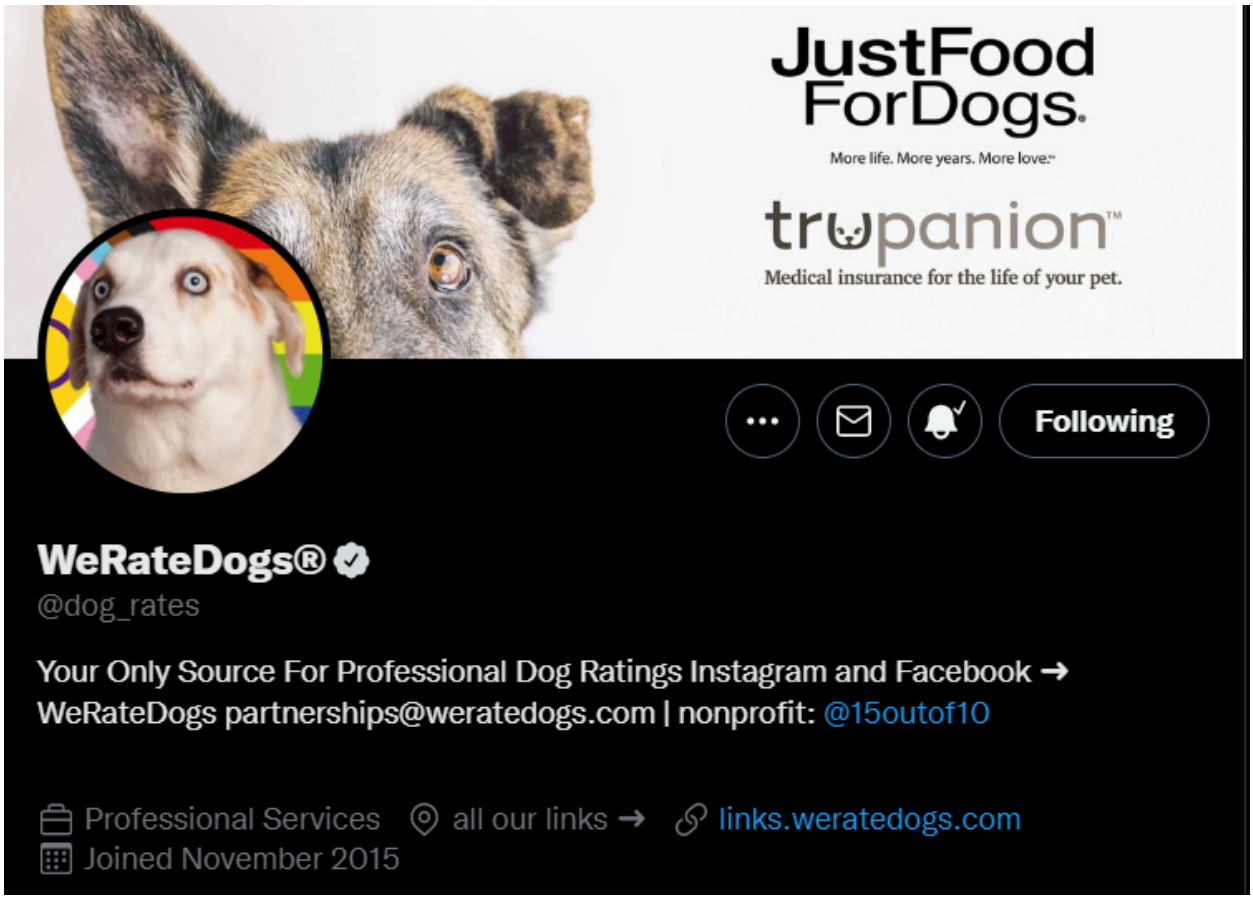
Data Analyst Nanodegree: Project Report

WRANGLING WERATEDOGS' TWITTER ARCHIVE

Tevin Aduma 29/06/22



Introduction

Udacity in collaboration with WeRateDogs (**later referred to as WRD in this document**) provided datasets and links to additional resources required for wrangling data, analyzing and visualizing generated insights.

This report details the process used to undertake my data wrangling efforts from discovery, structuring , cleaning , enriching and validating the data.

Step 1: Data Acquisition/Gathering

I created a custom function `open_set()` to read each piece of data whatever the format. Data was gathered from three separate sources:

1. WRD via Udacity provided a csv file of their entire Twitter Archive that was manually downloaded through the browser and read into the notebook using Pandas.
2. The second dataset was a file containing image predictions that, according to a neural network, is present in each tweet. This content in this file was obtained via a provided url with Python's `Request library` and written into a file named `image-predictions.tsv`
3. The final dataset was to be obtained using the Twitter API through Python's `Tweepy library` and then stored into a file named `tweet_json.txt`. I decided to extract the following metrics for each tweet: `favorite_count`, `retweet_count`, `geo_data` and `lang_data.`

The gathered data are loaded into three different DataFrames:

1. `df_tw_arch`: Loaded data from `twitter-archive-enhanced.csv`
2. `df_img_pred`: Loaded data from `image-predictions.tsv`
3. `df_tw_data`: Loaded data from `tweet_json.txt` I later saved this data as a csv file named `tw_data.csv` in order to visually assess the data using a spreadsheet program.

