Wrangling Report

Section #1 - Gathering Data

I began by manually downloading `'twitter-archive-enhanced.csv'` from Udacity's given URL. Then, I created the module `gather.py` ***(detailed docstrings within the module)***, which consists of the functions I needed in order to gather the required data. The first method - `gather.download_file_from_url` was used to download and save the `image_predictions.tsv` file programmatically. Then, I went on to create the functions necessary for getting WeRateDogs' retweet and favorites counts:

1. `gather.get_auth`, which returns an auth object required for getting tweet's data from the tweeter API.
2. `gather.dump_tweets_data`, which dumps json-formatted counts data into `tweets_data.json`.

SIDE-NOTE: It was crucial to take account of the APIs rate limit, otherwise a lot of valid data would be ignored by the function.

Finally, I loaded the collected data into three DataFrames:

1. `archive_original` for `'twitter-archive-enhanced.csv'`.
2. `tweet_cnts` for `tweets_data.json`.
3. `img_preds` for `image_predictions.tsv`.

At a quick glance, I noticed that in all of the above frames the id column is cast by default as int, So as a quick quality fix I've cast them to `str`, as the latter would be the appropriate data type for id.

Section #2 - Assessing Data

Total quality issues - 8

Total tidiness issues - 4

Total issues - 12

`archive_original` issues:

Quality issues - 7

- *completeness*
  - missing tweet links urls in `expanded_urls` data, even in non empty cells. Detected visually - `archive_original.iloc[313]`, `archive_original.iloc[35]`
  - because we would like to clearly see if the relevant id is an original tweet, a boolean column `is_original` should be included.
  - because we would like to clearly see if the relevant id is a reply, a boolean column `is_reply` should be included.
- *validity* -

    - inappropriate data type for columns `in_reply_to_status_id`, `in_reply_to_user_id`, `retweeted_status_id`, `retweeted_status_user_id`: float -> string
    - inappropriate data type for columns `timestamp`, `retweeted_status_timestamp`: object -> datetime. both issues detected by using programmatic detection- `archive_original.info()`
    - duplicated links within the same cell in `expanded_urls`. Detected visually - `archive_original['expanded_urls'].iloc[98]`
- *consistency* -
    - name 'a' appears 55 times in the `name` column, probably representing unknown names, while there are defined None's in the column. Used visual, then programmatic detection- `archive_original['name'].value_counts()`

Tidiness issues - 3

- the 4 leftmost columns describe the same data, should be melted into one categorical column.
- retweeted data columns would be better separated as a standalone dataframe.
- replies data would be better separated as a standalone dataframe.

`tweet_cnts` issues:

Quality issues - 1

- *validity*
    - count columns should be converted to int to avoid confusion.

Tidiness issues - 1

- dataframe would be better joined to archive_original, as it consists of general tweets data.

`img_preds` issues:

Used both programmatical and visual assessments, did not observe any issues. Also, no multiple records of same id's. One picture for each id. Used programmatic detection: `len(img_copy[img_copy['tweet_id'].duplicated()])`

Section #3 - Cleaning Data

`archive_original` cleaning:

Issue #1: missing `expanded_urls` data.

- concat `tweet_id` to WeRateDogs url prefix where `expanded_urls` is null or tweet url is missing:
    - first fill nan
    - then check entire column for missing pattern
    - finally, append a comma then the pattern to all missing it.

Issue #2: because we would like to clearly see if the relevant id is an original tweet, a boolean column is_original should be included.

- if `retweeted_status_id` is null, then `is_original` is True, else False.

Issue #3: inappropriate data type for columns in_reply_to_status_id, in_reply_to_user_id, retweeted_status_id, retweeted_status_user_id: float -> string

- iterate over the columns
- then, convert the dtype of each one to int in order to disable the scientific notation.
- finally, convert the dtype of each one to string.

Issue #4: inappropriate data type for columns `timestamp`, `retweeted_status_timestamp`: object -> datetime.

- convert given columns to datetime using Pandas' `to_datetime` method.

Issue #5: because we would like to clearly see if the relevant id is a reply, a boolean column `is_reply` should be included.

- add a boolean column, `is_reply`: `True` if id has reply data, else `False`.

Issue #6: duplicated links within the same cell in `expanded_url`.

- split strings at `,` to list.
- set() on the output.
- join the set using `,`
- apply to column using Pandas' `map` function.

Issue #7: name 'a' appears 55 times in the `name` column, probably representing unknown names, while there are defined None's in the column.

- check if cell value is 'a' or 'None'
- if true, convert to Pandas' `NA`
- else, do nothing.
- apply to column using Pandas' `map`.

Issue #8: the 4 leftmost columns describe the same data, should be melted into one categorical column.

- Use Pandas' `melt` function to melt the columns into a category and value column.
- create in a temporary dataframe that consists of only the records which do have a category(not 'None').
- drop its excess `value` column
- instead of simply removing duplicated id's, format the categories' column in the temporary dataframe to include multiple categories for each id, as each tweet can validly contain multiple categories.
- convert temp to dtype string for proper nullable data type on left join.
- ***NOTE - AS records may have some combination of categories, a string dtype instead would be more appropriate.***
- left join `archive_copy` with temp on `tweet_id`
- drop the old category columns from `archive_copy`

Issue #9: retweeted data columns would be better separated as a standalone dataframe.

- create a DataFrame from these columns `retweet_data`.
- include only retweets(`is_original` False)

- drop columns from `archive_copy`.

Issue #10: replies data would be better separated as a standalone dataframe.

- create a DataFrame from these columns
- include only replies(`is_reply` true)
- drop columns from `archive_copy`.

`tweet_cnts` cleaning:

Issue #11: `tweet_cnts` would be better joined to archive_original, as it consists of general tweets data.

- left join `archive_copy` and `tweet_copy` on id.
- drop excess id column.

Issue #12: count columns should be converted to int to avoid confusion.

- Address the issue in `archive_copy', as it includes the tweet counts dataframe.
- convert columns to Pandas' `Int64Dtype`

Bonus

Issue #13 - reindex all columns by priority left to right in`archive_copy`

- reorder columns using reindex method.

Issue #14 - use accurate dtype for each column in `master_df`

- Although most column dtypes are accurate, should convert `object` -> `pd.StringDtype`, as well as all p_dog columns to `boolean`.

Finally, After clearing all the issues above, I've stored the clean data in three csv files:

- `twitter_archive_master.csv` - containing merged tweet archive, images datasets, and tweet counts. Containing all records from the original archive.
- `retweet_data.csv` - containing retweet data for the retweets in tweet archive.
- `reply_data.csv` - consists of reply data for the replies in tweet archive.