Wrangling report

Overview:

WeRateDogs twitter account, the most popular account that rates dogs every day, had gathered an archive of old tweets' data from 2015 to 2017 and sent it to Udacity for students to use and make interesting insights and visualizations. The data contains interesting information but like any data that rarely comes clean. Here comes in the steps used to gather, assess and clean data.

This document shows the wrangling steps briefly that prepared data for analyses and visualizations, so let's dive in.

Introduction:

In this project, we have three files that are required to get successful and attractive data analysis.

Twitter Archive: file downloaded by WeRateDogs account and sent to Udacity via email for students to use in project.

Image Predictions: file that contains neural network results of nearest predictions to images passed as input. Udacity took images from tweet text in each tweet.

Tweet_json: file that contains any additional needed data about tweets.

We will gather each file from the server it resides in, asses each gathered data to determine quality and tidiness issues, and finally clean it to make it functional for any insights needed.

Gather Data:

Data is gathered from three different sources:

Twitter Archive: Udacity attached CSV file in project resources for direct use. File is downloaded and saved in same folder of our project. This file is loaded to dataframe directly using pandas.read_csv function.

Image Predictions file: Udacity recorded all the results of neural network to its servers and provided students with the URL of the results location on them. Requests library is required to access data from servers and save it to TSV file. The TSV file is then loaded directly to dataframe using pandas.read_csv function, but this time it is important to specify sep='\t' to function to indicate values are separated by Tabs.

Tweet_json file: The provided data of the above two files is not enough to make us draw interesting statistics, so it is essential to query additional data from Twitter. To query data, we need to access Twitter API. Twitter API requires authorization by requesting developer account. After Twitter API access is achieved, it is easy to use twitter tweepy package to query data using tweet ids provided in archive. Queried data is retrieved and stored in JSON format to text file. This file is loaded directly to dataframe using pandas.read_json function.

Assess Data:

Since we have data loaded successfully to three dataframes, it is time to assess data to pick all valuable issues keeping us far from clean functional data. These assessments are done both visually and programmatically.

The types of assessment:

- Visual Assessment is from its name visual means skimming data with eyes. Checking CSV file on Excel, this step is the start for archive and prediction tables before diving in their data programmatically.
- Programmatic Assessment: This is where the pandas package (lifesaving package to data analysis) comes into play. JSON data is loaded to entire dataframe using read_json function, which loads data to dataframe in tabular format. This format helps skimming through json visually before any programmatic assessing. For deep look through all data (the real programmatic assessment), various functions helped investigate deeply such as value_counts function, shape tuple, info function, duplicated function, isna function.

The assess issues:

For spotting issues, it is important to keep in mind that any unclean data have common issues that anyone working with analyses will find them while wrangling and specific issues that are related to data wrangled for further study.

In our case, the Twitter Archive, Image Prediction and JSON data, like any stored data in our world, have the following common issues:

- Incorrect datatype: Checking columns with pandas info function, gives us clear picture of datatypes, count of data in each column, memory allocated to dataframe. Datatype of each column is important based on need of this column in any downstream purposes, the accuracy of data representation, the future use of data and other infinite reasons that differ from data to another. In our project case, it is important to change datatype of tweet ids to string to preserve their accuracy, and timestamp to datetime, so we can use them in our insights. In the specific issues section, we will discuss one incorrect datatype discovered due to other specific issue.

- Unneeded rows and columns: It depend on what data should be available for our analysis. In our case, we only need data for original tweets, so any row in archive indicating reply or retweet is not needed. The columns indicating whether data is a reply or retweet are also unused.
- NAN values: NAN stands for Not A Number. Approximately most of the data existing in our world are subject to NAN values. This existence due to, in the best case, unfilled optional data for example: if people at certain company are filling a survey about anything, it is mostly expected that they will leave optional data unfilled, which is stored in database as empty. But, in many cases, important columns holding important data may come unfilled too. The pandas recognize empty cells as NAN. In the case of our data, it is important to have any numeric data taken to further measure like rating numerator, rating denominator, retweets count, favorite count and image prediction percentages columns fully filled with measurable data. To help find out in these large amounts of data, pandas isna function detects missing values and gives us Boolean result whether certain cell in column is an NA or not. To get count of all NA values in a column, add sum function to sum up all cells (DataFrame.isna().sum() or DataFrame.column.isna().sum()).
- Duplicate values: It Is very crucial that any column used as a primary key must have unique values. Primary key is considered the address of a row, what makes info of certain observation unique from another. In our case, the tweet_id is the primary key or unique address for each row information in the three gathered files. As a result, it is very important to check that we have no duplicate tweet_id which would result in more than one row queried for each tweet. This check is achieved with pandas duplicated function, which returns the result of each row or cell depending on whether function call is on dataframe or dataframe.column. To get full count of duplicates, add sum function to sum up all cells (DataFrame.duplicated.sum() or DataFrame.column.duplicated().sum()). In the specific issues section, we will discuss, duplicate values issue specific to our gathered data.
- Ambiguous column name: did you get a start indication about what data a column might have from the first look? If the answer is no, column needs renaming. Sometimes too, you get a clear picture of data, but column is better understandable if we rename too. In our case, columns are ambiguous without the help of Twitter dictionary that twitter has in its public documentation on search engines. We would have a considerable time investigating data. It is better to save time for us if we came back to these data any other time or people who will reproduce our analyses again to rename columns to clear names. In our case, most columns should be renamed to meaningful names: like url to tweet_url, timestamp to tweet_timestamp, text to tweet_text, and other columns. This makes it clear anytime data is revisited what these columns refer to.
- Data integrity: How can we get clear analyses to give us big picture, draw interesting results and take important decisions from incorrect data? If data is incorrect or