The aim of this data wrangling process is to obtain enriched data from all three datasets and merge them into one master dataset named `df_master`, that is subsequently copied from `df_copy_master`.

Step 2: Data Assessing

This study uses both visual and programmatic techniques to attempt to identify Quality and Tidiness issues across the datasets.

I used MS Excel as the spreadsheet program for visual assessment, other alternatives like Google Sheets, DB Browser etc are also applicable.

I employed Python Pandas and Numpy libraries along with Visual Studio Code IDE to programmatically comb through the data for any issues that the eye couldn't have caught at a glance.

I have created a table summarized issues in each dataframe:

- Techniques are represented as `V for Visual Assessment` and `P for Programmatic assessment`

```
df_tw_arch
```

Quality Issues

Method used	Dimension Affected	Columns affected	Issue Description
V	Completeness	`in_reply_to_status`, `in_reply_to_user_id`, `retweeted_status_user_id`, `retweeted_status_timestamp`	There are a few columns with glaringly empty fields
P	Consistency	`rating_numerator`	There are a few ratings that are not consistent across the board. Certain numbers are as high as 1700 while some are as low as 0.
P	Accuracy	`timestamp`	The datatype for this column should be a datetime object and not a string
P	Completeness and Integration	`name`	A few missing names in this column as well as names that would not make sense in the real life setting. For instance, a number of records have a value of "a" as a name.

P	Consistency and Conformance	`text`	There are a number of inconsistencies in terms of unwanted data for sentiment analysis e.g. hyperlinks, hashtags, Twitter handles etc that will have to be cleaned out

Tidiness Issues

V	Integration and Consistency	`doggo`, `puppo`, `pupper`,`floofer`	The four columns describing the "stage" the dog is in should be transposed into one column
P	Accuracy and Integration	`source`	The values in the columns are not accurately representative of the origin each tweet was sourced from.
P	Duplication	`expanded_url`	The `expanded_url` column is not required for this study as it just the full hyperlink to WRD's tweets
P	Conformance and Accuracy	`rating_numerator`	Based on WRD's rating convention, the mean of `rating_denominator` should not exceed 10.

```
df_tw_data
```

Quality Issues

Method used	Dimension Affected	Columns affected	Issue Description
V	Completeness and Quality	`geo_data`	The `geo_data` column seem to have just one unique entries. Generating any meaningful insights will seemingly prove futile from this column.
P	Completeness	`favorite_count`	There are a number of missing values as they were pulled from the Twitter API. These seem to have been retweets of WRD's own tweets causing a disparity to the original tweets' favorite metric. This is an issue on Twitter's end.

Tidiness Issues

P	Completeness	all the relevant columns	The main issue is this dataset SHOULD be concatenated into the main twitter archive `df_tw_arch` dataset to gauge how much interaction each dog's WRD tweet has.

```
df_image_pred
```

Quality Issues

Method used	Dimension Affected	Columns affected	Issue Description
V/P	Accuracy and Integration	`geo_data`	The `geo_data` column seems to have just one unique entry. Generating any meaningful insights will seemingly prove futile from this column.
P	Accuracy	`p1_dog`	Needs a better descriptive name
P	Consistency and Accuracy	`p1_dog`	Need to reformat strings so that they are more legible i.e. getting rid of the underscores

Tidiness Issues

P	Completeness	all the relevant columns: `p1`	The main issue is this dataset SHOULD be concatenated into the main twitter archive `df_tw_arch` dataset to gauge how much interaction each dog breed got in WRD tweets.
P	Accuracy	`p2`, `p2_conf`, `p2_dog`, `p3`, `p3_conf`, `p3_dog`	Unnecessary columns for this study.

Step 3: Cleaning

This is the stage where I employ a number of programmatic techniques to refine and shape the data into an enriched dataset that will be of value during Exploratory Data Analysis. The cleaning segue starts off with making intrinsic copies of all the dataframes which then involves **defining the issues** and how I will solve the issues, **writing code that will fix** the issues and, **finally asserting or testing** that the premeditated solutions have worked.

Bearing in mind that data wrangling is an iterative process, after I create the master dataset `df_copy_master ⇒ df_master` , I will attempt to clean any problems that arise in it as well.

Below is a table summarizing the cleaning process for each table where **the Issue Description is where I describe the problem**, **Solution is where I state what measures were taken to eradicate the issue**, and **Check is where I performed various assertion tests** to ensure the issues were taken care of and are no longer in the dataset.

