Wrangle data report

By : Irfan Septiyana Putra

In this project, I am doing data wrangling step on data from WeRateDogs twitter archive. All step of data wrangling such as gathering data, asses data and cleaning data is done. In this project gathering data is done by downloading data from udacity, from url and twitter api. After assessing data, found 9 quality issues and 2 issues in tidiness which will explain below. Each issue is cleaned with three steps : define problem, code and test. Those steps are really help data cleaning process since each issue is cleaned and tested one by one.

Gathering data is the first process in data wrangling. The first one ,tweeter archive data, I get by downloading it in udacity. I need to make request to url using get method in request library and then store it into image-prediction.tsv file. The final data is gotten from twitter api by helping from tweepy library. To get data from tweeter, authentication process is needed befor pulling data from twitter. Then, requesting data is done by requesting data with tweet_id (from first data) and this process needs timer to make the pull request is not passing tweeter api limit request. This data stored as tweet_json.txt.

The next step is asses data. First step is read stored data and make it on dataframe. Tweet archive and image prediction data is read using read_csv method. I need to get data line per line using json load and store it to list of dictionaries, exracting tweet_id, text, retweeted_count and favorite count and change it into dataframe.

Quality Issues Assesment

First assesed is display each data head, info and describe. Gotten first issue in tweeter archive .Name column is floofer, this must be replace with floof.

```
twitter_archive_enh.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2356 entries, 0 to 2355
Data columns (total 17 columns):
tweet_id                    2356 non-null int64
in_reply_to_status_id       78 non-null float64
in_reply_to_user_id         78 non-null float64
timestamp                   2356 non-null object
source                      2356 non-null object
text                        2356 non-null object
retweeted_status_id         181 non-null float64
retweeted_status_user_id    181 non-null float64
retweeted_status_timestamp  181 non-null object
expanded_urls               2297 non-null object
rating_numerator            2356 non-null int64
rating_denominator          2356 non-null int64
name                        2356 non-null object
doggo                       2356 non-null object
floofer                     2356 non-null object
pupper                      2356 non-null object
puppo                       2356 non-null object
```

Figure 1 twitter archive info

Seeing twitter archive info, gotten timestamp in string type, must be indatetime type. in_reply_to_status_id and in_reply_to_user_id must be in integer instead of float. Checking retweet_status_id, gotten several retweet data or duplication data which is not desireable, fourth issue.

	tweet_id	in_reply_to_status_id	in_reply_to_user_id	retweeted_status_id	retweeted_status_user_id	rating_numerator	rating_denominator
count	2.356000e+03	7.800000e+01	7.800000e+01	1.810000e+02	1.810000e+02	2356.000000	2356.000000
mean	7.427716e+17	7.455079e+17	2.014171e+16	7.720400e+17	1.241698e+16	13.126486	10.455433
std	6.856705e+16	7.582492e+16	1.252797e+17	6.236928e+16	9.599254e+16	45.876648	6.745237
min	6.660209e+17	6.658147e+17	1.185634e+07	6.661041e+17	7.832140e+05	0.000000	0.000000
25%	6.783989e+17	6.757419e+17	3.086374e+08	7.186315e+17	4.196984e+09	10.000000	10.000000
50%	7.196279e+17	7.038708e+17	4.196984e+09	7.804657e+17	4.196984e+09	11.000000	10.000000
75%	7.993373e+17	8.257804e+17	4.196984e+09	8.203146e+17	4.196984e+09	12.000000	10.000000
max	8.924206e+17	8.862664e+17	8.405479e+17	8.874740e+17	7.874618e+17	1776.000000	170.000000

Figure 2 tweeter archive describe result

From statistic data above, it's strange that rating numerator and denominator have big max number which is very far from mean and thirt quartile, adding additional column needed.

	rating_numerator	rating_denominator	normalize_rating
count	2356.000000	2356.000000	2356.000000
mean	13.126486	10.455433	inf
std	45.876648	6.745237	NaN
min	0.000000	0.000000	0.000000
25%	10.000000	10.000000	1.000000
50%	11.000000	10.000000	1.100000
75%	12.000000	10.000000	1.200000
max	1776.000000	170.000000	inf

