Brian_Reppeto_DSC550_Week_3

March 31, 2024

0.0.1 DSC 550 Week :

Activity 3.2

Author: Brian Reppeto 3/25/2024

[55]:
```
# import libraries

import pandas as pd
from textblob import TextBlob
import re
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.stem import PorterStemmer
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfVectorizer
from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer
from sklearn.metrics import accuracy_score
```

[5]:
```
# import movie file

data_path='labeledTrainData.tsv'
movie_df=pd.read_csv(data_path, delimiter='\t')
```

[6]:
```
# head new df

movie_df.head()
```

[6]:

	id	sentiment	review
0	5814_8	1	With all this stuff going down at the moment w…
1	2381_9	1	\The Classic War of the Worlds\" by Timothy Hi…
2	7759_3	0	The film starts with a manager (Nicholas Bell)…
3	3630_4	0	It must be assumed that those who praised this…
4	9495_8	1	Superbly trashy and wondrously unpretentious 8…

[7]:
```
# How many of each positive and negative reviews are there?

review_counts=movie_df['sentiment'].value_counts()
```

```
print(review_counts)
```

```
sentiment
1    12500
0    12500
Name: count, dtype: int64
```

[8]:
```
# create function to classify sentiment

def classify_review(review):
    blob=TextBlob(review)
    return 'positive' if blob.sentiment.polarity >= 0 else 'negative'
```

[9]:
```
# apply function to each review

movie_df['predicted_sentiment']=movie_df['review'].apply(classify_review)
```

[10]:
```
# display the first few rows to verify

print(movie_df[['review', 'predicted_sentiment']].head())
```

```
                                              review predicted_sentiment
0  With all this stuff going down at the moment w…            positive
1  \The Classic War of the Worlds\" by Timothy Hi…            positive
2  The film starts with a manager (Nicholas Bell)…            negative
3  It must be assumed that those who praised this…            positive
4  Superbly trashy and wondrously unpretentious 8…            negative
```

[11]:
```
# create function to change the calc sent to number

def classify_review(review):
    blob=TextBlob(review)
    return 1 if blob.sentiment.polarity >= 0 else 0
```

[12]:
```
# apply function to each review

movie_df['predicted_sentiment_num']=movie_df['review'].apply(classify_review)
```

[13]:
```
# head df

movie_df.head()
```

[13]:

	id	sentiment	review	\
0	5814_8	1	With all this stuff going down at the moment w…	
1	2381_9	1	\The Classic War of the Worlds\" by Timothy Hi…	
2	7759_3	0	The film starts with a manager (Nicholas Bell)…	
3	3630_4	0	It must be assumed that those who praised this…	

```
4  9495_8          1  Superbly trashy and wondrously unpretentious 8…
```

```
  predicted_sentiment  predicted_sentiment_num
0            positive                        1
1            positive                        1
2            negative                        0
3            positive                        1
4            negative                        0
```

[80]:
```
# find the matches and non matches

match=movie_df['sentiment'] == movie_df['predicted_sentiment_num']

review_comp=match.value_counts()

print(review_comp)
```

```
True     17131
False     7869
Name: count, dtype: int64
```

[66]:
```
# calc accuracy

acc=accuracy_score(movie_df['sentiment'], movie_df['predicted_sentiment_num'])

print(f"Percent match:",acc)
```

```
Percent match: 0.68524
```

based on the calculated matches there is a greater accuracy vs a random guess.

Extra Credit

[67]:
```
# initialize VADER sentiment intensity analyzer

analyzer=SentimentIntensityAnalyzer()

# function to classify sentiment based on VADER scores

def classify_sentiment(review_text):
    vs=analyzer.polarity_scores(review_text)
    return 1 if vs['compound']>=0 else 0

# apply sentiment classification to the review texts

movie_df['predicted_sentiment_var']=movie_df['review'].apply(classify_sentiment)

# calc accuracy
```

```
accuracy=accuracy_score(movie_df['sentiment'],␣
 ↪movie_df['predicted_sentiment_var'])


print(f"Vader accuracy:",accuracy)
```

```
Vader accuracy: 0.65284
```

based on the calculated matches there is a greater accuracy vs a random guess.

Part 2 Prep Text

[14]:
```
# convert text to lowercase

movie_df['review']=movie_df['review'].str.lower()
```

[15]:
```
# head column

movie_df['review'].head(15)
```

[15]:
```
0     with all this stuff going down at the moment w…
1     \the classic war of the worlds\" by timothy hi…
2     the film starts with a manager (nicholas bell)…
3     it must be assumed that those who praised this…
4     superbly trashy and wondrously unpretentious 8…
5     i dont know why people think this is such a ba…
6     this movie could have been very good, but come…
7     i watched this video at a friend's house. i'm …
8     a friend of mine bought this film for £1, and …
9     <br /><br />this movie is full of references. …
10    what happens when an army of wetbacks, towelhe…
11    although i generally do not like remakes belie…
12    \mr. harvey lights a candle\" is anchored by a…
13    i had a feeling that after \submerged\", this …
14    note to george litman, and others: the mystery…
Name: review, dtype: object
```

[23]:
```
# Remove punctuation and special characters

movie_df['review']=movie_df['review'].apply(lambda x: re.sub(r'[^\w\s]', '', x)␣
 ↪if pd.notnull(x) else x)
```

