



DATA WRANGLING REPORT

Wrangling the master dataset



CREATED BY
Habiba Hisham Hatata

Steps

Gathering Data

We were mainly provided with 3 datasets

- CSV file twitter-archive-enhanced

Which I downloaded directly from the classroom

```
[ ] #csv file
df = pd.read_csv('twitter-archive-enhanced.csv')
```

- TSV file image-predictions.tsv

which I downloaded through the link provided in the classroom with the library “request”

```
[ ] #tsv file
a = requests.get("https://d17h27t6h51sa5.cloudfront.net/topher/2017/August/599fd2ad_image-predictions/image-predictions.tsv")

[ ] with open('image-predictions.tsv', 'w') as f:
    f.write(a.text)

[ ] data = pd.read_csv('image-predictions.tsv', sep = '\t')
```

- JSON file

which we can get using the Twitter API, but because accessing the Twitter API requires having a premium account on X (Twitter), I got the JSON file directly from the classroom sources

```
[ ] #json file
# I got the json file from the downloads in the classroom
data_json = pd.read_json('tweet-json', lines = True)
```

Assessing data

- Functions used:

- Scan function

```
[ ] def scan(dataframe):  
    print('these are the main information')  
    print()  
    print(dataframe.info())  
    print()  
    print('there are ', dataframe.duplicated().sum(), 'duplicates')  
    print()  
    print('these are the nan values per column ', '\n', dataframe.isna().sum() )  
    print()  
    print('numeric columns information', '\n' , dataframe.describe())
```

I used this function to avoid writing each line of code every time I wanted to scan the dataset, especially since I prefer to scan the dataset before and after cleaning it to make sure that everything is cleaned properly

- Remove function

```
[ ] def remove(df, col, values):  
    return df[~df[col].isin(values)]
```

I used the remove function because there were some columns with inaccurate values that must be removed from the dataset. This function was very useful in performing this task.

- Visual assessment

- CSV assessment

- 1- So much inaccurate data

The data is supposed to be for only dogs, but surprisingly, I found many tweets for other animals, even after cleaning the data.

This polar bear was staged as a pupper!

- 2- Overvaluation

Most of the dogs' ratings are higher than 10

And the average rating after changing all the ratings

Higher than 10, with 10 being 9. And most of the ratings

lower than 9 were not for dogs

- TSV assessment

By reviewing the columns, the main problem was the inaccurate image predictions

For example, this column

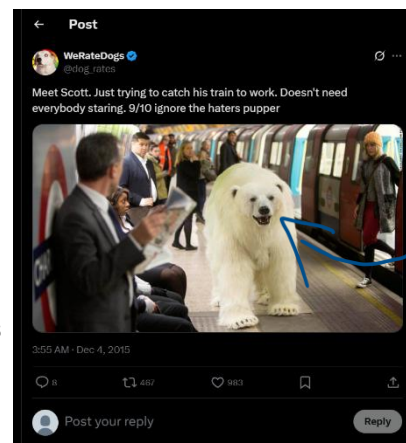


image.csv X

11 to 20 of 2075 entries Filter

	tweet_id	jpg_url	img_num	p1	p1_conf	p1_d
10	666063827256086533	https://pbs.twimg.com/media/CT5Vg_wXIAAXfnj.jpg	1	golden_retriever	0.77593	True
11	666071193221509120	https://pbs.twimg.com/media/CT5cN_3WEAAIoZ.jpg	1	Gordon_setter	0.503672	True
12	666073100786774016	https://pbs.twimg.com/media/CT5d9DZXAAALcwe.jpg	1	Walker_hound	0.260857	True
13	666082916733198337	https://pbs.twimg.com/media/CT5m4VGWEAAIKc8.jpg	1	pug	0.489814	True
14	66609400022159362	https://pbs.twimg.com/media/CT5w9gUW4AAsBNN.jpg	1	bloodhound	0.195217	True
15	666099513787052032	https://pbs.twimg.com/media/CT51-JJUEAA6hV8.jpg	1	Lhasa	0.58233	True
16	666102155909144576	https://pbs.twimg.com/media/CT54YGIWUAEZnoK.jpg	1	English_setter	0.298617	True
17	666104133288665088	https://pbs.twimg.com/media/CT56LSZW0AAUj2.jpg	1	hen	0.965932	False
18	666268910803644416	https://pbs.twimg.com/media/CT8QCd1WEAADXws.jpg	1	desktop_computer	0.086502	False
19	666273097616637952	https://pbs.twimg.com/media/CT8T1mtUwAA3aqm.jpg	1	italian_greyhound	0.176053	True

Show 10 per page 1 2 3 10 100 200 208

The prediction shows that the image includes a desktop computer, but includes a dog!



→ JSON assessment

The main problem I faced with this dataset was the useless columns.

display_text_range	entities	extended_entities
[0, 85]	{'hashtags': [], 'symbols': [], 'user_mentions': []}	{'media': [{'id': 892420639486877696, 'id_str':...
[0, 138]	{'hashtags': [], 'symbols': [], 'user_mentions': []}	{'media': [{'id': 892177413194625024, 'id_str':...
[0, 121]	{'hashtags': [], 'symbols': [], 'user_mentions': []}	{'media': [{'id': 891815175371796480, 'id_str':...
[0, 79]	{'hashtags': [], 'symbols': [], 'user_mentions': []}	{'media': [{'id': 891689552724799489, 'id_str':...
[0, 138]	{'hashtags': [], 'text': 'BarkWeek', 'indices': ...}	{'media': [{'id': 891327551943041024, 'id_str':...