inconsistent, we will for sure get incorrect analyses, not to mention weird results in many cases. Since the integrity issues differ from data to another, we will now navigate to the specific issues that are related to data to discuss this issue and add other issues that might halt having functional data.

<u>Specific issues related to our gathered data:</u>

This section has three issues that are discussed in common issues section that fall under both common issues and specific issue categories:

- Duplicate values: Although we have checked that our primary key tweet_id is not duplicated; we have duplication discovered using pandas duplicate function. The image classification results of tweets were duplicated in the image prediction file, but the primary key was not duplicated. Why? Because simply we do not only have tweets in our archive but also retweets, and of course retweets are too saved to database with the same original tweet's image, but different tweet_id. The person who worked on inputting images to neural network from archive passed them sequentially. We got for one image duplicate prediction results, resulting from passing images of tweets and retweets.
- Data integrity: Although it is a common to find integrity issues in any world data, the integrity required from one to another differs. Our data has the following integrity issues:
  - ➢ There are two important columns that are taken as measurable variables in analyses. The numerator and denominator rating columns. These columns are filled using an extractor that takes the first occurrence of numerator/denominator in tweet text with no criteria of choice. This makes us jump to a conclusion that rating extracted may not have Twitter page ratings. How can this happen? What might tweet text have with this form other than ratings? Yes, we should expect other values. Let us take this example: one of the tweets texts is "Meet Sam. She smiles 24/7 & secretly aspires to be a reindeer. Keep Sam smiling by clicking and sharing this link: https://gofundme.com/sams-smile" . This tweet has the form numerator/denominator which is 24/7, and the extractor worked as expected extracting 24 to numerator column and 7 to denominator column which is wrong.
  - ➢ Other numerator and denominator rating issue is decimal numbers extracted as integers. The extractor does not extract decimal values but integer values only. Not only extracts integers, but also extracts the numbers after decimal point which leads to complete wrong rating for example rating 9.75/10 is extracted as 75/10. Decimal ratings must be corrected too. In addition, our numerator and denominator columns must have datatypes float not integer. Discussed more in specific issue bulleted Incorrect datatype section.
  - ➢ One of the columns has the dog names tweeted in tweets texts. This column is also filled with extractor that extracts tweet names. Using the pandas value_counts function, we find set of values with small letters at the end of list displayed. These

values are a, an, very, such, and more; impossible to have them as dog names, besides names do always begin with capital letters. By checking their tweets, we find these names are verbs that are part of tweet sentence not dog names.

- Incorrect columns division: This is one of the issues that may or may not occur with gathered data, because it depends on whether group of data across different columns are related or not to the same variable or group. In our case, the data in columns doggo, pupper, puppo and floofer belong to the same group which is dog stage. It is useless having them in different columns and should be merged.
- Incorrect table division: It is like the incorrect columns division but occurs when data is gathered from different sources like our case. Data is not in the same table as doggo, pupper, puppo and floofer, but gathered from three different sources and saved in three different file formats. They belong to the same entity which is tweet_id. Since this id is the common and primary key in all three files or tables discussed in gather data, they should be merged to one table or file.
- Incorrect datatype: We have two specific incorrect datatypes:
    - ➢ Due to previously discussed point of numerator and denominator issue of decimal numbers extracted as integers, we immediately deduce that these columns should have datatypes of float not int64. By looking at numerator and denominator rating Dtypes displayed with info function just described in first common issue, we find they are of int64 datatypes. This makes us navigate back to one of reasons why we need to change datatypes. In this case, the accurate measures of data are taken when columns are float not int.
    - ➢ As a result of merging three files issue discussed in incorrect table division point, the retweet_count and favorite_count datatypes changed from int64, which is their correct datatypes, to float. This issue occurred after merging tables, which lead to new specific incorrect datatypes.

Note: Wrangle_act notebook has assessment issues listed in detail in assess data sub-section of **Assess and clean loaded dataframes** and **Assess and clean master dataframe sections** with their right division explained above: the visual assessment and the programmatic assessment.

Clean Data:

The data is collected, assessed and major issues are listed. Now, it is time to clean these issues to have data functional.

Unclean data falls under two categories:

- Dirty data which is a quality issues data.
- untidy data is not an issue in data but needs enhancement. Untidy data holds the learned definition:

every variable is a column, each observation is a row, and each type of observational unit is a table.

The clean of assessed issues:

Pandas package continues to provide us favors. It not only assists in assess phase but also in clean phase.

