There are 3 datasets required for this analysis. They are:

The twitter_archive_enhanced.csv dataset

The image_predictions.tsv dataset

The Twitter API dataset

Columns within each dataset:

The **twitter_archive_enhanced.csv** has the following columns:

1. **tweet_id**: This is the unique identifier for each tweet
2. **in_reply_to_status_id**: This is a unique identifier for a tweet in response to a Twitter status.
3. **in_reply_to_user_id**: This is a unique identifier for a tweet in response to a user's tweet.
4. **Timestamp**: This gives the date and time a tweet was posted.
5. **Source**: This gives the device from which a tweet was made (i.e. phone, laptop, etc)
6. **Text**: This refers to the content of the tweet itself.
7. **retweeted_status_id**: This is the unique identifier for a Twitter status that was retweeted by other Twitter users.
8. **retweeted_status_user_id**: This is the unique identifier for the retweeted status of a user.
9. **retweeted_status_timestamp**: This gives the time when a Twitter user's status was retweeted.
10. **expanded_urls**: This links to the exact tweet on the Twitter application.
11. **rating_numerator**: This is the number at the numerator value of the ratings given in the tweet (in this case, the text column)
12. **rating_denominator**: This is the number at the denominator of the ratings given
13. **name**: This refers to the name of the dogs.
14. **Doggo**: This refers to a dog with the following description "A big pupper, usually older."
15. **Floofer**: This refers to a dog with the following description "Any dog with seemingly excess fur."
16. **Pupper**: This refers to a dog with the following description "A small doggo, usually younger"
17. **Puppo**: This refers to a dog with the following description "A transitional phase between pupper and doggo."

The **image_predictions.tsv** file has the following columns:

1. **tweet_id**: This is the unique identifier for each tweet.
2. **jpg_url**: This is the link to each image for each tweet
3. **img_num**: This refers to the number of pictures posted by the owner of the tweet.
4. **p1**: This is the breed of the dog based on the first or best predictions of a machine learning algorithm.
5. **p1_conf**: This is the confidence level of the first or best predictions of the machine learning algorithm.
6. **p1_dog**: This is a column to indicate whether the image is a dog or not.
7. **p2**: This is the breed of the dog based on the second-best predictions of a machine learning algorithm.

8. **p2_conf**: This is the confidence level of the second-best predictions of the machine learning algorithm
9. **p2_dog**: This is a column to indicate whether the image is a dog or not.
10. **p3**: This is the breed of the dog based on the third-best predictions of a machine learning algorithm.
11. **p3_conf**: This is the confidence level of the third-best predictions of the machine learning algorithm
12. **p3_dog**: This is a column to indicate whether the image is a dog or not.

The **API Dataset** contains the following columns:

1. **tweet_id**: This is the unique identifier for each tweet.
2. **Num_likes**: This is the number of likes each dog that was posted accumulated.
3. **Num_retweets**: This is the number of retweets each dog that was posted accumulated.
4. **Display_text_range**: This is a list indicating the beginning and ending the number of characters in the tweet.

Data Gathering

The modules important for this data gathering step are:

- Pandas
- Numpy
- Tweepy (OAuthHandler)
- Json
- Requests

Twitter Archive Data:

Gathering the twitter_archive.csv file was just a matter of importing the dataset using the Pandas .read_csv() method since the data was already available.

Image Predictions Data:

Gathering the image_predictions.tsv file required passing in the Udacity url into the Requests module's .get() method to acquire a response from the Udacity server.

Using a context manager, the response content was written into a file called "image_predictions.tsv" using the write bytes ("wb") mode.

The file is then read using the Pandas .read_csv() method, but this time, with a "sep" argument which was set to "\t" to let Pandas know it is a tab-separated file.

Twitter API Data:

This data was gotten by using the consumer keys and access tokens given by the Twitter Development team.

The Tweepy module was used to get the authentication object from the Twitter API stored as tweepy.api.API

The .get_status() method of the API object was used to get the content of the tweets and they are written into a "tweet_json.txt" file using the context manager.

There was a failure to retrieve 689 tweets implying that only 1667 tweets are available through the API.

The "tweet_json.txt" file now created, is read using the context manager once more into a list called api_tweets. As a result, we get a list with a very large dictionary.

The dictionary is then read from the list and through a for loop, is used to create a Dataframe called "api_df"

Data Assessment

This involves performing some investigations on the datasets and noting points of imperfection. More specifically, the idea is to discover Data Quality issues as well as Data Tidiness issues.

Data Quality refers to the Completeness, Validity, Accuracy, and Consistency of the data within the dataset, while Data Tidiness refers to the presentation of the data, its structure, and overall clarity.

Twitter Archive Data:

The following steps were taken to assess the Twitter Archive dataset:

1. First, the tweet_id was checked for duplicates, for which there were none. The reason for this is to remove data redundancy from the dataset.
2. The timestamp column's datatype was checked because time can sometimes be represented as a string. The result of this was that the timestamp column was of datatype "Object" which is indeed a string in Pandas. This column needed to have a DateTime datatype.
3. Next, the expanded_url column is checked because the .info() method revealed that some of these URLs were missing. These URLs were cross-referenced with the api_df URLs by merging both datasets on the tweet_id column, in hopes that they might contain the missing URLs. However, this was not the case. As such, the expanded_urls column cannot be completely filled up.
4. The .value_counts() method was applied to the rating_numerator column to find out the dog ratings along with their corresponding number of occurrences. It was observed that there were

some really larger ratings and some small ratings. This prompted further investigation into these edge cases.

