Udacity Data Analyst Nanodegree

Wrangle and Analyze Data

Project Overview

The project is about the complete end to end data wrangling process, as:

- Gather Data
- Asses Data
- Clean Data

The tool which is used is called Jupyter Notebook.

The exercise is to clean and to harmonize three data sources, being able to do analysis. The is sourced from an archive of Twitter user @dog_rates, also known as WeRateDogs. WeRateDogs is a Twitter account that rates people's dogs with a humorous comment about the dog.

Data Source Overview

As mentioned already 3 data sources need to be analyzed.

- twitter_archive_enhanced.csv: A twitter archive extract, which can be download locally
- image_predictions.tsv: The result of a neural network predicting the breed of a dog based on images. Can be downloaded from the URL.
- Twitter API: Last but not least query the Twitter API for each tweet's JSON data and store the data in a file called tweet_json.txt.

Wrangling Activities

Gather Data

Data Source twitter_archive_enhanced.csv

I downloaded manually the file twitter_archive_enhanced.csv and stored it a data frame called twitter_archive.

Data Source image_predictions.tsv

I downloaded programmatically the file image_predictions.tsv and stored it in a data frame called image_predictions.

Data Source Twitter API

I called based on the already gathered twitter ids from twitter_archive, the twitter API and gathered 2 more attributes (retweet_count and favourite_count) and stored them together with the tweet_id in a file called tweet_json.txt. After that I loaded the file tweet_json.txt and stored the data in a data frame called twitter_api.

Asses Data

I assessed the data primarily with the following commands and technques:

- .info()
- .head()
- .value_counts()
- .sample()
- .describe()
- .query()
- Boolean Indexing

After a first round I put together the structure of the three datasets.

twitter_archive

Field	Decription
tweet_id	the unique identifier of tweets
in_reply_to_status_id	if the represented Tweet is a reply, this field will contain the integer representation of the original Tweet's ID
in_reply_to_user_id	if the represented Tweet is a reply, this field will contain the integer representation of the original Tweet's author ID
timestamp	timestamp creation of the Tweet
source	device from where the Tweet was posted, as an HTML-formatted string. e.g. Twitter for Android, Twitter for iPhone, Twitter Web Client
text	actual text of the status update
retweeted_status_id	if the represented Tweet is a retweet, this field will contain the integer representation of the original Tweet's ID
retweeted_status_user_id	if the represented Tweet is a retweet, this field will contain the integer representation of the original Tweet's author ID
retweeted_status_timestamp	timestamp of retweet
expanded_urls	tweet URL
rating_numerator	numerator of the rating of a dog
rating_denominator	denominator of the rating of a dog
name	name of the dog
doggo	respective dog stage
floofer	respective dog stage
pupper	respective dog stage
puppo	respective dog stage

twitter_api

Field	Decription
id	the unique identifier of tweets
retweet_count	number of times this Tweet has been retweeted
favorite_count	indicates approximately how many times this Tweet has been liked by Twitter users

image_predictions

Field	Decription
tweet_id	the unique identifier of tweets
jpg_url	dog's image URL
img_num	the image number that corresponded to the most confident prediction (numbered 1 to 4 since tweets can have up to four images)
p1	algorithm's #1 prediction for the image in the tweet
p1_conf	how confident the algorithm is in its #1 prediction
p1_dog	whether or not the #1 prediction is a breed of dog
p2	algorithm's #2 prediction for the image in the tweet
p2_conf	how confident the algorithm is in its #2 prediction
p2_dog	whether or not the #2 prediction is a breed of dog
p3	algorithm's #3 prediction for the image in the tweet
p3_conf	how confident the algorithm is in its #3 prediction
p3_dog	whether or not the #3 prediction is a breed of dog

Figure 1 overview 3 datasets

After that I consistently examined each dataset and came up with the following issues separated by quality issues and tidiness issues.

- Quality: issues with content. Low quality data is also known as dirty data.
- Tidiness: issues with structure that prevent easy analysis. Untidy data is also known as messy data. Tidy data requirements:
    - Each variable forms a column.
    - Each observation forms a row.
    - Each type of observational unit forms a table.

