@WerateDogs – Data Wrangling Project

CONTENTS

1.0 Introduction

Data Wrangling is one of the key steps in Data Analysis, as it takes 80% or more part of Data Analyst. Real world data is often dirty and unstructured which make data analysis harder. Fortunately, latest software advancements like Python, and libraries like Pandas, Numpy, etc., makes data analyst's life easier for making the data wrangling process faster, smoother.

At a high level, the data wrangling comes in 3 different steps, as mentioned below:

- Gather
- Assess
- Clean

Lets dive deeper into each of the steps for the **@WeRateDogs** data to get meaningful insights.

2.0 Data Gathering

First and most important step in data wrangling is '**data collection**' or '**data gathering**'. Here we look at the initial data provided and check if the initial data has any missing values.

For this project, initial we have provided with archived twitter data (*twitter-archive-enhanced.csv*), which has about 2356 tweets. But this initial file, is missing some key fields like 'retweet count', 'favorite count', 'image_urls', etc.,. These information can be gathered using **tweepy** python library. '**statuses_lookup()**' API has been used to retrieve the tweet details in bulk. After querying for the tweet details, all the results are stored in '**tweet_json.txt**'.

Additionally, Udacity have provided prediction results for some of the dog images posted in the tweets, and the results are available in here. '**requests**' python library have been used to download this file from the webserver. The fetched file is stored in '**image-predictions.tsv**'.

3.0 Data Assessment

After gathering all the data, data need to be assessed to identify **dirty and messy data** issues.

Dirty data relates to data issues like missing, duplicate, invalid.

Messy data relates to data issues like structural issues. Any data which does not follow the below rules, it means it has structural issues:

1. Each variable forms a column.
2. Each observation forms a row.
3. Each type of observational unit forms a table.

Assessments can be done in two ways:

1. Visual Assessment
2. Programmatic assessment.

Visual Assessments are done using text editors like Excel, Visual Code. I have used both of them for identifying the issues.

Programmatic Assessments are done using pandas in-built functions like info(), describe(), etc.,

3.1 Twitter Archive Data Assessment

Below are the list of issues identified in twitter archive data:

Dirty Data Issues:

- rating_denominator - About 20 records have rating Denominator greater than 10. As per Wiki, the rating scale is one to ten. This is possible, multiple dogs can be rated in one tweet.
- rating_numerator - About 24 records have Rating Numerator greater than 20. This is unusual.
- name - Some dog names have come up as 'a', 'an', 'this', 'the'.
- dog stage - Not all tweets have dog stage names
- timestamp - Tweet Created Time is not in datetime type
- Missing Values - retweet count and favorite count values are missing for each tweet.

Messy Data Issues:

- dog stage - 'puppo', 'doggo', 'floffer', 'pupper' - these are different dog stages. In other words, these are values. These needs to be tracked under one variable 'dog_stage'

3.2 Twitter Details Data Assessment

Dirty Data Issues:

- Missing Values - Original we queried twitter for 2356 Tweets, but we have only 2342 Tweet Details. We are missing about 14 Tweet Details.
- contributors, coordinates,geo, in_reply_to_screen_name,in_reply_to_status_id, in_reply_to_status_id_str,in_reply_to_user_id, in_reply_to_user_id_str, is_quote_status, possibly_sensitive, quoted_status, quoted_status_id, quoted_status_id_str, quoted_status_permalink, truncated,user, retweeted_status - Remove these columns as we are not planning to use these columns.
- id, id_str - These are duplicate columns.'id_str' column value does not match with all the tweet ids in twitter archive data. So, we need to remove 'id_str' column

Tidy Data Issues:

- extended_entities - This column contains the data in JSON format. This need to be parsed. Also, a tweet may contain upto 4 images. Each Image is an observation and need to be a row in the dataset.
- entities - Need to parse this column to get the hashtags from each tweet.

3.3 Twitter Image Prediction Data Assessment

Dirty Data Issues:

- Missing Data - We have 2356 tweets in twitter archive data, but image prediction is available only for 2075 tweets.

