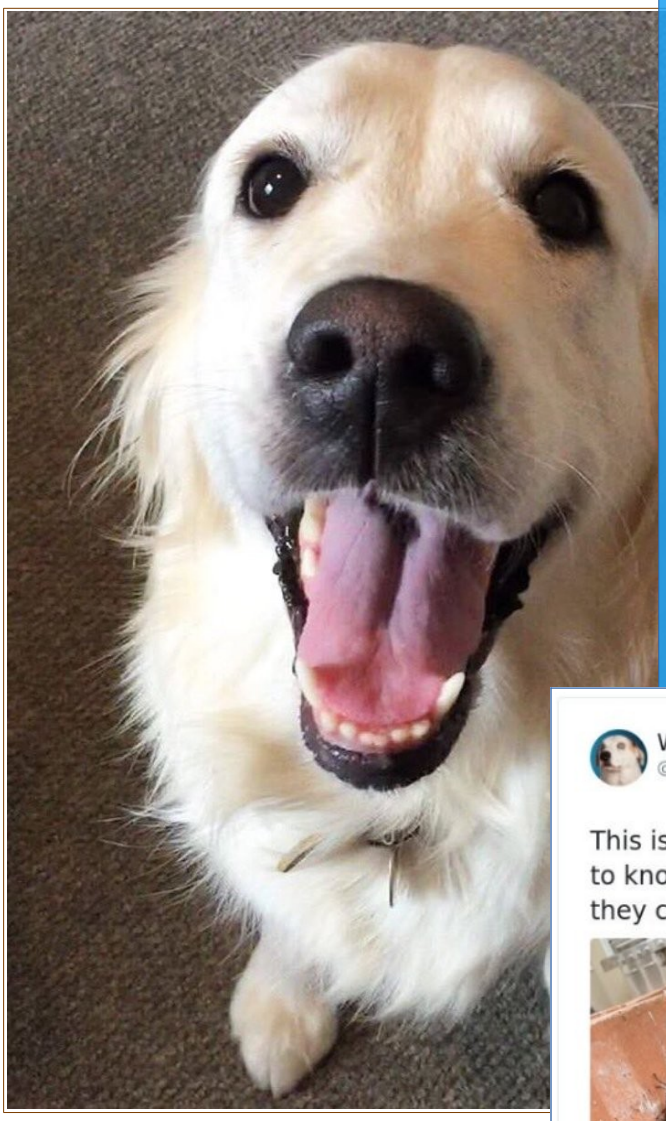
Data Wrangling Project

Udacity - FWD Initiative

By: Fatimah Ehab



Data Wrangling Report

By Fatimah Ehab
January, 2020

This report includes illustrations of the data wrangling efforts made for Udacity data wrangling project. The course is a part of Data Analyst Udacity nanodegree, provided by the Egyptian FWD initiative.

Summary
This project aimed to wrangle, visualize and analyze data for a dog-lover Twitter account *WeRateDogs* (*@dog_rates)*. Data was gathered from 3 different sources. Then got assessed and cleaned according to *define-code-test* approach. And lastly, the cleaned data was used to draw visualizations and perform analysis, in order to come up with some useful insights. One of these insights is that the account started with normal rating system, then they gradually adopted their special rating system, where the numerator is almost always higher than the denominator. And in the next few pages, a more detailed reporting will be documented.

I. Data Gathering
This was the first step, which is the base for the whole project. I collected data from three different main sources mentioned as following:

1. Twitter_archive_enhanced.csv file, which was available for manual downloading on Udacity course webpage. After downloading it, I imported it into a Pandas dataframe using pd.read_csv function. The file contained the account tweets main data with some missing attributes, such as retweets, favorites and followers.
2. Image_prediction.tsv file, which was hosted on a given site webpage. I downloaded it programmatically from the relevant URL using the Requests Python library. Then imported it into a Pandas dataframe using the same pd.read_csv function, with a particular parameter for TSV files. The file contained image predictions for the dogs breeds obtained by ML algorithm, that was conducted on most of the tweets in the archive file.
3. Tweet.json file, which was gathered from Twitter API using Tweepy library to query the API and obtain the rest of needed attributes that were missing from the archive file. I extracted only the retweets and favorites count data.