[24]:
```
# head df

movie_df.head(15)
```

[24]:
```
         id  sentiment                                             review  \
0    5814_8          1  with all this stuff going down at the moment w…
```

```
1    2381_9          1  the classic war of the worlds by timothy hines…
2    7759_3          0  the film starts with a manager nicholas bell g…
3    3630_4          0  it must be assumed that those who praised this…
4    9495_8          1  superbly trashy and wondrously unpretentious 8…
5    8196_8          1  i dont know why people think this is such a ba…
6    7166_2          0  this movie could have been very good but comes…
7   10633_1          0  i watched this video at a friends house im gla…
8     319_1          0  a friend of mine bought this film for 1 and ev…
9   8713_10          1  br br this movie is full of references like ma…
10   2486_3          0  what happens when an army of wetbacks towelhea…
11  6811_10          1  although i generally do not like remakes belie…
12  11744_9          1  mr harvey lights a candle is anchored by a bri…
13   7369_1          0  i had a feeling that after submerged this one …
14  12081_1          0  note to george litman and others the mystery s…

   predicted_sentiment  predicted_sentiment_num
0             positive                        1
1             positive                        1
2             negative                        0
3             positive                        1
4             negative                        0
5             positive                        1
6             negative                        0
7             positive                        1
8             positive                        1
9             positive                        1
10            negative                        0
11            positive                        1
12            positive                        1
13            positive                        1
14            positive                        1
```

[26]:
```
# Remove stop words

# download stop words

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

```
[nltk_data] Downloading package punkt to
[nltk_data]     /Users/brianreppeto/nltk_data…
[nltk_data]   Unzipping tokenizers/punkt.zip.
[nltk_data] Downloading package stopwords to
[nltk_data]     /Users/brianreppeto/nltk_data…
[nltk_data]   Unzipping corpora/stopwords.zip.
```

[26]:
```
True
```

[27]:
```
# load stop words

stop_words = set(stopwords.words('english'))
```

[28]:
```
# function to remove stopwords

def remove_stopwords(text):
    if pd.notnull(text):
        # Tokenize the text string
        word_tokens = word_tokenize(text)
        # Remove stop words
        filtered_sentence = [word for word in word_tokens if word.lower() not␣
 ↪in stop_words]
        # Rejoin words
        return ' '.join(filtered_sentence)
    return text

# apply function to remove stop words from the column

movie_df['review'] = movie_df['review'].apply(remove_stopwords)
```

[29]:
```
# head df

movie_df.head()
```

[29]:

	id	sentiment	review	predicted_sentiment	predicted_sentiment_num
0	5814_8	1	stuff going moment mj ive started listening mu…	positive	1
1	2381_9	1	classic war worlds timothy hines entertaining …	positive	1
2	7759_3	0	film starts manager nicholas bell giving welco…	negative	0
3	3630_4	0	must assumed praised film greatest filmed oper…	positive	1
4	9495_8	1	superbly trashy wondrously unpretentious 80s e…	negative	0

[33]:
```
# apply NLTK porterstemmer

# initialize the porterstemmer

stemmer = PorterStemmer()
```

[35]:
```
# function to stem words in the text
```

```
def stem_text(text):
    if pd.notnull(text):

        # tokenize the text string into words

        word_tokens = word_tokenize(text)

        # stem each word

        stemmed_words = [stemmer.stem(word) for word in word_tokens]

        # rejoin the stemmed words into a single string

        return ' '.join(stemmed_words)
    return text


movie_df['review'] = movie_df['review'].apply(stem_text)
```

[36]:
```
# head df

movie_df.head(15)
```

[36]:

	id	sentiment	review	predicted_sentiment	predicted_sentiment_num
0	5814_8	1	stuff go moment mj ive start listen music watc…	positive	1
1	2381_9	1	classic war world timothi hine entertain film …	positive	1
2	7759_3	0	film start manag nichola bell give welcom inve…	negative	0
3	3630_4	0	must assum prais film greatest film opera ever…	positive	1
4	9495_8	1	superbl trashi wondrous unpretenti 80 exploit …	negative	0
5	8196_8	1	dont know peopl think bad movi got pretti good…	positive	1
6	7166_2	0	movi could good come way short cheesi special …		
7	10633_1	0	watch video friend hous im glad wast money buy…		
8	319_1	0	friend mine bought film 1 even grossli overpr …		
9	8713_10	1	br br movi full refer like mad max ii wild one…		
10	2486_3	0	happen armi wetback towelhead godless eastern …		
11	6811_10	1	although gener like remak believ remak wast ti…		
12	11744_9	1	mr harvey light candl anchor brilliant perform…		
13	7369_1	0	feel submerg one wouldnt better right must loo…		
14	12081_1	0	note georg litman other mysteri scienc theater…		

6	negative	0
7	positive	1
8	positive	1
9	positive	1
10	negative	0
11	positive	1
12	positive	1
13	positive	1
14	positive	1

[49]:
```
# initialize the CountVectorizer

vectorizer = CountVectorizer()

# fit and transform

bow_matrix = vectorizer.fit_transform(movie_df['review'])

# BOW Dimension

print(f"BOW Dimension:", bow_matrix.shape)
```

```
BOW Dimension: (25000, 92528)
```

[50]:
```
# create a ttf-idf

# initialize the TfidfVectorizer

tfidf_vectorizer = TfidfVectorizer()

# fit and transform the stemmed text to create the TF-IDF matrix

tfidf_matrix = tfidf_vectorizer.fit_transform(movie_df['review'])

# display the dimensions of the TF-IDF matrix

print(f"Dimensions of the TF-IDF matrix:", tfidf_matrix.shape)
```

```
Dimensions of the TF-IDF matrix: (25000, 92528)
```

[ ]:
```

```