Figure 3 twiiter archive with normalize column

```
twitter_archive_enh.normalize_rating.sort_values(ascending=False)
```

```
313          inf
979    177.600000
189     66.600000
2074    42.000000
188     42.000000
290     18.200000
340      7.500000
695      7.500000
516      3.428571
763      2.700000
1712     2.600000
55       1.700000
285      1.500000
291      1.500000
64       1.400000
418      1.400000
924      1.400000
68       1.400000
186      1.400000
```

Figure 4 sort values normalize_rating

```
twitter_archive_enh.query('normalize_rating==inf')
```

eeted_status_user_id	retweeted_status_timestamp	expanded_urls	rating_numerator	rating_denominator	name	doggo	floofer	pupper	puppo	normalize_rating
NaN	NaN	NaN	960	0	None	None	None	None	None	inf

Figure 5 tweeter archive inf value row

```
list(twitter_archive_enh.query('normalize_rating==inf').text)
```

```
["@jonnysun @Lin_Manuel ok jomny I know you're excited but 960/00 isn't a valid rating, 13/10 is tho"]
```

Figure 6 inf value rows, text content

From four pictures about, concluded rating_denominator and numerator is wrong.

	tweet_id	jpg_url	img_num	p1	p1_conf	p1_dog	p2	p2_conf	p2_dog	p3	p3_conf	p3_dog
0	666020888022790149	https://pbs.twimg.com/media/CT4udn0WwAA0aMy.jpg	1	Welsh_springer_spaniel	0.465074	True	collie	0.156665	True	Shetland_sheepdog	0.061428	True
1	666029285002620928	https://pbs.twimg.com/media/CT42GRgUYAA5iDo.jpg	1	redbone	0.506826	True	miniature_pinscher	0.074192	True	Rhodesian_ridgeback	0.072010	True
2	666033412701032449	https://pbs.twimg.com/media/CT4521TWwAEvMyu.jpg	1	German_shepherd	0.596461	True	malinois	0.138584	True	bloodhound	0.116197	True
3	666044226329800704	https://pbs.twimg.com/media/CT5Dr8HUEAA-lEu.jpg	1	Rhodesian_ridgeback	0.408143	True	redbone	0.360687	True	miniature_pinscher	0.222752	True
4	666049248165822465	https://pbs.twimg.com/media/CT5IQmsXIAAKY4A.jpg	1	miniature_pinscher	0.560311	True	Rottweiler	0.243682	True	Doberman	0.154629	True

Figure 7 image prediction info

Inconsistent of using lower and uppercase and using '_' and '-' as separator make dog prediction stage is not clear. Using value_counts method, found several duplication data / same jpg_url.

```
image_predictions.jpg_url.value_counts()
```

```
https://pbs.twimg.com/media/C2oRbOuWEAAbVSl.jpg                                    2
https://pbs.twimg.com/media/CvJCabcWgAIoUxW.jpg                                    2
https://pbs.twimg.com/media/CUN4Or5UAAAa5K4.jpg                                    2
https://pbs.twimg.com/media/Cs_DYr1XEAA54Pu.jpg                                    2
https://pbs.twimg.com/media/CiibOMzUYAA9Mxz.jpg                                    2
https://pbs.twimg.com/media/CVMOlMiWwAA4Yxl.jpg                                    2
https://pbs.twimg.com/media/CYLDikFWEAAIy1y.jpg                                    2
https://pbs.twimg.com/media/CmoPdmHW8AAi8BI.jpg                                    2
https://pbs.twimg.com/media/Ct2qO5PXEAE6eB0.jpg                                    2
https://pbs.twimg.com/media/CtKHLuCWYAA2TTs.jpg                                    2
https://pbs.twimg.com/tweet_video_thumb/CeBym7oXEAEWbEg.jpg                        2
https://pbs.twimg.com/media/CtzKC7zXEAALfSo.jpg                                    2
https://pbs.twimg.com/media/CcG07BYW0AErrC9.jpg                                    2
https://pbs.twimg.com/media/CW88XN4WsAAlo8r.jpg                                    2
https://pbs.twimg.com/media/CsVO7ljW8AAckRD.jpg                                    2
https://pbs.twimg.com/media/CVuQ2LeUsAAIe3s.jpg                                    2
https://pbs.twimg.com/ext_tw_video_thumb/807106774843039744/pu/img/8XZg1xW35Xp2J6JW.jpg    2
https://pbs.twimg.com/media/Ct72q9jWcAAhlnw.jpg                                    2
https://pbs.twimg.com/media/CWyD2HGUYAQ1Xa7.jpg                                    2
```

