CodingPortion

April 4, 2022

0.0.1 HW #3 Coding Portion - Sentiment of a Movie Title.

[198]:
```
# Import the data.
import pandas as pd
training_data = pd.read_csv('./train.tsv', sep='\t')
test_data = pd.read_csv('./test.tsv', sep='\t')
dataset = pd.concat([training_data, test_data], ignore_index=True)
dataset
#training_data
```

[198]:

	PhraseId	SentenceId	Phrase	Sentiment
0	1	1	A series of escapades demonstrating the adage …	1.0
1	2	1	A series of escapades demonstrating the adage …	2.0
2	3	1	A series	2.0
3	4	1	A	2.0
4	5	1	series	2.0
…	…	…	…	…
222347	222348	11855	A long-winded , predictable scenario .	NaN
222348	222349	11855	A long-winded , predictable scenario	NaN
222349	222350	11855	A long-winded ,	NaN
222350	222351	11855	A long-winded	NaN
222351	222352	11855	predictable scenario	NaN

[222352 rows x 4 columns]

[203]:
```
# Make a BOW for training data.
from sklearn.feature_extraction.text import CountVectorizer
from nltk.tokenize import RegexpTokenizer
tokenizer = RegexpTokenizer(r'[a-zA-Z0-9]+')
vectorizer = CountVectorizer(tokenizer = tokenizer.tokenize, ngram_range=(1,1),␣
 ↪stop_words='english')
training_counts = vectorizer.fit_transform(training_data['Phrase'])
#print(training_counts)
training_counts.shape
```

[203]:
```
(156060, 14988)
```

[204]:
```
# Make a BOW for testing data.
from sklearn.feature_extraction.text import CountVectorizer
from nltk.tokenize import RegexpTokenizer
test_counts = vectorizer.transform(test_data['Phrase'])
#print(test_counts)
test_counts.shape
```

[204]:
```
(66292, 14988)
```

[213]:
```
# Lets see how good this is.
# Note: The highly specific test_size is to make the sizes of the data line up,␣
 ↪as this code kept screaming at me for not having data of a similar size. You␣
 ↪can replicate this by changing the test size to literally anything else. If␣
 ↪you have any feedback on what I should be doing here instead, that would be␣
 ↪much appreciated!
from sklearn.model_selection import train_test_split
X_train, X_test, Y_train, Y_test = train_test_split(training_counts,␣
 ↪training_data['Sentiment'], test_size=0.4247853389, random_state=123)
# Compile the model.
from sklearn.naive_bayes import MultinomialNB
thingy = MultinomialNB() # I know what this is doing I just wanted a funny name␣
 ↪for the variable.
thingy.fit(X_train, Y_train)
```

[213]:
```
MultinomialNB()
```

[214]:
```
# Lets put this against the test data.
from sklearn import metrics
expected = thingy.predict(test_counts)
score = metrics.accuracy_score(expected, Y_test)
print("Prediction Accuracy:", str('{:04.2f}'.format(score*100))+'%')
```

```
Prediction Accuracy: 39.90%
```