Tidy Data Issues

- p1, p2, p3,p1_conf, p2_conf, p3_conf, p1_dog, p2_dog, p3_dog - These are just column names. Ideally, they should have been tracked in 4 variables (Prediction Number, Breed Prediction, Prediction Confidence Score, Is Prediction a dog?)
- Finally, We can reduce the three data-frames into 2 data frames. One for tweet details and one for image predictions

4.0 Cleaning

Once we have documented all the issues we have identified, cleaning can be performed. Cleaning should be programmatic as much as possible.

Cleaning involves 3 steps:

- Define – Pseudo code to fix the issue.
- Code – Actual code to fix the issue.
- Test – Check, if the issue has been fixed.

Also, before proceeding to clean up process, make sure, we can a copy of all original dataframes, and perform clean-up operations on the copied data. So, that, we can always go back to our original data if needed.

4.1 Twitter Archive Clean Up - Definitions

- Drop ununsed columns ('in_reply_to_status_id', 'in_reply_to_user_id', 'retweeted_status_id', 'retweeted_status_user_id', 'retweeted_status_timestamp').
- Using Pandas Melt function, transform the columns ('puppo', 'doggo', 'floffer', 'pupper') into one single variable named 'dog stage', and remove the following columns ('puppo', 'doggo', 'floffer', 'pupper') once transformed.
- Some of the dog stage values have 'None'. Need to replace 'None' with np.NaN.
- Some of the dog names have come as 'a', 'an', 'this', 'the'. Use regular expression to identify the name. If the name is not provided (some of the dog names are just breed names not real names), then return np.NaN. Also, it seems some of the dog names follow this convention 'named <value>'
- Convert 'timestamp's object type to datetime type.
- Identify all the rating pattern matches in the tweet text. If the denominator is greater than 10, check if its a multiple of ten, as rating can be given to pack of gods. If its a multiple of 10, then normalize the numerator & denominator values i.e (numerator * 10) / denominator. The resulting value can be used as numerator and denomintor can be set to 10. If the denominator is

not a multiple of 10, then its highly likely rating is not provided. For these cases, we will use np.NaN.

- Some of the rating numerators are greater than 20. ie. We have ratings like 182/10, 1776/10. Let's assign value 20.0 for these ratings, so that our analysis does not skew.

4.2 Twitter Details Clean Up - Definitions

- Drop un-necessary columns 'contributors', 'coordinates', 'created_at', 'display_text_range', 'geo', id', 'in_reply_to_screen_name', 'in_reply_to_status_id', 'in_reply_to_status_id_str', 'in_reply_to_user_id', 'in_reply_to_user_id_str', 'is_quote_status','place', 'possibly_sensitive', 'quoted_status','quoted_status_id', 'quoted_status_id_str', 'quoted_status_permalink', 'retweeted_status','truncated', 'user', 'source'
- Rename 'id_str' column to 'tweet_id'
- Expand entities object, and extract hashtags & user_mentions.
- Expand extended_entities object and extract media information like Image URLs, Media Type. Store all the expanded extended entities in a dataframe and store them in a list. Concat all the dataframes into one dataframe. Also, remove unnecessary fields in extended_entities object like 'display_url', 'media_url', 'url', 'source_status_id', 'source_status_id_str', 'source_user_id', 'source_user_id_str', 'additional_media_info','video_info'. Rename 'id_str' in media object to 'media_id'. Merge the expanded dataframe with original twitter details dataframe.
- Merge the Twitter API Details with Twitter Archive Information.
- Extract Source Information from HTML tags.

4.3 Twitter Image Prediction Clean Up - Definitions

- There are 3 Prediction Results in the dataframe. Select each prediction data in a separate dataframe, and rename the columns appropriately. i.e. p1 -> 'prediction_breed', p1_conf -> 'confidence_score', p1_dog -> 'is_dog' and so on.
- Include a new variable called 'prediction_number' to identify whether its p1/p2/p3 result.
- Combine each of the prediction group results into one single dataframe

5.0 Storing the Cleaned Data

It's always a best practice to store the saved data into SQL Database, or in file formats like (CSV, JSON, etc.,)

In this project, all the cleaned data have stored in 'twitter_archive_master.sqlite' SQLite database.