Figure 8 value_counts result of jpg_url column in image prediction

```
image_predictions[image_predictions['jpg_url']=='https://pbs.twimg.com/media/CsGnz64WYAEIDHJ.jpg']
```

	tweet_id	jpg_url	img_num	p1	p1_conf	p1_dog	p2	p2_conf	p2_dog	p3	p3_conf	p3_dog
1441	775085132600442880	https://pbs.twimg.com/media/CsGnz64WYAEIDHJ.jpg	1	chow	0.316565	True	golden_retriever	0.241929	True	Pomeranian	0.157524	True
1601	799774291445383169	https://pbs.twimg.com/media/CsGnz64WYAEIDHJ.jpg	1	chow	0.316565	True	golden_retriever	0.241929	True	Pomeranian	0.157524	True

Figure 9 Duplication data example in image prediction

Analyze json data, duplication data found by extracting text that having 'RT @'

```
cek_rt_json = json_df.full_text.str.extract(r'(RT @.*)')
cek_rt_json[cek_rt_json[0].notnull()]
```

	0
31	RT @Athletics: 12/10 #BATP https://t.co/WxwJmv...
35	RT @dog_rates: This is Lilly. She just paralle...
67	RT @dog_rates: This is Emmy. She was adopted t...
72	RT @dog_rates: Meet Shadow. In an attempt to r...
73	RT @dog_rates: Meet Terrance. He's being yelle...
77	RT @rachel2195: @dog_rates the boyfriend and h...
90	RT @dog_rates: This is Coco. At first I though...
95	RT @dog_rates: This is Sierra. She's one preci...
105	RT @dog_rates: This is Dawn. She's just checki...
119	RT @dog_rates: Say hello to Cooper. His expres...
125	RT @rachaeleasler: these @dog_rates hats are 1...

Figure 10 duplication data by extracting text

Tidiness Data

From figure 1, timestamp contain date and time. This makes data untidy, data must be separated.

Text contain both tweeter content and url, this need to be fixed too.

```
twitter_archive_enh['text'][571]
```

```
"This is Wallace. He'll be your chau-fur this evening. 12/10 eyes on the road Wallace https://t.co/p1RD39XjUe"
```

Figure 11 Content value example in tweeter archive

Cleaning data process will be explained below in per issue.

Quality issues

1. timestamp and retweeted status timestamp has type string, which will good in date type

Fixed using astype method for each column.

```
#change data type in timestamp into datetime64
twit_arc_copy['timestamp'] = twit_arc_copy.timestamp.astype('datetime64')
```

```
#change data type in retweet_status_timestamp into datetime64
twit_arc_copy['retweeted_status_timestamp'] = twit_arc_copy.retweeted_status_timestamp.astype('datetime64')
```

Figure 12 code change astype fro 2 columns

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2356 entries, 0 to 2355
Data columns (total 18 columns):
tweet_id                    2356 non-null int64
in_reply_to_status_id       78 non-null float64
in_reply_to_user_id         78 non-null float64
timestamp                   2356 non-null datetime64[ns]
source                      2356 non-null object
text                        2356 non-null object
retweeted_status_id         181 non-null float64
retweeted_status_user_id    181 non-null float64
retweeted_status_timestamp  181 non-null datetime64[ns]
expanded_urls               2297 non-null object
rating_numerator            2356 non-null int64
rating_denominator          2356 non-null int64
name                        2356 non-null object
doggo                       2356 non-null object
floofer                     2356 non-null object
pupper                      2356 non-null object
puppo                       2356 non-null object
normalize_rating            2356 non-null float64
dtypes: datetime64[ns](2), float64(5), int64(3), object(8)
memory usage: 331.4+ KB
```

Figure 13 Test result, timestamp and retweet status timestamp type changed into datetime

2. wrong column name, floofer must be floof

```
In [81]: # Change column named 'floofer' become 'floof'
         twit_arc_copy.rename(columns={'floofer':'floof'}, inplace = True)
```

Figure 14 changing column name

Changing column name using syntax above and recheck it by listing column name.

