Introduction

1 Gather

2 Assess

3 Clean

4 Insights

Wrangling Efforts

Byron Kilian

Introduction

This report is grouped into the four categories that form part of the data wrangling process

> 1. GATHER
> 2. ASSESS
> 3. CLEAN
> 4. INSIGHTS

In each of these sections I will briefly describe what I did in order to accomplish a clean dataset.

1 Gather

During this step of the process I look at sourcing the following three data requirements

> 1.1 twitter-archive-enhanced.csv
> 1.2 image-predictions.tsv
> 1.3 tweet_json.txt

1.1 twitter-archive-enhanced.csv

This file was provided and I could manually download it and place it within the working directory.

1.2 image-predictions.tsv

This file I automatically download using the Pyhton requests library. I used the request objects content and wrote that to a tab seperating file called ‘image-predictions.tsv’

1.3 tweet_json.txt

In order to create this file I had to create an application Twitter account, obtain all the security tokens and read over the Twitter API documentation. The library I used to gather this information and gain access to Twitters API is the Tweepy library. The API method I used is called ‘get_status’

The response from this method is a JSON structure that I passed to the JSON dumps method which then gets written to a text file called ‘tweet_json.txt’ Within the notebook called ‘wrangle_act’ I display an image to show what the JSON structure looks like.

2 Assess

Before I could assess the data I first had to convert the file content into objects I could work with. To read in the csv file I used the pandas python library, to read in the tsv file I used the the pandas python library, but with delimiter = t (tab). Finally to read in the twitter data I looped over every JSON line item within the .txt file and parsed the JSON using the JSON loads method. I build up an array of dictionary data that I then passed to the data frame constructor to create a data frame.

To make the analysis process easier and to store the data neatly I wrote the dataframes to SQL using the following libraries: sqlalchemy, sqlite3 and pandas. The database is called ‘WrangleAndAnalyzeProjectDB.db’ and it has been provided as part of the submission.

Now that the data has been gathered and stored, the next step in the wrangling process is to assess the data. During this process I will look at two catergories:

> 2.1 Messy Data (Structure issues)
> 2.2 Dirty Data (Quality issues)

2.1 Messy Data

During this investigation I want to accomplish a tidy dataset. This is a dataset with the following characteristics:

> Each variable is a column
> Each row is an observation
> Each type of observational unit forms a table

2.2 Dirty Data

During this investigation I will look at the following criteria:

> Datatypes
> Duplication
> Completeness
> Accuracy
> Validity
> Consistency

Datatypes:

I had to fix some of the datatypes within the datasets. In the define section below I indicate those that had to be fixed.

Duplication:

There was no duplication problems within the datasets.

Completeness:

There was only some completeness issues within the archive set, but the missing values was on variables that I did not use within my analysis and since it did not have any impact I kept the missing values as is.

Accuracy:

During my analysis in SQL I noticed that some observations contain a rating denominator > 10. In some instances the rating was just stripped incorrectly due to other numbers in the text variable that has the same pattern ‘x/y’. These entries are valid, but has to be fixed to the correct rating. I also noted that the source variable contained tag information that had to be parsed in order to get and store the proper url value.

The dog name was also not consistanly extracted correctly, since the text variable did not always contain the pattern ‘…this is…’. The expanded urls variable contained a long list like structure, for this variable I used the split method and only used the first url value to update the variable.

Validity:

During my analysis of the data in SQL, I noticed that some of the text string contains phrases that indicate that the tweets are not valid, since they do not relate to dogs. These phrases included:

> ….we only rate dogs….
> …we usually don’t… (this is subjective but I still think its invalid)
> …we normally don’t… (this is subjective but I still think its invalid)
> …we don’t rate…
> …please don’t…
> ….only send..

I’ve also observed where one tweet relates to many dogs. It’s true that these observations do relate to dogs, but its invalid, since each observation has to present the demographics of one dog. This caused the ratings for these observations to be very high (factor of 10) and it also contained multiple dog stages.

Consistency:

I also noticed some inconsistencies in the prediction outcome of the algorithm. I did not want to change them as this is the outcome of a model. I would rather fix inconsistency issues caused by human error during manual capturing.

3 Clean

The next step was to create copies of the original dataframes on which to perform the cleaning steps using the pandas method ‘copy’. In order to perform the cleaning activity I devided it into the three stages:

> Define
> Code
> Test

Define

> TABLE twitterarchive

The variables: doggo, floofer, pupper and puppo can be converted into one new catergorical variable called ‘dogstage’ with 4 categories using the python melt method.

> TABLE imageprediction

The prediction variables can be converted as follow:

p1, p2 and p3 can be converted into one variable - prediction (string variable).
p1_conf, p2_conf and p3_conf can be converted into one variable - confidencerating (float variable). p1_dog, p2_dog and p3_dog can be converted into one variable - clasification (boolean variable indicating if the breed is of type dog or not). Using the python melt method.

> TABLE twitterarchive
> DataTypes:
> tweet_id convert to string
> in_reply_to_status_id convert to string
> in_reply_to_user_id convert to string
> timestamp convert to datetime
> retweeted_status_id convert to string
> retweeted_status_user_id convert to string
> retweeted_status_timestamp convert to datetime

> TABLE imageprediction
> DataTypes:
> tweet_id convert to string
> p1_dog convert to boolean
> p2_dog convert to boolean
> p3_dog convert to boolean

> TABLE twitterapidata
> DataTypes:
> create_at convert to datetime
> tweet_id convert to string

> TABLE twitterarchive
> source : strip the information for the http url instead of the tagged string
> some observations has more than one dogstage specified (most often it's multiple dogs on one image presented as one observation (tweet))
> some observations has incorrect ratings
> expanded_urls contains a list of similar urls, only one is required

> TABLE twitterarchive
> text: Multiple dogs in an image form part of one observation - remove
> text : Observations does not relate to dogs - remove

Code

The code used for the cleaning of the data is all done within python and captured within the notebook called 'wrangle_act'

Test

The testing has also been completed in python within the notebook called 'wrangle_act. I mostly made use of the python assert statement to validate if the changes took affect,

4 Insights

After I completed all the cleaning and testing, I wrote the clean dataframes to SQL. In SQL I created a view (the code provided in file 'Create_View_Script.sql'). called FinalDataset. The records within this view I wrote to a csv file using the python pandas library.

The final csv file called 'twitter_archive_master.csv' was produced and I imported this into my R Markdown document for analysis and plots. Please see file 'act_report.html' for a report on the data and insights.