- Programmatic assessment

→ CSV assessment

➤ quality issues

The rating_numerator is larger than the rating_denominator

The timestamp must be in the datetime type.

Some names in the 'names' column are not valid, like 'an, 'a, 'the', all', etc.

handling nan values

Replace with unknown in name and expanded URLs

Remove retweets and replies

The retweets are the rows which have a value in the retweet_status_id

The replies are the rows which have a value in the in_reply_to_status_id

➤ tidiness issues

remove useless columns in_reply_to_status_id, in_reply_to_user_id, because we will remove the replies, retweeted_status_id, retweeted_status_user_id, retweeted_status_timestamp, because we will remove retweets, source won't be used in the analysis

Remove the denominator column and write its value, which is 10, in the numerator column name

merge puppo, pupper, floofer, doggo in one column called 'age'

→ TSV assessment

➤ quality issues

Some columns include images not of dogs, for example: a hen, a turtle

➤ tidiness issues

Rename the columns to be clear and understandable

→ JSON assessment

➤ quality issues

Remove all retweets and replies, and quoted tweets

Handle nan values

1- tidiness issues

Remove unwanted columns

Rename the columns to be clear and understandable, and the column id to 'retweet_id' to match other datasets

Cleaning data

The last step of the wrangling process is the cleaning, where we solve the problems we found in the assessing step.

These are some screenshots from the cleaning step.

```
[ ] # there are some columns include images not for dogs for example: hen , turtle
cleaned_tsv = cleaned_tsv[cleaned_tsv['p1_dog'] != False]
cleaned_tsv = cleaned_tsv[cleaned_tsv['p2_dog'] != False]
cleaned_tsv = cleaned_tsv[cleaned_tsv['p3_dog'] != False]

[ ] # manual checking for the urls
cleaned_tsv['jpg_url'].tail(50)

Show hidden output

[ ] # tidiness issue: remove the columns which won't be used in the analysis
cleaned_tsv = cleaned_tsv.drop(['p1_conf', 'p2_conf', 'p3_conf',
                              'p1_dog', 'p2_dog', 'p3_dog'], axis = 1)
```

```
[ ] # 2-timestamp must be in datetime type
cleaned_df['timestamp'] = pd.to_datetime(cleaned_df['timestamp'])

[ ] cleaned_df['timestamp'].dtype
datetime64[ns, UTC]

[ ] # 3-some names in the 'names' column are not valid like 'an', 'a', 'the', 'all', etc
cleaned_df = remove(cleaned_df, 'name', [ 'such', 'a', 'quite', 'not', 'one', 'mad', 'an',
                                           'very', 'o', 'just', 'my', 'his', 'Bookstore', 'getting',
                                           'actually', 'this', 'unacceptable', 'all', 'old', 'the', 'nan'
                                           ])

[ ] cleaned_df['name'].isin([ 'such', 'a', 'quite', 'not', 'one', 'mad', 'an',
                             'very', 'o', 'just', 'my', 'his', 'Bookstore', 'getting',
                             'actually', 'this', 'unacceptable', 'all', 'old', 'the', 'nan'
                             ])
#changed successfully
```

```
[ ] # 3-remove the denominator column and write its value in the numerator column
cleaned_df.drop('rating_denominator', axis = 1, inplace = True)
cleaned_df.rename(columns = {'rating_numerator': 'rating_of_10'}, inplace = True)

[ ] # 4-merge puppo, pupper, floofer, doggo in one column called 'age'
cleaned_df[['doggo', 'floofer', 'pupper', 'puppo']] = cleaned_df[['doggo', 'floofer', 'pupper', 'puppo']].fillna(' ')

[ ] cleaned_df['age'] = cleaned_df[['doggo', 'floofer', 'pupper', 'puppo']].agg(' '.join, axis = 1)

[ ] cleaned_df['age'].value_counts() #we need to exploit these columns with two
```

	count
age	
	1676
pupper	210
doggo	70
puppo	23

```
[ ] # the mode is zero for them both
cleaned_json.fillna(0, inplace = True)

[ ] cleaned_json.drop(columns = ['entities', 'extended_entities', 'source', 'user', 'retweeted'], inplace = True)

[ ] cleaned_json.drop(columns = ['possibly_sensitive', 'possibly_sensitive_appealable'], inplace = True) #all their values are 0s

[ ] #rename the columns to be clear and understandable and the columns to match other datasets
cleaned_json = cleaned_json.rename(columns= {'id' : 'tweet_id', 'lang' : 'language', 'favorited': 'is_favorited', 'created_at' : 'timestamp'})

[ ] scan(cleaned_json) #final scan...we removed too many columns but they won't be necessary and it will be a part of a large merged dataset

these are the main information

<class 'pandas.core.frame.DataFrame'>
Index: 2068 entries, 0 to 2353
Data columns (total 7 columns):
#   Column                Non-Null Count  Dtype
---  -
0   timestamp              2068 non-null   datetime64[ns, UTC]
1   tweet_id               2068 non-null   int64
2   text                   2068 non-null   object
3   retweet_count          2068 non-null   int64
4   favorite_count         2068 non-null   int64
5   is_favorited           2068 non-null   bool
```

Suggestions and Areas of Improvement

- 1- Use multiple sources for dog information, not just WeRateDogs, to gain more valuable insights.
- 2- Data cleanliness was a major challenge in the wrangling process, with many NaN values, and some columns having no non-null entries at all.