```
# listing all column names in twit archive clean
list (twit_arc_copy.columns)
```

```
['tweet_id',
 'in_reply_to_status_id',
 'in_reply_to_user_id',
 'timestamp',
 'source',
 'text',
 'retweeted_status_id',
 'retweeted_status_user_id',
 'retweeted_status_timestamp',
 'expanded_urls',
 'rating_numerator',
 'rating_denominator',
 'name',
 'doggo',
 'floof',
 'pupper',
 'puppo',
 'normalize_rating']
```

Figure 15 tweeter archive column name listing

3. in_reply_to_status_id and in_reply_to_user_id type is in float, but it must be in integer

```
# change in_reply_to_status_id and in_reply_to_user_id data type to int
twit_arc_copy.in_reply_to_status_id = twit_arc_copy.in_reply_to_status_id.astype(int)
twit_arc_copy.in_reply_to_user_id = twit_arc_copy.in_reply_to_user_id.astype(int)
```

Figure 16 change type using astype

Test the result using info method, resulting in_reply_to_status_id and in_reply_to_user_id in Integer.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2356 entries, 0 to 2355
Data columns (total 18 columns):
tweet_id                    2356 non-null int64
in_reply_to_status_id       2356 non-null int64
in_reply_to_user_id         2356 non-null int64
timestamp                   2356 non-null datetime64[ns]
source                      2356 non-null object
text                        2356 non-null object
retweeted_status_id         181 non-null float64
retweeted_status_user_id    181 non-null float64
retweeted_status_timestamp  181 non-null datetime64[ns]
expanded_urls               2297 non-null object
rating_numerator            2356 non-null int64
rating_denominator          2356 non-null int64
name                        2356 non-null object
doggo                       2356 non-null object
floof                       2356 non-null object
pupper                      2356 non-null object
puppo                       2356 non-null object
normalize_rating            2356 non-null float64
```

Figure 17 Info result tweeter archive

4. rating_numerator and rating_denominator in row 313 value on normalize_rating is 13/10 instead of 960/0

```
twit_arc_copy.rating_numerator[313] = 13
twit_arc_copy.rating_denominator[313] = 10
```

```
twit_arc_copy.normalize_rating[313] = twit_arc_copy.rating_numerator[313]/twit_arc_copy.rating_denominator[313]
```

Figure 18 Change rating nominator and denumerator value in row 313, and recalculate retweet favorite ratio

Directly change value in row 313 and checking the result using describe.Now there are no inf value in the last column.

	tweet_id	in_reply_to_status_id	in_reply_to_user_id	retweeted_status_id	retweeted_status_user_id	rating_numerator	rating_denominator	normalize_rating
ount	2.356000e+03	2.356000e+03	2.356000e+03	1.810000e+02	1.810000e+02	2356.000000	2356.000000	2356.000000
nean	7.427716e+17	2.468150e+16	6.668307e+14	7.720400e+17	1.241698e+16	12.724533	10.459677	1.222065
std	6.856705e+16	1.341142e+17	2.293821e+16	6.236928e+16	9.599254e+16	41.518626	6.741801	4.082618
min	6.660209e+17	0.000000e+00	0.000000e+00	6.661041e+17	7.832140e+05	0.000000	2.000000	0.000000
25%	6.783989e+17	0.000000e+00	0.000000e+00	7.186315e+17	4.196984e+09	10.000000	10.000000	1.000000
50%	7.196279e+17	0.000000e+00	0.000000e+00	7.804657e+17	4.196984e+09	11.000000	10.000000	1.100000
75%	7.993373e+17	0.000000e+00	0.000000e+00	8.203146e+17	4.196984e+09	12.000000	10.000000	1.200000
max	8.924206e+17	8.862664e+17	8.405479e+17	8.874740e+17	7.874618e+17	1776.000000	170.000000	177.600000

