Retweet count and Favorite count,**

From the scatter plot below there is a positive linear Relationship between favorite count and Retweet Count. Using the Pearson correlation coefficient, there is a strong linear relationship between Retweet count and favorite count as increase in Retweet count tends to increase the Favorite count. Which means that the most popular tweets by WeRateDog gets the highest retweet count and favorite count.



3. The most frequent Image Number according to the Neural Network Prediction is '1'

In the Pie chart below it can be seen that the most frequent Image Number that correspond with the Neural Network prediction is '1' with About 85% of frequency when calculated then the other image Number.