(2.1) Preprocessing-under 20

June 5, 2021

1 Overview

This notebook is used to preprocess the three datasets output by the notebook **Initial analysis of dataset.ipynb**, and while there are three notebooks for this preprocessing steps in total (for each age group's data file), the **relevant detailed comments are only shown in this one**, as the other two are identical in terms of the preprocessing steps.

[1]:

```
import numpy as np
import pandas as pd
import string

import nltk
nltk.download('punkt')
```

```
[nltk_data] Downloading package punkt to
[nltk_data]     /Users/aayushmarishi/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
```

[1]:

```
True
```

[2]:

```
data = pd.read_csv('../Data/blog_under_20.csv')

# Convert all texts to lower case for convenience
data['text'] = data.apply(lambda row: row.text.lower(), axis=1)
display(data)
```

```
                                                    text
0                   info has been found (+/- 100 pages,...
1                   these are the team members:   drewe...
2                   in het kader van kernfusie op aarde...
3                         testing!!!  testing!!!
4                          o= optimist p= pessimist  my...
...                                                  ...
234899                     urllink    ah jun.. &.. hung...
234900                     urllink    glen chong & kris...
234901                     urllink    ah jun~!!.. hahaz...
234902                     urllink    come~!.. let me i...
234903                     urllink    yi biao~!.. hahaz...
```

```
[234904 rows x 1 columns]
```

2 Filtering and splitting

Upon initial overview of the data files, we noticed several points that need to be preprocessed/filtered out before we carry on: 1. Some of the blog posts are not written in English. As most of the blogs are in English, it is expected that these non-English posts could have an inverse effect on the model training. Therefore, posts written in other languages should be filtered out. 2. Many of the texts in the dataset are not “clean” and they often contain the keyword “urllink”. Apparently, paragraphs that contain this keyword should also be filtered out to avoid negative impact on the modelling. Considering that the dataset is fairly large (in the sub-dataset of posts under 20, there are 234,000+ observations), filtering these paragraphs will not affect the size of the training data.

2.1 Filtering out non-English paragraphs

Since most of the blogs are written in English, a simple test is used here to determine whether a paragraph is in English: a paragraph is considered to be English if it contains the words “the” and the word “and”, which are two fairly unique words for English compared to other languages.

[3]:

```
filter_english = data.loc[data.text.str.contains("the")\
                         ].loc[data.text.str.contains("and")\
                              ].reset_index(drop=True)
```

2.2 Splitting the texts into sentences and further filtering

[4]:

```
# Split into sentences
sentence_list = []
blogs = filter_english['text'].tolist()
for b in blogs:
    # nltk.tokenize.sent_tokenize can split paragraphs into sentences
    # according to common sentence-ending punctuations
    sentences = nltk.tokenize.sent_tokenize(b)

    # Filter out sentences that include the word 'urllink'
    sentences = ['' if 'urllink' in s else s for s in sentences]

    # Remove punctuations and filter length (both a lower and upper
    # limit are imposed to make the data more consistent)
    sentences = [s.strip(string.punctuation\
                        ).strip() if (len(s)>50 and len(s) < 150\
                                     ) else '' for s in sentences]

    # Filter out empty sentences
    sentences = list(filter(None, sentences))
    sentence_list += sentences
```

```
[5]: print("There are %d sentences in total" % (len(sentence_list)))
```

```
There are 1183933 sentences in total
```

2.2.1 Example of the above for loop

In this section, one of the paragraphs is used to illustrate the preprocessing of the above block of code.

```
[6]: # Example paragraph
     display(blogs[5])
```

```
"                  i've fallen so deep, so fast. i don't know what to do with␣
 ↪myself. but i know i feel so good. i love to look at him, listen to the voice␣
 ↪of the aries. i wish i had the courage to go up to him and kiss him. i want to,␣
 ↪but yet fade away into the shadows of fear and questioning. dose aries like me?␣
 ↪how can i tell? what if he doesn't and i'm making a fool of myself, stumbling␣
 ↪over my feelings. a little girl with a crush just out of her reach. what if he␣
 ↪dose like me too? if he asks me out do i say yes? of course i do! even though␣
 ↪we will not see eachother as much as we'd like, who says it cant work! right...?
 ↪ but what if it won't work... i guess we'll have to find out...                "
```

```
[7]: # Use the nltk function to split the example paragraph into sentences
     sentences = nltk.tokenize.sent_tokenize(blogs[5])
     display(sentences)
```

```
["                  i've fallen so deep, so fast.",
 "i don't know what to do with myself.",
 'but i know i feel so good.',
 'i love to look at him, listen to the voice of the aries.',
 'i wish i had the courage to go up to him and kiss him.',
 'i want to, but yet fade away into the shadows of fear and questioning.',
 'dose aries like me?',
 'how can i tell?',
 "what if he doesn't and i'm making a fool of myself, stumbling over my feelings.",
 'a little girl with a crush just out of her reach.',
 'what if he dose like me too?',
 'if he asks me out do i say yes?',
 'of course i do!',
 "even though we will not see eachother as much as we'd like, who says it cant␣
 ↪work!",
 'right...?',
 "but what if it won't work... i guess we'll have to find out..."]
```

```
[8]: # Strip the sentences of their punctuations and add a length filter
     sentences = [s.strip(string.punctuation\
                         ).strip() if (len(s)>50 and len(s) < 150\
                                      ) else '' for s in sentences]
```

```
display(sentences)
```

```
['',
 '',
 '',
 'i love to look at him, listen to the voice of the aries',
 'i wish i had the courage to go up to him and kiss him',
 'i want to, but yet fade away into the shadows of fear and questioning',
 '',
 '',
 "what if he doesn't and i'm making a fool of myself, stumbling over my feelings",
 '',
 '',
 '',
 '',
 "even though we will not see eachother as much as we'd like, who says it cant␣
 ↪work",
 '',
 "but what if it won't work... i guess we'll have to find out"]
```

[9]:

```
# Delete sentences that have been turned into empty strings
sentences = list(filter(None, sentences))
display(sentences)
```

```
['i love to look at him, listen to the voice of the aries',
 'i wish i had the courage to go up to him and kiss him',
 'i want to, but yet fade away into the shadows of fear and questioning',
 "what if he doesn't and i'm making a fool of myself, stumbling over my feelings",
 "even though we will not see eachother as much as we'd like, who says it cant␣
 ↪work",
 "but what if it won't work... i guess we'll have to find out"]
```

As the output of the last cell shows, the preprocessing step splits the paragraph into sentences, remove punctuations, and in the end only keep sentences with certain lengths.

3 Output

[12]:

```
df = pd.DataFrame(sentence_list)
print('There are %d sentences (observations) after the preprocessing'\
      %df.shape[0])
```

```
There are 1183933 sentences (observations) after the preprocessing
```

[14]:

```
sample = df.sample(n = 10000).reset_index(drop=True)
sample.to_csv('../Data/sample_under_20.csv', index=False)
```

[15]:

```
sample2 = df.sample(n = 5000).reset_index(drop=True)
sample2.to_csv('../Data/sample2_under_20.csv', index=False)
```