Figure 19 describe result tweeter archive

5. Retweet data is in dataframe must be deleted

Listing all duplication data with checking value retweeted status data, and deleted listed row.

```
#find Retweet tweet id
rt_tweet = twit_arc_copy[twit_arc_copy.retweeted_status_id.notnull()]['tweet_id'].values.tolist()
len(rt_tweet)
```

```
181
```

```
test_cp = twit_arc_copy.copy()
```

```
#delete rows based on tweet id
for x in rt_tweet:
    rt_ind = twit_arc_copy[twit_arc_copy['tweet_id']== x].index
    twit_arc_copy.drop(rt_ind, inplace=True)
```

Figure 20 Listing duplicated row and delete it from dataframe

```
twit_arc_copy[twit_arc_copy.retweeted_status_id.notnull()]
```

tweet_id	in_reply_to_status_id	in_reply_to_user_id	timestamp	source	text	retweeted_status_id	retweeted_status_user_id	retweeted_status_timestamp	expan

Figure 21 testing result by checking if there are still not nul value in retweeted status id

6. dog prediction will be more clear by replacing '_' separator into space

```
# change '_' and '-' with space
img_pred_copy['p1'] = img_pred_copy['p1'].str.replace('_', ' ', regex=True)
img_pred_copy['p2'] = img_pred_copy['p2'].str.replace('_', ' ', regex=True)
img_pred_copy['p3'] = img_pred_copy['p3'].str.replace('_', ' ', regex=True)
img_pred_copy['p1'] = img_pred_copy['p1'].str.replace('-', ' ', regex=True)
img_pred_copy['p2'] = img_pred_copy['p2'].str.replace('-', ' ', regex=True)
img_pred_copy['p3'] = img_pred_copy['p3'].str.replace('-', ' ', regex=True)
```

Figure 22 Changing char in value using str.replace methode

Using str.replace, we can change all unwanted char in column.

	tweet_id	jpg_url	img_num	p1	p1_conf	p1_dog	p2	p2_conf	p2_dog	p3	p3_conf	p3_dog
0	666020888022790149	https://pbs.twimg.com/media/CT4udn0WwAA0aMy.jpg	1	Welsh springer spaniel	0.465074	True	collie	0.156665	True	Shetland sheepdog	0.061428	True
1	666029285002620928	https://pbs.twimg.com/media/CT42GRgUYAA5iDo.jpg	1	redbone	0.506826	True	miniature pinscher	0.074192	True	Rhodesian ridgeback	0.072010	True
2	666033412701032449	https://pbs.twimg.com/media/CT4521TWwAEvMyu.jpg	1	German shepherd	0.596461	True	malinois	0.138584	True	bloodhound	0.116197	True
3	666044226329800704	https://pbs.twimg.com/media/CT5Dr8HUEAA-lEu.jpg	1	Rhodesian ridgeback	0.408143	True	redbone	0.360687	True	miniature pinscher	0.222752	True
4	666049248165822465	https://pbs.twimg.com/media/CT5IQmsXIAAKY4A.jpg	1	miniature pinscher	0.560311	True	Rottweiler	0.243682	True	Doberman	0.154629	True
5	666050758794694657	https://pbs.twimg.com/media/CT5Jof1WUAEuVxN.jpg	1	Bernese mountain dog	0.651137	True	English springer	0.263788	True	Greater Swiss Mountain dog	0.016199	True
6	666051853826850816	https://pbs.twimg.com/media/CT5KoJ1WoAAJash.jpg	1	box turtle	0.933012	False	mud turtle	0.045885	False	terrapin	0.017885	False

Figure 23 Testing result, there are no _ and - as separator

7. Several data duplication in image prediction stored with different tweet_id

```
#delete rows based on tweet id
for x in rt_tweet:
    rt_ind = img_pred_copy[img_pred_copy['tweet_id']== x].index
    img_pred_copy.drop(rt_ind, inplace=True)
```

