# WeRateDogs - Twitter Data

## Introduction

Real-world data rarely comes clean. Using Python and its libraries, we will gather data from a variety of sources and in a variety of formats, assess its quality and tidiness, then clean it. This is called data wrangling.

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

# **Key Points that were specified by Udacity:**

Key points to keep in mind when data wrangling for this project:

- You only want original ratings (no retweets) that have images. Though there are 5000+ tweets in the dataset, not all are dog ratings and some are retweets.
- Assessing and cleaning the entire dataset completely would require a lot of time, and is not necessary to practice and demonstrate your skills

in data wrangling. Therefore, the requirements of this project are only to assess and clean at least 8 quality issues and at least 2 tidiness issues in this dataset.

- Cleaning includes merging individual pieces of data according to the rules of tidy data.
- The fact that the rating numerators are greater than the denominators does not need to be cleaned. This unique rating system is a big part of the popularity of WeRateDogs.
- You do not need to gather the tweets beyond August 1st, 2017. You can,
   but note that you won't be able to gather the image predictions for
   these tweets since you don't have access to the algorithm used.

# **Wrangling efforts:**

# i) Gathering Data

• The WeRateDogs Twitter archive, was given by course instructor. The file manually downloaded by clicking the following link:

```
twitter archive enhanced.csv
```

The tweet image predictions, i.e., what breed of dog (or other object, animal, etc.) was present in each tweet according to a neural network.
 The file (image\_predictions.tsv) was hosted on Udacity's servers and was downloaded programmatically using the Requests library and the following URL:

https://d17h27t6h515a5.cloudfront.net/topher/2017/August/599fd2ad\_imag e-predictions/image-predictions.tsv

• Each tweet's retweet count and favorite ("like") count at minimum, and any additional data we find interesting. The tweet json.txt was

directly downloaded from udacity as there were some issues using twitter API.

# **Assessing Data for this Project**

After gathering each of the above pieces of data, assessed them visually and programmatically for quality and tidiness issues. The assessed document have least eight (8) quality issues and two (2) tidiness issues in our wrangle\_act.ipynb Jupyter Notebook.

### Dataset Issues found in archive dataset:

### i.QUALITY:

- Missing values in [in\_reply\_to\_status\_id, in\_reply\_to\_user\_id, retweeted\_status\_id, retweeted\_status\_user\_id, retweeted\_status\_timestamp, expanded\_urls] columns
- 2. Rating\_numerator and rating\_denominator had some inconsistent values in the numerator and denominator.
  - 3. tweet id 835246439529840640 had a rating of denominator = 0
- 4. in\_reply\_to\_status\_id, in\_reply\_to\_user\_id, retweeted\_status\_id, retweeted\_status\_user\_id must be integers instead of float ( They had id's similar to tweet id)
- 5. timestamp and retweeted\_status\_timestamp were object but they should be datetime.
- 6. The columns doggo, floofer, pupper, puppo had missing values has None instead of NaN

7. Dogs name such as - 'a', 'actually', 'all', 'an', 'by', 'getting', 'his', 'incredibly', 'infuriating', 'just', 'life', 'light', 'mad', 'my', 'not', 'officially', 'old', 'one', 'quite', 'space', 'such', 'the', 'this', 'unacceptable', 'very' were not likely.

#### ii.Tidiness:

1. Dog stages were found in multiple columns, They should be passed in under a single column. That reduced the dimensionality of the dataframe

### Dataset Issues found in tweet dataset:

#### i.QUALITY:

- 1. Missing values in [geo, coordinates, place, contributors, possibly\_sensitive, possibly\_sensitive\_appealable, retweeted\_status, quoted\_status\_id, quoted\_status\_id\_str, quoted\_status, in\_reply\_to\_status\_id, in\_reply\_to\_user\_id, retweeted\_status\_id , retweeted\_status\_user\_id, extended\_entities ] columns
  - 2. We needed to remove retweet and replay

#### ii.Tidiness:

- 1. User column had data in dictionaries and have several unrequired data stored, we needed followers\_count separately so we could access them easily.
- 2. Retweets and Favorites had to be joined to the archive data table, because all the tweets information was found there.

# Dataset Issues found in ip (image-predictions) dataset:

#### i.QUALITY:

1. Only 2075 tweet\_id have images.

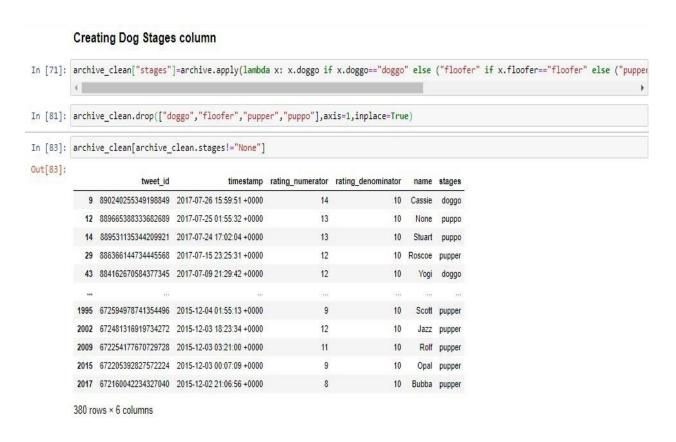
#### ii.Tidiness:

1. All the prediction outputs from different algorithms have to be joined with archive and tweet.

# **Cleaning Data**

Cleaning few of the issues you documented while assessing. Performed these cleaning in wrangle\_act.ipynb.

