B455Project4RobertKellems

April 18, 2022

The script below was run locally on my computer to generate the CSV files used later in this notebook. It iterates through all of the reviews in the "train" and "test" folders from the dataset and puts them in a CSV along with their sentiment classification based on which folder they were located in.

[ ]:
```
import os
import csv
import pandas as pd

training = pd.DataFrame()
for file in os.listdir('train/pos'):
    with open('train/pos/'+file, 'r', encoding='utf-8') as f:
        txt = f.read()
        training = training.append([[txt, 1]], ignore_index=True)
for file in os.listdir('train/neg'):
    with open('train/neg/'+file, 'r', encoding='utf-8') as f:
        txt = f.read()
        training = training.append([[txt, 0]], ignore_index=True)

training.to_csv('training.csv', index=False, encoding='utf-8')

testing = pd.DataFrame()
for file in os.listdir('test/pos'):
    with open('test/pos/'+file, 'r', encoding='utf-8') as f:
        txt = f.read()
        testing = testing.append([[txt, 1]], ignore_index=True)
for file in os.listdir('test/neg'):
    with open('test/neg/'+file, 'r', encoding='utf-8') as f:
        txt = f.read()
        testing = testing.append([[txt, 0]], ignore_index=True)

testing.to_csv('test.csv', index=False, encoding='utf-8')
```

[26]:
```
import pandas as pd
import re
import numpy as np

from nltk.stem.porter import PorterStemmer
```

```
from sklearn.pipeline import Pipeline
from sklearn.linear_model import LogisticRegression
from sklearn.feature_extraction.text import TfidfVectorizer
from warnings import simplefilter
from sklearn.exceptions import ConvergenceWarning
```

Here we import two CSV files which I have made locally on my computer that contain each review and its corresponding sentiment. We then combine them into one DataFrame in order to go about preparing the text for use in the algorithm.

[32]:
```
train = pd.read_csv('training.csv')
test = pd.read_csv('test.csv')
df = pd.concat((train,test)).reset_index()
df = df.drop(df.columns[0], axis=1)
print(df)
```

```
                                                       0  1
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
…                                                    … ..
49995  I occasionally let my kids watch this garbage …  0
49996  When all we have anymore is pretty much realit…  0
49997  The basic genre is a thriller intercut with an…  0
49998  Four things intrigued me as to this film - fir…  0
49999  David Bryce's comments nearby are exceptionall…  0

[50000 rows x 2 columns]
```

This function takes a string and removes all unnecessary icons and groups different forms of a word before recombining all of the words into a single string which is then returned. Based on code from Python Machine Learning, 2nd Edition, by Sebastian Raschka and Vahid Mirjalili, PACKT books, 2017.

[23]:
```
def textPrep(text):
  #cleaning the text of unnecessary icons
  text = re.sub('<[^>]*>', '', text)
  emoticons = re.findall('(?::|;|=)(?:-)?(?:\)|\(|D|P)', text)
  text = (re.sub('[\W]+', ' ', text.lower()) + ' '.join(emoticons).replace('-',␣
 ↪''))

  #grouping different forms of one word using Porter Stemmer algorithm
  porter = PorterStemmer()
  text = " ".join([porter.stem(w) for w in text.split()])
  return text
```

Now we apply the function to each review in our dataset.

[33]:
```
df['0'] = df['0'].apply(textPrep)
print(df)
```

```
                                                       0  1
0      bromwel high is a cartoon comedi it ran at the…  1
1      homeless or houseless as georg carlin state ha…  1
2      brilliant over act by lesley ann warren best d…  1
3      thi is easili the most underr film inn the bro…  1
4      thi is not the typic mel brook film it wa much…  1
…                                                    … ..
49995  i occasion let my kid watch thi garbag so they…  0
49996  when all we have anymor is pretti much realiti…  0
49997  the basic genr is a thriller intercut with an …  0
49998  four thing intrigu me as to thi film firstli i…  0
49999  david bryce s comment nearbi are except well w…  0

[50000 rows x 2 columns]
```

Now we begin using a logistic regression model to classify reviews using 5-fold cross validation. For each split of the data we use term frequency-inverse document frequency in order to transform the reviews into feature vectors where the frequency of each word is weighted depending on relevancy. These vectors are then fed into our logistic regression model for training. The test accuracy of this model is typically around 89%.

[31]:
```
simplefilter("ignore", category=ConvergenceWarning)

#creating 5 random subsets
subsets = []
newData = df
for i in range(5):
  sample = newData.sample(n = 10000) #n = 50000/5
  subsets += [sample]
  newData = newData.drop(sample.index)

modelPerformance = []

#begin testing
for i, fold in enumerate(subsets):
  testData = fold
  trainData = df.drop(testData.index)

  xTrain = trainData.drop(trainData.columns[1], axis=1)
  yTrain = trainData[trainData.columns[1]]
  xTest = testData.drop(testData.columns[1], axis=1)
  yTest = testData[testData.columns[1]]

  xTrain = xTrain.to_numpy().flatten()
  yTrain = yTrain.to_numpy()
```

```
  xTest = xTest.to_numpy().flatten()
  yTest = yTest.to_numpy()

  #initializing object which will be used to transform reviews into
  #feature vectors reflecting each word's relevancy
  tfidf = TfidfVectorizer(strip_accents=None, lowercase=False,␣
↪preprocessor=None)

  model = Pipeline([('vect', tfidf), ('clf', LogisticRegression())])
  model.fit(xTrain, yTrain)
  modelPerformance += [model.score(xTest, yTest)]
  print("Fold " + str(i+1) + " complete")


print("Logistic regression performance for each fold:",modelPerformance)
print("Average:", sum(modelPerformance)/len(modelPerformance))
```

```
Fold 1 complete
Fold 2 complete
Fold 3 complete
Fold 4 complete
Fold 5 complete
Logistic regression performance for each fold: [0.8909, 0.8989, 0.8977, 0.8935,
0.8936]
Average: 0.8949199999999999
```

The code below is for converting the notebook to PDF.

[ ]:
```
!wget -nc https://raw.githubusercontent.com/brpy/colab-pdf/master/colab_pdf.py
from colab_pdf import colab_pdf
colab_pdf('B455Project4RobertKellems.ipynb')
```

```
File ‘colab_pdf.py’ already there; not retrieving.


WARNING: apt does not have a stable CLI interface. Use with caution in scripts.
```