Gathering data from each source was generally straight forward, I think the biggest issue was registering my twitter account to be given access to the API where I ran into some issues having twitter confirming my phone number.

Once I had gathered the necessary datasets and importing them into their own dataframes I began my assessment. Having experience working with RDBMS led me to look at timestamps almost immediately upon opening the csv files. Changing the timestamps to their appropriate data type allowed for the later analysis and always serves as a good anchor point when querying data. I had set about parsing out the '+0000' nano second digits from each timestamp but realized it should work just fine having it there.

Removing retweets was even mentioned in the project description so I was on the lookout for that from the beginning. Standardizing denominators to all equal 10 seemed like a good way to legitimize the rating system while preserving the notion of "They're good dogs Brent".

Inspecting the data visually through Excel I came across several name entries where the name was written with some odd characters which eventually led me to a really useful stack overflow post on how to screen out non-ascii characters. That also led me to filter out name entries where there was random words or single letter characters. Lastly it made sense to only keep entries where the neural network was confident it was looking at a dog because it later allowed me to analyze and visualize the confidence level of the neural network at identifying dogs.

In terms of tidiness once rating denominator was cleaned to represent only values that are equal to 10, I had combined the numerator and denominator columns in to one 'rating' column but doing so meant having to convert the inputs to string format which made further analysis into ratings difficult so I chose not to merge the columns together in the final report.
Pulling the different dog types into a 'dog_stage' variable was one of the first things that jumped out at me when I looked for tidiness issues. After contemplating for a bit how to best approach doing this I came across a stack overflow post that mentioned extracting phrases from text and it seemed to be a perfect fit for this case. Additionally, consolidating all 3 data sources into one master file seemed almost necessary and luckily is a relatively straightforward process in pandas.

One of the more general difficulties I faced was deciding the exact order in how to approach the tidiness and quality issues. Initially I had written the code to address tidiness issues first i.e. combine columns and dataframes into one and then address the quality of the entire dataframe but near the end I backtracked and had to rewrite the code to address the quality issues of each dataframe first then lastly pull together each cleaned source into one master data frame ready for storage. I think this approach made the report significantly more accessible to the reader as it allowed the reader to track which data sources I was operating on and how they all tied in together to produce the final dataframe.