In this section, we will go through each assessed issue by placing it under its correct category (quality or tidiness issue) and providing the solution to solve it.

Quality issues:

- Incorrect datatype: This issue is addressed easily by casting columns or objects to specified datatype.
    - ➢ The column tweet_id in three tables is casted to recommended type string, using astype function in pandas library.
    - ➢ Timestamp column, in archive file indicating tweeting time of tweet, is casted to datetime using pandas to_datetime function.
    - ➢ Numerator and denominator columns in archive file are casted to float, using astype function in pandas library.
    - ➢ The retweet_count and favorite_count columns in JSON file are casted to int from float in merged dataframe, using astype function in pandas library,
- Unneeded rows and columns: All rows indicating reply or retweet data are dropped. Also, columns flagging reply and retweets are dropped too. These steps are achieved with drop pandas function. These cleanups enable us to have full unique data across each row. The archive table has just original tweets data with no duplication, retweets, or replies, which we will use in solving the issue we have in duplicate values section
- NAN values: In this project, we have 7 NAN values in retweet_count and favorite_count columns in JSON file. NAN values are filled directly with pandas fillna function. These 7 columns have NANs means these tweets did not receive attention by neither retweet nor likes, so 0 is the correct fill to these empty values.
- Duplicate values: There is only one duplicate issue which is image results passed to the classifier (explained in assess section). Issue is solved using pandas merge function to merge two files or tables: archive and image predictions, taking their intersection on common key between them which is tweet_id. How did merge solve the issue? Remember, we dropped any row indicating data is reply or retweet. And, the data resulting from classifier is duplicated due to image passed more than once. The intersection of merge drops the duplicates in image prediction table since tweet_ids of retweets are not in archive table. The merge result is a clean data of basic tweet info and its image predictions. Important info about merge is clarified in tidy issues section.

- Ambiguous column name: We have many columns that need meaningful renaming, which is achieved using pandas rename function. The following shows how I renamed some columns, but any renaming is acceptable. The clarity point of view absolutely differs from one person to another.
    - Archive table : columns expanded_url is renamed to tweet_url, timestamp is renamed to tweet_timestamp, text is renamed to tweet_text, and finally name is renamed to dog_name.
    - Image predictions table: columns jpg_url is renamed to tweet_img_url, img_num is renamed to nearest_prediction, p1 is renamed to prediction1,..
    - JSON file: id is renamed to tweet_id. This column must have this renaming since all data relates together based on it.
- Data integrity:
    - Large amounts of incorrect data pushes anyone to sit back and think of tools or algorithm(s) that can facilitate finding and correcting integrity issues in large data. Integrity issues that comes in millions of records are hard to spot and need a tool or algorithm to facilitate. Luck point here is we do not have much of them in numerator/denominator first format extraction and is easily fixed manually. With the help of pandas loc property, we just need to locate rows and columns (numerator and denominator) incorrectly extracted to fix them.
    - Another issue that occurred with extraction is decimal ratings incorrectly extracted. With the help of regex expressions, pandas retrieved us all decimal ratings. The decimal ratings retrieved are then fixed one by one using pandas loc property too to fix numerator cells in specific rows
    - Incorrect dog names are fixed too with pandas replace function that took list of verbs in our data and replaced them with None, since they are part of tweet sentence not dog names.

<u>Tidiness issues:</u>

- Incorrect columns division: the data in columns doggo, pupper, puppo and floofer belong to the same group which is dog stage. These columns are concatenated to one column using set of steps where pandas assisted too in.
- Incorrect table division: Each table has set of values that belong to tweets. The data in three files or tables are better merged to one table. Let us get back to the duplicated image results that we documented under quality issues; The duplication in data is a quality issue that was solved by solving a tidiness issue. If we did not have any duplication, the merging of three table would have been recorded completely under tidiness section. But since we solved duplication issue by merging archive and image prediction tables, here we will complete the merging of third table of JSON file to two merged table. By merging three tables, we now have a data that is completely, correctly connected to each other, cleaned and ready for our analyses.

Note: Wrangle_act notebook has cleaning issues listed in detail in clean data sub-section of **Assess and clean loaded dataframes** and **Assess and clean master dataframe sections** with their right division explained above: quality and tidiness issues

Data Clean process has many ways of sequencing the steps in notebooks. In this project, I followed the sequence of **Define**, **Code**, **Test**:

- Define has the issue solution described in words
- Code has the code used to solve the issue. All issues are solved with help of pandas package.
- Test: has the code to ensure cleaning steps were done successfully. Also, test is achieved with the help of pandas package.

Summary:

Data wrangling is the process of gathering, assessing, and cleaning data, to have it functional for a variety of downstream purposes such as analytics. In this project, Data is gathered from three different sources and loaded to pandas dataframes for further processing, assessed to identify quality and tidiness issues, and last cleaned to have it functional.

Packages imported:

requests

json

tweepy

timeit from default_timer

numpy

pandas

matplotlib.pyplot

seaborn

Important Note:

This is a documentation of the wrangle efforts and information deduced from my work on this project.