5. From the inspection done on the rating_numerator column, the following was derived:
   "From the above table, we observe that the reasons for really large ratings are:
   a. *The user was not talking about any dog in particular. These are the tweets that are replies to other tweets. these can be observed in the "in_reply_to_status_id, retweeted_status_id, retweeted_status_user_id, in_reply_to_user_id" columns.*
   b. *Some people have rated more than one dog and have multiplied both numerator and denominator by the number of dogs rated.*
   c. *Some rated their dogs with decimals as numerators. But only the tenths and hundredths values of the decimal have been recorded.*
   d. *Tweet with id "810984652412424192" did not tweet a rating. He/She only mentioned 24/7 which refers to day and time, not a rating.*
   e. *There are some tweets like the one in row 313 with seemingly double ratings. This should be investigated as the wrong value was recorded in the ratings columns."*

6. For the rating_denominator, while the majority of the data had a denominator of 10, out of the edge cases, far more were given a rating greater than 10, prompting further investigation.
   The following was gleaned from this investigation:
   "Majority of the large denominators are as a result of multiplying the numerator and denominator by the number of dogs Some of the large denominators are as a result of reading the wrong values"
7. During visual inspection of the rating_numerator and denominator columns of the dataset, it was observed some of the rating_numerator values were decimals and these were not properly captured in the rating column.
8. While inspecting the "name" column for the dog names, there were some non-dog names with a distinct characteristic of being lowercase names. In fact, these non-dog names were articles, definite articles, verbs, and so on.

Image Predictions Data:

1. The ratio of the number of dogs to non-dogs gotten from the "p1_dog" column which is Boolean, is taken with respect to the total number of tweets and the percentage of dogs in the dataset could be found.
   There were about 74% dogs in the entire dataset.
2. Since this dataset contains all the images, it was cross-referenced with the twitter_archive dataset using the .isin() method and this revealed that there were 281 tweets without images in the twitter_archive dataset.
3. Also, visual inspection revealed that some pictures were not dogs.

Twitter API Data:

1. The display_text_range is in form of a list and all we really need is the total number of characters in each tweet.

Data Assessment Summary

In all, the Data Assessment issues identified were:

Visual Assessment

Data Quality

- The columns in_reply_to_status_id, in_reply_to_user_id, retweeted_status_id, retweeted_status_user_id,retweeted_status_timestamp, have about 3 - 8% of their data available. This implies that removing them from the final dataset is important.
- The structure of the source column is not elegant.

Tidiness

- The 3 datasets have the tweet_id column in common. This means that we can combine the tables using this column.
- The dog stage is split into 4 different columns

Programmatic Assessment

Data Quality

- We must change the timestamp column's datatype from Object to datetime object
- Correct the wrongly inputted decimal ratings and other rating issues
- Remove all tweets that are not original dog ratings i.e. replies and retweets. These are all identified by the in_reply_to_status_id, in_reply_to_user_id, retweeted_status_id, retweeted_status_user_id,retweeted_status_timestamp columns
- Replace non-dog names with their real names where possible
- There are some pictures that are not dogs. They need to be removed from the final dataset
- There are 3 prediction confidence columns. We need just 1: the best prediction.
- The dog breeds are not uniform. Some are completely lowercase while others have their first letter capitalized.
- The display_text_range is a list. We only need the maximum number of characters of each tweet.
- Not all the column names have intuitive meaning
- The data type of the tweet_id is integer. We need it as a string

Tidiness

- There is no tidiness issue to report during the Programmatic Assessment

In all, we have 12 Data Quality issues, and 2 Data Tidiness issues to deal with

Data Cleaning

The following steps were taken to clean the dataset:

1. Copies of the 3 datasets gathered were made for record and reference purposes.
2. The 3 copies are then merged into a dataset called df_merged.
3. The first issue to be dealt with was the timestamp datatype issue. This was corrected by applying Pandas' .to_datetime() method which helped change the datatype from Object to Datetime.
4. Next, the display_text_range column was dealt with. Since its values are lists with just two items, the lambda function was applied to the column and only the second value indexed by a 1 is retained. This step also saw the change of the name "display_text_range" to the more intuitive "tweet_length".
5. The tweets that were not dogs were removed by assigning where "p1_dog" = True, to the original dataframe, df_merged. The other columns for the second and third best predictions were also removed because the first prediction was good enough.
6. For the Source column, the user's device, embedded in the anchor tag was extracted by applying the .str.split(">") chained methods to extract the user's device from the greater than and less than sign enclosing the user's device in the anchor html tag.
7. Next, a dog_stage column was created to hold the values in the puppo, doggo, floofer, and pupper columns since these were 4 columns talking about the same category.
8. The column names were changed into something more intuitive. The nomenclature is given below:
   "*timestamp ----> date_and_time*
   text ----> tweet
   expanded_urls ----> tweet_url
   name ----> dog_name
   jpg_url ----> dog_pic_url
   img_num ----> number_of_pics
   p1 ----> dog_breed
   p1_conf ----> dog_prediction_certainty"