Dataset	Issue Description	Solution(s)	Check(s)
df_tw_arch	Represent each value in the source column accurately	Use Pandas' mask() method to replace each value on certain conditions.	Assert the old hyperlink values have been replaced.
	timestamp datatype requires change	Use Pandas to.datetime_time() to alter the columns' datatype	Assert with Pandas' datetime datatype assertion test to ensure the change worked.
	Drop the fields with no insights and null values. The floofer has been included here due to its definition: any dog, really. It's	Use Pandas' drop() method with the inplace parameter set to True in order to return the dataframe with changes saved.	Run an assertion check to confirm those columns are no longer in the dataframe

	basically redundant in this entire study even if set to None in the columns.		
	Transpose the repetitive columns into one categorical column	Use Pandas apply() method chained with string.join() method to concatenate the three stages and represent each stage accurately using a custom function that applies Pandas' mask() method to replace the values in the new dog_stage column.	Assert all the old columns have been dropped and the dataframe has a new column containing my new naming convention.
df_tw_data	The anomalies in both rating_numerators and rating_denominators require feature engineering techniques.	For the rating_numerator, I propose to only use single or double digits in this column. Any values with more than two digits will be transformed using regex's findall() method Unfortunately, I decided to get rid of the rating_denominator rendering it useless due to massive differences across the board for different tweets.	Employ unittest's assertRegex function to ensure the field only contains 1 or at most 2 digits.

	Concatenate the dataset to the twitter-archive dataset due to its inability to stand alone as an observational unit. This is because this study is meant to be about WRD and the dogs, not just Twitter metrics	Pandas' merge () function to join both datasets on tweet_it on an outer join method i.e. a union of the two datasets	Assert the new dataframe is of the same size as the merge code snippet by using Pandas' assert_frame _equal method
	Geo_data provides no insight due to all null values.	Use Pandas' drop() method will be sufficient to get rid of this column	Assert the column is no longer in the list of the dataset's columns
df_image_pred	Getting rid of predictions that are not actually dog breeds	Retrieve a copy of the dataframe that has only True boolean values in the p1_dog column since it's the most reliable prediction value of all three.	Assert that the dataframe has no False values in the dog_breed column
	Drop the unnecessary columns	Use Pandas' drop method()	Assert the discussed columns are no longer in the dataframe
	Concatenate the dataset into the master dataset's copy	Use Pandas' merge on the tweet_id using the a union join method	Assert that the size of the new dataframe is equal to a merge of both dataframes using Pandas' assert_frame_equal
df_master	Rename the p1_column to a more descriptive name i.e. dog_breed	Use Pandas' rename() function to rename the column	Assert the column name change worked by confirming the old number has been gotten rid of while the new one is in the

			dataframe
	Format the strings in the dog_breed column to be more legible	Use a custom function with str.replace(), str.title() and str.capwords() and Pandas 'apply function for all values that are not null.	Assert the dog_breed column contains values that follow the proper naming techniques
	I will clean up the weird characters in the text columns.	I employ various regex patterns to get rid of hashtags, ratings, Twitter handles, numbers that are not descriptive as well as hyperlinks	Perform unnitest with assertNotRegex to ensure the various regex patterns are no longer found in the text

This is the final structure of the master dataset 👇🏾

	tweet_id	name	dog_stage	rating_numerator	text	source	timestamp	retweet_count	favorite_count	geo_data	lang_data	dog_breed
0	892420643555336193	Phineas	none	13	This is Phineas. He's a mystical boy. Only eve...	Twitter for iPhone	2017-08-01 16:23:56	7007.0	33809.0	None	en	NaN
1	892177421306343426	Tilly	none	13	This is Tilly. She's just checking pup on you....	Twitter for iPhone	2017-08-01 00:17:27	5301.0	29329.0	None	en	Chihuahua
2	891815181378084864	Archie	none	12	This is Archie. He is a rare Norwegian Pouncin...	Twitter for iPhone	2017-07-31 00:18:03	3480.0	22048.0	None	en	Chihuahua
3	891689557279858688	Darla	none	13	This is Darla. She commenced a snooze mid meal...	Twitter for iPhone	2017-07-30 15:58:51	7226.0	36938.0	None	en	NaN
4	891327558926688256	Franklin	none	12	This is Franklin. He would like you to stop ca...	Twitter for iPhone	2017-07-29 16:00:24	7759.0	35310.0	None	en	Basset

Limitations

- There were a number of deleted tweets that didn't send back any favorite_count metrics. It would have been convenient to obtain these metrics for efficient EDA.
- The values in the name column make it quite hard to perform any analysis on the impact on dog names to both WRD's rating awardance and audience sentiment.
- The anomalies in WRD's rating convent make it hard to determine if there is a standard rating system and therefore subsequent analysis will be made to get a deeper understanding into this.
- Certain retweets were not part of WRD's tweets and seemed to get jumbled up in the data archive therefore skewing numbers in the retweet_count column.
- The currency of the data limited its relevance in the current world and whether the data in this study still holds water in relation to audience preference at the moment.

Resources

1. Data Wrangling in 6 Steps: An Analyst's Guide For Creating Useful Data (hevodata.com)
2. https://stackoverflow.com/questions/28596493/asserting-columns-data-type-in-pandas
3. https://pandas.pydata.org/docs/reference/api/pandas.api.types.is_datetime64_ns_dtype.html

For further information about this study, here's a link to the Github repository