II. Data Assessment
During this second step, data is assessed to look for its quality and tidiness issues. I assessed the gathered data using two approaches. The first was visual assessment by displaying the data tables inside Jupyter notebook with Panadas and inside LibreOffice Calc. The second was programmatic assessment using Pandas functions, such as *df.info()* or *df.head()*. I listed and categorized the found issues into four sections, inside each section the issues sorted by the corresponding dataframe. This categorization was done by priority as following:

1. Tidy issues related to only one dataframe

- `archive`
  - The columns `doggo`, `floofer`, `pupper` and `puppo` are different value representations of the same variable `dog_stage`.
  - Add a gender table using the tweet `text` column.

2. Quality issues of completeness

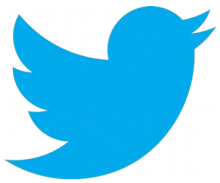
- `archive`
  - Missing values in tables (`in_reply_to_status_id`, `in_reply_to_user_id`, `retweeted_status_id`, `retweeted_status_user_id`, `retweeted_status_timestamp`).
  - Missing values stored as the string "None" in tables (`name`, `doggo`, `floofer`, `pupper`, `puppo`) that may be retrievable from the column `text`.
  - Columns (`in_reply_to_status_id`, `in_reply_to_user_id`, `retweeted_status_id`, `retweeted_status_user_id`, `retweeted_status_timestamp`) won't be needed after eliminating their whole records.
- `image_predictions`
  - There are missing values for image predictions, since there are originally 2355 tweets, while here 2074.

3. Tidy issues that involves multiple dataframes

- Merge all dataframes together.

4. The rest of quality issues

- `archive`
  - Vague unclear column names.
  - Some observations at `name` column are inaccurate.
  - Some observations at `rating_numerator` are inaccurate.
  - Invalid datatypes for `in_reply_to_status_id`, `in_reply_to_user_id`, `retweeted_status_id`, `retweeted_status_user_id`, `timestamp` and `retweeted_status_timestamp`.
- `image_predictions`
  - Vague unclear column names.
- `api`
  - Vague column name for `id`.



III. Data Cleaning

During the final step of wrangling data, I followed the best practice approach of Defining the issue, coding it, and testing the code. This was helpful as I went through the categorized issues. I defined what is needed to be done as instructions, started using Pandas functions to execute them, and finally tested every step to make sure each issue was successfully fixed. This table illustrates the issues fixed:

Issue Type	Dataset	Issue	Solution
Tidiness	Archive dataframe	The columns `doggo`, `floofer`, `pupper` and `puppo` are different value representations of the same variable `dog_stage`.	- Made a function to check whether dog stages are mutually exclusive or not. - Combined data stages into 1 attribute.
		Add a `gender` column using the tweet `text` column.	Created new gender column using a for loop and an empty list.
	Image dataframe	Convert predictions, their confidence and the breed, so that each represent 1 attribute.	Reshaped the dataframe using Pandas wide_to_long function.
Quality	Archive dataframe	Replace empty string values with NaN.	Replaced them using replace function.
		Convert meaningless names with highest frequency into NaN.	Replaced them using replace function.
	Image dataframe	Drop irrelevant rows for retweets and replies.	Dropped them using conditional slicing.
	API dataframe	Drop irrelevant rows for retweets, replies and tweets that have no images.	Dropped them using conditional slicing.
Tidiness	Master dataframe	Merge `archive` and `api` dataframes.	Done it using the Pandas merge function.
		Add the image url column from `image_prediction` dataframe.	Done it using the Pandas merge function, then dropping irrelevant columns.
Quality	Master dataframe	Rename vague column names.	Made it using rename function.
		Convert `tweet_timestamp` type into datetime object.	Made this using Pandas to_datetime function.
		Fix inaccurate `rating_numerator` and `rating_denominator` values.	- Inspected bizarre values. - Adjusted some occurrences manually. - Adjusted group ratings programmatically.
	Image dataframe	Rename vague column names.	Made it using rename function.

After finishing, I stored the cleaned data into two dataframes:
1. *Master_df*, which contained all cleaned tweets data and their attributes.
2. *Image_df*, which contained the cleaned table of the image breed prediction file.

Conclusion

Data Wrangling is the start point of any data analysis task. It makes the work much more easier. I found myself at situations when I had to go through the data wrangling steps over again. It’s an iterative process and has three iterative steps as well. This project data wrangling was challenging as a first time, and fun due to the subject nature.