Data Wrangling Report

The process starts with the gathering of data of from various sources, which will then be cleaned and combined to create a master dataset that I can use to analyse and visualise the data. The first piece of data was the easiest to retrieve as it was provided beforehand as a simple download. All that was required was to read into my Juypter notebook as a dataframe, which I called df_twitter_archive.

The next dataset required the use of the requests module to programmatically download a TSV file that contained the data. I read this data into another dataframe called df_image_prediction and I made sure to specify that it was tabular data to ensure that Pandas was able to handle it correctly.

The third and final dataset was by far the most complex to obtain as it required getting apply to Twitter to get access to its API and using the Tweepy module to download the JSON data of every tweet listed in the df_twitter_archive dataframe into a TXT file. The entire process took about 30 minutes as Twitter has limits on the amount of data you can download at a time. Once all the tweets had been downloaded, I read them into a new dataframe line by line using Pandas. I name the dataframe df_tweet_addition.

Once all the data had been collected, it was time to assess what I had on my hands. The initial analysis showed that df_twitter_archive had 2,356 tweets, df_image_prediction had 2,075 and df_tweet_addition has 2,326 so I immediately knew that there was some missing data. Before I proceed any further, I made sure to narrow my focus to ensure that I only had original tweets that also have images. This means I could eliminate all retweets and replies as well as ignore tweets that do not have image URLs.

It was now time to assess the data using visual and programmatic methods which highlighted several issues with the data. Some of these included names and ratings being incorrectly extracted for tweets in df_twitter_archive; there were incorrect datatypes for some of the columns and number of tweets were not at all related to dogs. The data was also not tidy, with multiple columns used for the same variables instead of just one being. Once I identified all of the issues, I created copies of all the dataframes and proceed with the cleaning process, which involved multiple iterations of defining the cleaning action, coding and testing the result. I started by removing tweets from df_twitter_archive that could not be downloaded from the Twitter API, I also removed all the tweets that did not have an associated image URL. Retweets and replies were also removed as well as tweets that did not accurately represent dogs.

I then proceed to rename the relevant columns and variables that I found to be incorrect and combined the four columns for ‘doggo’, ‘floofer’, ‘pupper’ and puppo into one column called dog_stage as the data represented the same information. I did the something with the df_image_prediction columns for [p1, p2, p3], [p1_conf, p2_conf, p3_conf], [p1_dog, p2_dog, p3_dog]. The columns were narrowed down to ‘dog_type’ to show the breed of dog in the tweet and ‘confidence level’ to show how confident we are of the breed.

The final step was to combine the three dataframes into one master dataframe called twitter_archive_master and performing minor cleaning takes to make the data more presentable. This included renaming columns, capitalising some words and dropping unnecessary columns and ensured that all incorrect datatypes had been properly converted.

When I has satisfied that the data had been sufficiently cleansed, I stored twittter_archive_master into its own CSV file for further analysis.