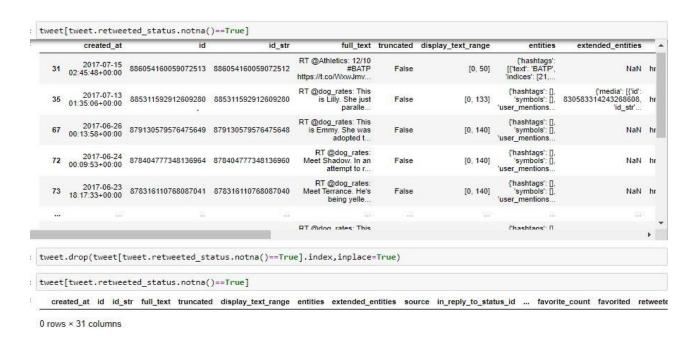
Creating a single column for dog stages and removing "doggo", "floofer",
 "pupper" and "puppo" columns.



### Removing Reply Tweets



### Removing Retweets



- Removal of other unnecessary columns from the datasets.
- Timestamp Datatype

### Changing timestamp object to datetime

```
In [170]: archive_clean["timestamp"] = pd.to_datetime(archive_clean.timestamp)
```

• Changing unacceptable names

```
Changing unacceptable names to "unknown"

145]: cr_name=["None",'a', 'actually', 'all', 'an', 'by', 'getting','his', 'incredibly', 'infuriating', 'just', 'life', 'light', 'mad', archive_clean.name=archive_clean.name.apply(lambda x:"Unknown" if (x in cr_name) else x)

171]: archive_clean.query("name in ['a', 'actually', 'all', 'an', 'by', 'getting','his', 'incredibly', 'infuriating', 'just', 'life', 'archive_clean.query("name in ['a', 'actually', 'all', 'an', 'by', 'getting','his', 'incredibly', 'infuriating', 'just', 'life', 'archive_clean.query("name in ['a', 'actually', 'all', 'an', 'by', 'getting','his', 'incredibly', 'infuriating', 'just', 'life', 'archive_clean.query("name in ['a', 'actually', 'all', 'an', 'by', 'getting','his', 'incredibly', 'infuriating', 'just', 'life', 'archive_clean.query("name in ['a', 'actually', 'all', 'an', 'by', 'getting','his', 'incredibly', 'infuriating', 'just', 'life', 'archive_clean.query("name in ['a', 'actually', 'all', 'an', 'by', 'getting','his', 'incredibly', 'infuriating', 'just', 'life', 'archive_clean.query("name in ['a', 'actually', 'all', 'an', 'by', 'getting','his', 'incredibly', 'infuriating', 'just', 'life', 'archive_clean.query("name in ['a', 'actually', 'all', 'an', 'by', 'getting','his', 'incredibly', 'infuriating', 'just', 'life', 'archive_clean.query("name in ['a', 'actually', 'all', 'an', 'by', 'getting', 'his', 'incredibly', 'infuriating', 'just', 'life', 'archive_clean.query("name in ['a', 'actually', 'all', 'an', 'by', 'getting', 'his', 'incredibly', 'infuriating', 'just', 'life', 'archive_clean.query("name in ['a', 'actually', 'all', 'an', 'by', 'getting', 'his', 'incredibly', 'infuriating', 'just', 'life', 'archive_clean.query("name in ['a', 'actually', 'all', 'an', 'by', 'getting', 'his', 'archive_clean.query("name in ['a', 'actually', 'all', 'an', 'actually', 'all', 'archive_clean.query("name in ['a', 'actually', 'all', 'archive_clean.query("name in ['a', 'actually', 'all', 'actually', 'all', 'actually', 'all', 'actually', 'all', 'actually', 'all',
```

• Fixing Non-Integer rating\_numerator issue

#### Fixing Non-Integer rating\_numerator Issue

```
In [185]: pattern = (\langle d+ \rangle, \langle d+ \rangle)
            merge1.full_text.str.extract(pattern, expand = True)[0].dropna()
Out[185]: 39
             499
                       9.75/10
             549
                     11.27/10
             1359
                     11.26/10
            Name: 0, dtype: object
In [186]: | num = merge1.full_text.str.extract(pattern, expand = True)[0].dropna().str.split('/', n=1, expand=True)[0]
In [187]: num index = num.index
            num_values = num.values.astype("float64")
In [188]: merge1.rating_numerator = merge1.rating_numerator.astype("float64")
    merge1.rating_denominator = merge1.rating_denominator.astype("float64")
            mergel.loc[num_index, "rating_numerator"] = num_values
mergel.loc[num_index].rating_numerator
Out[188]: 39
                      13.50
            499
                       9.75
            549
                      11.27
            1359
                     11.26
            Name: rating_numerator, dtype: float64
```

The result dataset is a high quality and tidy pandas DataFrame that is used for Analyzing and Visualizing the data.