Wrangling Data Report

Scott Haluck

Gather

Manually download and read CSV file

To begin the project, a CSV file of Twitter info for the “We Rate Dogs” was provided. The file was manually downloaded and placed in the working directory. Then, the file was imported into a pandas dataframe (**`tweets`**) using the **`pandas.read_csv`** function.

Programmatically download and read TSV file

Another file of data gathered by machine-learning analysis of the “We Rate Dogs” photos was stored on a website. With the provided web address, the TSV file was downloaded with the **`requests`** library and stored in the working directory. The TSV file was then imported into a pandas dataframe (**`breed_predictions`**) using the **`pandas.read_csv`** function with the separator value as tab.

Use API to gather Twitter data, store in JSON, read JSON

The final pieces of the dataset were gathered using the Twitter API. Using the **`tweet_id`** values from the **`tweets`** dataframe, the status of each individual tweet was gathered and stored in JSON format in a TXT file. Due to rate limits on the Twitter API and volume of the dataset, the **`wait_on_rate_limit`** and **`wait_on_rate_limit_notify`** were set to True, which allowed the tweet data to be gathered autonomously.

With all of the available data gathered and stored in a TXT file, each line of JSON data was read into a Python list. The Python list of JSON data was then converted into a pandas dataframe (**`tweet_data`**) using the pandas.DataFrame constructor. The dataframe was then sliced to provide the specific data of interest.

Assess

Each dataframe was assessed for completeness, validity, accuracy, consistency, and tidiness (tidy data). The **`info`**, **`columns`**, **`shape`**, **`head`**, **`dtypes`**, **`describe`**, **`value_counts`** functions were used to analyze the dataframes or series within those dataframes. Boolean masking, query, and lambda functions were used to further investigate potential issues.

The following general issues were identified:

- Missing data (columns with null values)
- Invalid data (incorrect names, values that do not comply with schema, replies or retweets, irrelevant predictions)
- Inaccurate data types (data stored as incorrect type)
- Inconsistent or inefficient formatting of strings
- Inconsistent column headers
- Values that need to be further extracted or simplified
- Data spread over three dataframes

Clean

To begin, a copy was made of each of the dataframes to prevent data loss during the cleaning process. Then, each dataframe was fixed to allow for tidy data and smooth comparisons. This consisted of:

- Consistently naming the tweet_id column between dataframes
- Separating complex data into individual columns for each piece of data
- Combining all of the information from three tables into one table

After all data was merged into one table, the individual quality issues were repaired to allow for accurate and inefficient analysis. This consisted of:

- Fixing datatype errors (strings for identification data, datetime for time data, boolean data when appropriate)
- Removing inaccurate names
- Removing HTML markup from strings for source
- Removing retweets or replies (does not fit schema for original tweets only)
- Reformatting url for each tweet
- Drop a tweet with incorrect score when no score is present
- Correctly retranscribe three mistakes in ratings
- Simplifying predictions made by machine learning algorithm
- Calculate average ratings for better comparison when different numbers of dogs are present

After the entire data cleaning process, the data was stored in a CSV file using the **`pandas.to_csv`** function.