Figure 24 Deleting duplicated row in image prediction data

Deleting duplication by using listed tweet_id from issue 5. In image prediction, testing duplication can be done by see duplication in page_url. Resulted empty series.

```
# check if it still duplication jpg_url or retweeted tweet in dataframe
img_pred_copy[img_pred_copy['jpg_url'].duplicated()]
```

tweet_id	jpg_url	img_num	p1	p1_conf	p1_dog	p2	p2_conf	p2_dog	p3	p3_conf	p3_dog

Figure 25 result of jpg_url duplication search

8. name of all dogs predictions are variate with upper and lower case

Changing all char using str.lower() to make all char only in lower case.

```
img_pred_copy['p1'] = img_pred_copy['p1'].str.lower()
img_pred_copy['p2'] = img_pred_copy['p2'].str.lower()
img_pred_copy['p3'] = img_pred_copy['p3'].str.lower()
```

Figure 26 Change value in p1,p2 and p3 into lower case using str.lower() method

```
img_pred_copy[['p1', 'p2', 'p3']]
```

	p1	p2	p3
0	welsh springer spaniel	collie	shetland sheepdog
1	redbone	miniature pinscher	rhodesian ridgeback
2	german shepherd	malinois	bloodhound
3	rhodesian ridgeback	redbone	miniature pinscher
4	miniature pinscher	rottweiler	doberman
5	bernese mountain dog	english springer	greater swiss mountain dog
6	box turtle	mud turtle	terrapin
7	chow	tibetan mastiff	fur coat
8	shopping cart	shopping basket	golden retriever
9	miniature poodle	komondor	soft coated wheaten terrier
10	golden retriever	tibetan mastiff	labrador retriever

Figure 27 Test lowercasing result by displaying value in column p1,p2, and p3

9. retweeted tweets is in data

Deleting process same as issue 7, extracting 'RT@' in text used to test the result.

```
#delete rows based on tweet id
for x in rt_tweet:
    rt_ind = json_copy[json_copy['tweet_id']== x].index
    json_copy.drop(rt_ind, inplace=True)
```

Figure 29 delete unwanted row from listed tweet_id

```
# check if it still 'RT @' in full_text
cek_rt_cp = json_copy.full_text.str.extract(r'(RT @.*)')
cek_rt_cp[cek_rt_cp[0].notnull()]
```

	0

Figure 28 Result null series from extracting 'RT @' in text column, indicating there are no duplication data anymore

Tidiness Issues

1. **timestamp has date and time in one column**
   Parsing datetime is unique. Used pd.DatetimeIndex(twit_arc_copy['timestamp']).date to get date and pd.DatetimeIndex(twit_arc_copy['timestamp']).time to get time.

```
# extract date and time from timestamp
twit_arc_copy['date'] = pd.DatetimeIndex(twit_arc_copy['timestamp']).date
twit_arc_copy['time'] = pd.DatetimeIndex(twit_arc_copy['timestamp']).time
```

Figure 30 Parsing and storing date and time into different column

```
# check date and time validity
twit_arc_copy
```

tweeted_status_timestamp	expanded_urls	rating_numerator	rating_denominator	name	doggo	floof	pupper	puppo	normalize_rating	date	time
NaT	https://twitter.com/dog_rates/status/892420643...	13	10	Phineas	None	None	None	None	1.3	2017-08-01	16:23:56
NaT	https://twitter.com/dog_rates/status/892177421...	13	10	Tilly	None	None	None	None	1.3	2017-08-01	00:17:27
NaT	https://twitter.com/dog_rates/status/891815181...	12	10	Archie	None	None	None	None	1.2	2017-07-31	00:18:03
NaT	https://twitter.com/dog_rates/status/891689557...	13	10	Darla	None	None	None	None	1.3	2017-07-30	15:58:51
NaT	https://twitter.com/dog_rates/status/891327558...	12	10	Franklin	None	None	None	None	1.2	2017-07-29	16:00:24

Figure 31Test by displaying data head

2. **text column value has two informations, there are tweet content and link**
   This process needs help from str.extract() method using unique regex to get only content and url in text. Checked by visualizing data.

```
###### extract content in each row
content_raw = twit_arc_copy.text.str.extract(r'(.*)(http|\n|$)')
```

```
# droping https that is filtered by regex
content_raw[0][464]
```

```
"Meet Strudel. He's rather h*ckin pupset that your clothes clash. 11/10 click the link to see how u can help Strudel"
```

