The Wrangling Process

The “We Rate Dogs” twitter feed is silly feed that takes pictures of dogs and rates them out of 10 but typically the rating is over 10. My job was to gather data about their tweets with pictures of jpeg format and then clean up the data for further analysis.

Gathering the Data

The data was gathered from three different sources. The first source was general data on the tweets provided by Udacity as a csv. This data included the tweet id, the text, the source, the rating (as numerator and denominator), reply to and retweet data, and dogtionary terms (dogtionary is a set of silly terms created by this feed). I simply read the csv in a pandas dataframe.

The second source was a machine learning report that predicted the object in the tweet’s picture, specifically the breed of the dog. The data included the link to the jpg along with the top 3 predictions and their probabilities. I was able to programmatically download the tsv file and then read it in as a pandas dataframe.

The final source was the data directly from Twitter. Using Twitter’s API, I downloaded the json data for the tweets. I then stored them in a text files and read in the retweet and favorite counts for each of the tweets and put them in a dataframe. Unfortunately, this took me three attempts. First, I tried to get the data one tweet at a time, but I could not seem to get more than a thousand. Then, I tried batches of one hundred which downloaded the files, but I incorrectly assumed that they would be returned in the same order as the list I sent. Finally, I realized I needed to get the tweet id directly from the json file.

After all of the data was gathered, the dataframes were joined into a single dataframe and then saved as a csv.

Assessing and Cleaning the Data

The next step in the wrangling process is to assess and clean the data. I took a look at this programmatically, visually, and, to get more context, I looked at the actual tweet of individual items or a sample of items when I had questions.

The first thing that I noticed was that some there were columns for replies and retweets. The task was to only look at original tweets by “We Rate Dogs”, and replies and retweets are not original. I removed all records with data in these columns and then removed the columns themselves.

Next, I noticed that some of the retweet and favorite counts were missing. For the tweets that I had this data missing, I went back and checked that the tweets actually existed. In most cases they did (which is how I noticed that I had issues grabbing the tweet data as described above). So, I did a second pass to try to gather the data. After this pass, any tweet without the retweet and favorited data, I assumed was no longer in twitter and so I deleted it.

There were some tweets in which a jpeg was not listed. First, I went through the json files to look for that jpeg (programmatically of course). For tweets that still didn’t have this data, I looked at a sample on

twitter. These posts either had videos, a link to the image, or were quotes (and so were not original tweets). I deleted all of these.

Next there was cleaning up the data types. The tweet ID I changed into a string, as there would be no computation with these. I changed the retweets, favorites, numerator (which was a mistake as I'll explain later), denominator, and image number to integers. Finally, I found that the source column indicated where the tweet was posted from; I changed the data in this to indicate the source without the extra bits.

I then looked into the ratings. Through this analysis, I figured out that each dog in an image was rated out of ten and if there were multiple dogs the numerators and denominators were add (while mathematically incorrect, this did effectively give an average rating of the dogs). Unfortunately, some of these were incorrect (for instance 24/7) or not include. I ran a function that went through and found denominator multiples of ten then I added all of these along with the corresponding numerator (some of the numerators were decimals hence should have been floats). There were three that were not caught by this; I manually got the rating for two of these and deleted the third as it didn't have a rating. As I noticed that the mathematical mistake of adding numerators and denominators actually gave the mean, so I added a column for the rating as a decimal.

Then I looked at the predictions. I found the most likely dog breed, kept that breed and its probability and deleted the rest. I didn't delete tweets that could not identify a dog breed because one of the first images I looked at was a picture of a dog through the hole of a doughnut or a bagel so I assumed that the dog was missed by the machine learning algorithm.

Finally, I worked on the dogtionary terms in the tweet. First, I went to the dogtionary and I noticed that the term "floofer" was "floof", which I changed, and there were two more terms, "blep" and "snoot", which I added. I ran a non-case sensitive search on the tweets for any of the six dogtionary terms and added them to the appropriate column. Then created a dogtionary terms column which I put a list of all the terms if there were any there and a null if there were no terms.

The clean up now completed, I saved the resulting dataframe to a csv.