Quality Issues

twitter_archive data frame	
1 No Retweets	In the project description it is mentioned that only original tweets are relevant, so we need to take care on those "duplicates". Technically each tweet gets a new id referencing to the original one.
2 API errors	the twitter api access was returning 19 times "No status found with that ID.". Assumption those tweets have being purged.
3 Image predictions aren't matching twitter archive	The number of image_predictions and twitter_archive isn't matching.
4 Unnecessary Columns	After purging the retweets, the 3 columns retweeted_status_id are obsolete
5 Timestamp data type	Timestamp wrong datatype (string instead of date type)
6 Reduce to source	Source surrounding html can be dropped so that we have e.g. "Twitter for iPhone" left.
7 Name column cleaning	Name we can replace some of those 68 with NaN as they are not existent (e.g. "a")
8 Tweet ID data type	Tweet_id is integer can be converted to string
twitter_api data frame	
9 Tweet ID data type	Tweet_id is integer can be converted to string
image_predictions data frame	
10 New breed field	New breed field and get the p1 prediction into a separate field and capitalize the first letter
11 Tweet ID data type	Tweet_id is integer can be converted to string
12 Get rid of p1 and p2	Unnecessary fields, the 2nd and 3rd results of predictions can be dropped

Tidiness Issues

common across all three tables	
1 Merger	All data frames seem to be in 1 relationship to each other, hence they can be merged
twitter_archive data frame	
2 Fragmented Field:	Doggo, floofer, pupper and puppo are dog stages and can merged into a categorial field

Clean Data

After detecting all issues, I went to the data cleaning. I choose the following sequence:

- 1st Missing Data (Quality Issues)
- 2nd Tidiness issues
- 3rd Remaining Quality issues

During the cleaning activities I used the following command and techniques:

- I followed the approach of Define, Code and Test
- I used the following methods:
  merge() , extract() , drop() , isnan() , astype() , to_datetime() , islower() , replace() , rename(), loc[] , value_counts() , info() , head() , loops, regular expressions and functions
- I also used e.g. frameworks like “BeautifulSoup”

As mentioned already first issues related to missing or inconsistent data have being addressed, e.g. the API calls resulted in 19 errors (Tweets already deleted), hence the data need to be dropped from the archive data. Then the I handled the tidiness issues to get a well-structured and normalized dataset, e.g. I merged the 3 datasets as they have been in a 1:1 relationship, the leading key was always the tweet_id. After that I addressed the rest of the quality issues, like names which are wrong, unnecessary columns etc. Finally, I saved the merged and cleaned dataset to a file.

This resulted in the following volume metrics of the final dataset:

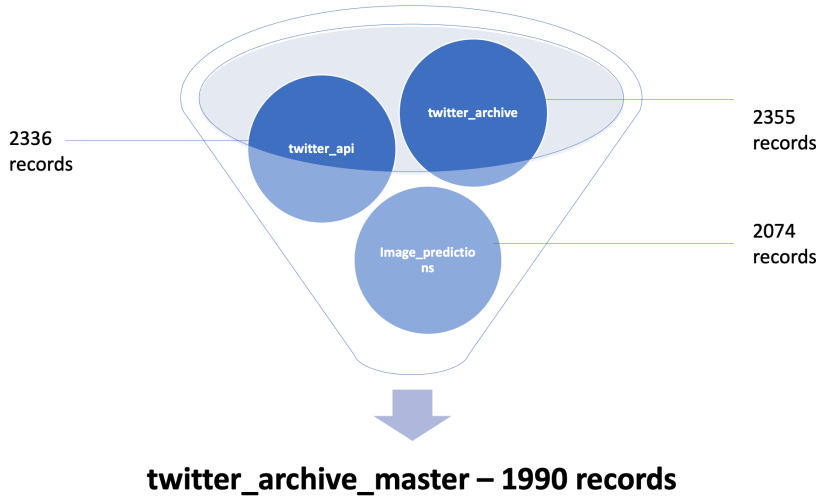

Figure 2 Volume Metrics