Figure 32 Extracting only content and example of extracted value

```
# extract url in content
content_url = twit_arc_copy.text.str.extract(r'(https://t.co/\w+)\ ?(https.*)?')
```

```
# change column name
content_url.rename(columns={0:'url_content_1', 1:'url_content_2'}, inplace = True)
```

```
# check content_url
content_url
```

	url_content_1	url_content_2
0	https://t.co/MgUWQ76dJU	NaN
1	https://t.co/0Xxu71qeIV	NaN
2	https://t.co/wUnZnhtVJB	NaN
3	https://t.co/tD36da7qLQ	NaN
4	https://t.co/AtUZn91f7f	NaN
5	https://t.co/kQ04fDDRmh	NaN
6	https://t.co/Zr4hWfAs1H	https://t.co/tVJBRMnhxl
7	https://t.co/v0nONBcwxq	NaN
8	https://t.co/9TwLuAGH0b	NaN
9	https://t.co/t1bfwz5S2A	NaN
10	https://t.co/dVPW0B0Mme	NaN

Figure 33 Extracting url, and example of extracted url data

Several contents have 2 url in content, and it makes only a few rows which have value in url_content_2.

```
# overview check parsed text
twit_arc_copy[['text', 'content', 'content_url_1', 'content_url_2']]
```

	text	content	content_url_1	content_url_2
0	This is Phineas. He's a mystical boy. Only eve...	This is Phineas. He's a mystical boy. Only eve...	https://t.co/MgUWQ76dJU	NaN
1	This is Tilly. She's just checking pup on you....	This is Tilly. She's just checking pup on you....	https://t.co/0Xxu71qeIV	NaN
2	This is Archie. He is a rare Norwegian Pouncin...	This is Archie. He is a rare Norwegian Pouncin...	https://t.co/wUnZnhtVJB	NaN
3	This is Darla. She commenced a snooze mid meal...	This is Darla. She commenced a snooze mid meal...	https://t.co/tD36da7qLQ	NaN
4	This is Franklin. He would like you to stop ca...	This is Franklin. He would like you to stop ca...	https://t.co/AtUZn91f7f	NaN
5	Here we have a majestic great white breaching ...	Here we have a majestic great white breaching ...	https://t.co/kQ04fDDRmh	NaN
6	Meet Jax. He enjoys ice cream so much he gets ...	Meet Jax. He enjoys ice cream so much he gets ...	https://t.co/Zr4hWtAs1H	https://t.co/tVJBRMnhxl
7	When you watch your owner call another dog a g...	When you watch your owner call another dog a g...	https://t.co/v0nONBcwxq	NaN
8	This is Zoey. She doesn't want to be one of th...	This is Zoey. She doesn't want to be one of th...	https://t.co/9TwLuAGH0b	NaN
9	This is Cassie. She is a college pup. Studying...	This is Cassie. She is a college pup. Studying...	https://t.co/t1bfwz5S2A	NaN
10	This is Koda. He is a South Australian decksha...	This is Koda. He is a South Australian decksha...	https://t.co/dVPW0B0Mme	NaN
11	This is Bruno. He is a service shark. Only get...	This is Bruno. He is a service shark. Only get...	https://t.co/u1XPQMl29g	NaN
12	Here's a puppo that seems to be on the fence a...	Here's a puppo that seems to be on the fence a...	https://t.co/BxvuXk0UCm	NaN
13	This is Ted. He does his best. Sometimes that'...	This is Ted. He does his best. Sometimes that'...	https://t.co/f8dEDcrKSR	NaN
14	This is Stuart. He's sporting his favorite fan...	This is Stuart. He's sporting his favorite fan...	https://t.co/y70o6h3isq	NaN

Figure 34 Test result of content and url extraction

All in all, all wrangling data is finished and completing these processes with dropping unwanted column like timestamp, retweeted_timestamp, retweeted_status_id, retweeted_status_user_id in tweeter archive and full_text in json data (stored as favorite retweeted filtered). Final step is storing cleaned data into all csv files.