Preprocessing

December 6, 2021

Table of Contents

1. Data download

Note: Output of the cell has been removed since it is irrelevant to the task and would take up too much unnecessary space if shown.

[ ]:
```
!wget https://ai.stanford.edu/~amaas/data/sentiment/aclImdb_v1.tar.gz
!tar -xzf aclImdb_v1.tar.gz
```

2. Combine datasets

[ ]:
```
import os, glob
import numpy as np
import pandas as pd
```

2.1 Train positive text files

[ ]:
```
%%time
files_folder = "aclImdb/train/pos"

# Read all .txt files in this folder and perform some preprocessing by removing␣
 ↪"\n" and "\t"
```

```
train_pos_files = [open(file, encoding="utf8").read().replace("\n", " ").
 ↪replace("\t", " ") for
                   file in glob.glob(os.path.join(files_folder ,"*.txt"))]

# Put the texts into dataframes
train_pos_files = [pd.DataFrame(file, index=[0], columns=['Comment']) for file␣
 ↪in train_pos_files]
```

```
Wall time: 3.17 s
```

[ ]:
```
# Concatenate all the dataframes
train_pos = pd.concat(train_pos_files, ignore_index=True)

# Create a column for these comments' labels
train_pos['Sentiment'] = 1

print(f'Number of positive comments in the train set: {train_pos.shape[0]}')
print('Preview of the dataframe:\n')
train_pos.head(5)
```

```
Number of positive comments in the train set: 12500
Preview of the dataframe:
```

[ ]:

	Comment	Sentiment
0	Bromwell High is a cartoon comedy. It ran at t...	1
1	Homelessness (or Houselessness as George Carli...	1
2	Brilliant over-acting by Lesley Ann Warren. Be...	1
3	This is easily the most underrated film inn th...	1
4	This is not the typical Mel Brooks film. It wa...	1

2.2 Train negative text files

[ ]:
```
%%time
files_folder = "aclImdb/train/neg"

# Read all .txt files in this folder and perform some preprocessing by removing␣
 ↪"\n" and "\t"
train_neg_files = [open(file, encoding="utf8").read().replace("\n", " ").
 ↪replace("\t", " ") for
                   file in glob.glob(os.path.join(files_folder ,"*.txt"))]

# Put the texts into dataframes
train_neg_files = [pd.DataFrame(file, index=[0], columns=['Comment']) for file␣
 ↪in train_neg_files]

# Concatenate all the dataframes
train_neg = pd.concat(train_neg_files, ignore_index=True)
```

```
# Create a column for these comments' labels
train_neg['Sentiment'] = 0

print(f'Number of negative comments in the train set: {train_neg.shape[0]}')
print('Preview of the dataframe:\n')
train_neg.head(5)
```

```
Number of negative comments in the train set: 12500
Preview of the dataframe:

Wall time: 3.65 s
```

[ ]:

	Comment	Sentiment
0	Story of a man who has unnatural feelings for ...	0
1	Airport '77 starts as a brand new luxury 747 p...	0
2	This film lacked something I couldn't put my f...	0
3	Sorry everyone,,, I know this is supposed to b...	0
4	When I was little my parents took me along to ...	0

2.3 Test positive text files

[ ]:
```
%%time
files_folder = "aclImdb/test/pos"

# Read all .txt files in this folder and perform some preprocessing by removing␣
 ↪"\n" and "\t"
test_pos_files = [open(file, encoding="utf8").read().replace("\n", " ").
 ↪replace("\t", " ") for
                  file in glob.glob(os.path.join(files_folder ,"*.txt"))]

# Put the texts into dataframes
test_pos_files = [pd.DataFrame(file, index=[0], columns=['Comment']) for file in␣
 ↪test_pos_files]

# Concatenate all the dataframes
test_pos = pd.concat(test_pos_files, ignore_index=True)

# Create a column for these comments' labels
test_pos['Sentiment'] = 1

print(f'Number of positive comments in the test set: {test_pos.shape[0]}')
print('Preview of the dataframe:\n')
test_pos.head(5)
```

```
Number of positive comments in the test set: 12500
Preview of the dataframe:
```

```
Wall time: 3.68 s
```

[ ]:

	Comment	Sentiment
0	I went and saw this movie last night after bei...	1
1	Actor turned director Bill Paxton follows up h...	1
2	As a recreational golfer with some knowledge o...	1
3	I saw this film in a sneak preview, and it is ...	1
4	Bill Paxton has taken the true story of the 19...	1

2.4 Test negative text files

[ ]:
```
%%time
files_folder = "aclImdb/test/neg"

# Read all .txt files in this folder and perform some preprocessing by removing␣
 ↪"\n" and "\t"
test_neg_files = [open(file, encoding="utf8").read().replace("\n", " ").
 ↪replace("\t", " ") for
                  file in glob.glob(os.path.join(files_folder ,"*.txt"))]

# Put the texts into dataframes
test_neg_files = [pd.DataFrame(file, index=[0], columns=['Comment']) for file in␣
 ↪test_neg_files]

# Concatenate all the dataframes
test_neg = pd.concat(test_neg_files, ignore_index=True)

# Create a column for these comments' labels
test_neg['Sentiment'] = 0

print(f'Number of negative comments in the test set: {test_neg.shape[0]}')
print('Preview of the dataframe:\n')
test_neg.head(5)
```

```
Number of negative comments in the test set: 12500
Preview of the dataframe:

Wall time: 4.07 s
```

[ ]:

	Comment	Sentiment
0	Once again Mr. Costner has dragged out a movie...	0
1	This is an example of why the majority of acti...	0
2	First of all I hate those moronic rappers, who...	0
3	Not even the Beatles could write songs everyon...	0
4	Brass pictures (movies is not a fitting word f...	0

3. Further Preprocessing

3.1 Convert texts to lower case

Stopwords can be better removed if all the texts are in lower case format.

```
[ ]: train_pos['Comment'] = train_pos['Comment'].str.lower()
     train_neg['Comment'] = train_neg['Comment'].str.lower()

     test_pos['Comment'] = test_pos['Comment'].str.lower()
     test_neg['Comment'] = test_neg['Comment'].str.lower()
```

3.2 Remove the html tags in the texts

```
[ ]: train_pos = train_pos.replace(to_replace=r'<.*?>', value=' ', regex=True)
     train_neg = train_neg.replace(to_replace=r'<.*?>', value=' ', regex=True)

     test_pos = test_pos.replace(to_replace=r'<.*?>', value=' ', regex=True)
     test_neg = test_neg.replace(to_replace=r'<.*?>', value=' ', regex=True)
```

Note: To avoid confusion, some preprocessing steps that are specific to certain models are shown in the corresponding notebooks. These preprocessings include steps such as removal of stop words and tokenization.