UDACITY

أكاديمية مسك
MISK ACADEMY

# Project #5

**Ghadi Alowaimer**

Data Analyst

Wrangle and Analyze Data

“Wrangle Report”

# Data Wrangling consist of three steps :

## 1-Gather : Firstly, I gathered three datasets from different sources, here is description for each

**1-The WeRateDogs twitter archive data :** "Existing file" the WeRateDogs Twitter archive contains basic tweet data for all 5000+ of their tweets, but not everything. One column the archive does contain though: each tweet's text, which they used to extract rating, dog name, and dog "stage" (i.e. doggo, floofer, pupper, and puppo) to make this Twitter archive "enhanced." Of the 5000+ tweets, they have filtered for tweets with ratings only (there are 2356)twitter_archive_enhanced.csv.

**2-The tweet image predictions data :** "Downable file" i.e., what breed of dog (or other object, animal, etc.) is present in each tweet according to a neural network. This file (image_predictions.tsv) is hosted on Udacity's servers and should be downloaded programmatically using the Requests library and the following URL: https://d17h27t6h515a5.cloudfront.net/topher/2017/August/599fd2ad_image-predictions/image-predictions.tsv.

**3-The twitter API data :** Unfortunately, twitter didn't give me the access to get the data as developer account so, instead I will use the tweet-json file that Udacity provided it. "JSON API file" Each tweet's retweet count and favorite ("like") count at minimum, and any additional data I find interesting. Using the tweet IDs in the WeRateDogs Twitter archive, query the Twitter API for each tweet's JSON data using Python's Tweepy library and store each tweet's entire set of JSON data in a file called tweet_json.txt file. Each tweet's JSON data should be written to its own line. Then read this .txt file line by line into a pandas DataFrame with (at minimum) tweet ID, retweet count, and favorite count.

## 2-Assess : I did both visual assessment and programmatic assessment and I found problems related to quality

and tidiness.

For clarification:

**Quality**

Quality: issues with content. Low quality data is also known as dirty data.

Data Quality Dimensions: Completeness, Validity, Accuracy, Consistency.

*Twitter_archive_enhanced_data:*

-Timestamp column is type object, so I want it to change it to datetime type because it more appropriate

-tweet_id column is int type it should be string (object) type because I will not do any calculation or manipulation

-Replace None value to nan to be more professional

-name column has wrong names that start with lower case and none values such as {'getting', 'actually', 'such', 'the', 'infuriating', 'by', 'incredibly', 'one', 'quite', 'a', 'nan', 'very', 'my', 'not', 'his', 'officially', 'this', 'all', 'old', 'just', 'unacceptable', 'an', 'light', 'space', 'O', 'None'} replace them with nan

*Image_predictions_data:*

-tweet_id column is int type it should be string (object) type because I will not do any calculation or manipulation

-Rename p1, p2 and p3 columns for more clarification

-p1, p2 and p3 columns values replace underscore with space to be clear and with lower values

-Duplicate jpg_url should be remove

*Tweet_json_data:*

-Rename id column to tweet_id in the Tweet_json_data to be consistent with other data table so I can merge them

-tweet_id column is int type it should be string (object) type because I will not do any calculation or manipulation

**Tidiness**

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.

*Twitter_archive_enhanced_data:*

-In Twitter_archive_enhanced_data doggo, floofer, pupper, puppo columns merge them in one column called dog_stage

*Twitter_archive_enhanced_data + Image_predictions_data + Tweet_json_data*

-Merge all the dataset in one table

## 3-Clean : In the cleaning part first, I copy the data in another *DataFrame* so I can do manipulation on it at the same time it doesn't affect the real data. The process is divided it into define, code and test. I cleaned all of the problems either related to quality or tidiness programmatically. As much as I could with my previous knowledge, I tried to do it with simple code to make it more understandable and not hard coded. When I finish everything, I merged the datasets together after checking that everything is good and worked as expected, for me it was much easier doing it at end in order to easily recognize any problem