
Question 1

- parse data line :

Method to take data from data line separate index 1 and 2 TO text and sentiment label and create a tuple out of it.

- preprocess :

Separated the words by spaces using split method into a list of array items.

Literally so excited I'm going to a Sam Smith concert in October

[Literally,so,excited,I'm,going,to,a,Sam,Smith,concert,in,October]

Question 2

Create a temporary dictionary to implement the idea of bag of words that holds the keys as the words and values as the count of the words, loop through the tokens, count it and add it to the dictionary.

The idea of Bag of words

Use this temporary dictionary to update the global dictionary.

```
{'song': 1,  
 '25th': 1,  
 'play-through': 1}
```

b) Alternate is to use CountVectorizer() and tfidf Vectorizer which is implemented in v3 of notebook , in the below table the accuracy improvement is also highlighted by the use of tfidf vectorizer.

Question 3

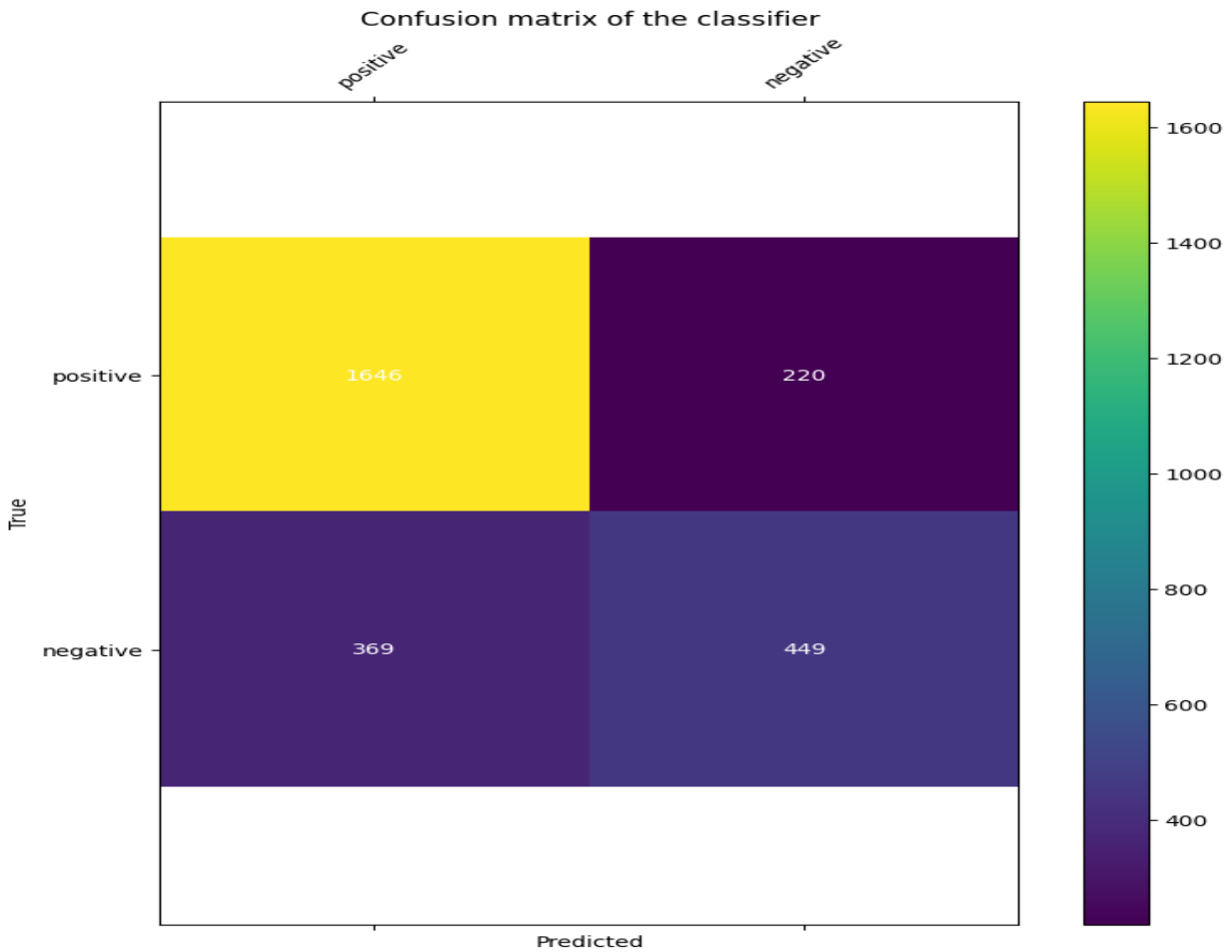
For 10 fold ten iterations are done in the outer loop , inner loop implements partitioning into 9 training set and 1 test set , starting from the first set being the test set for the first iteration to the last set being test set for the last iteration.

Question 4

There are more false negatives compared to false positives the ratio is almost doubled, its getting confused because it cannot identify negative sentiments due to lack of negative classes

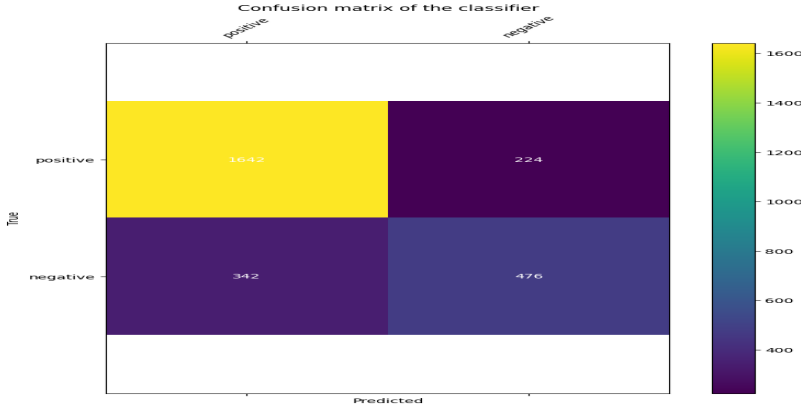
p 21932 n 11608

Data shows that there are 21932 positive classes and 11608 negative classes , so there is an imbalance in class training , which is reflected in class prediction



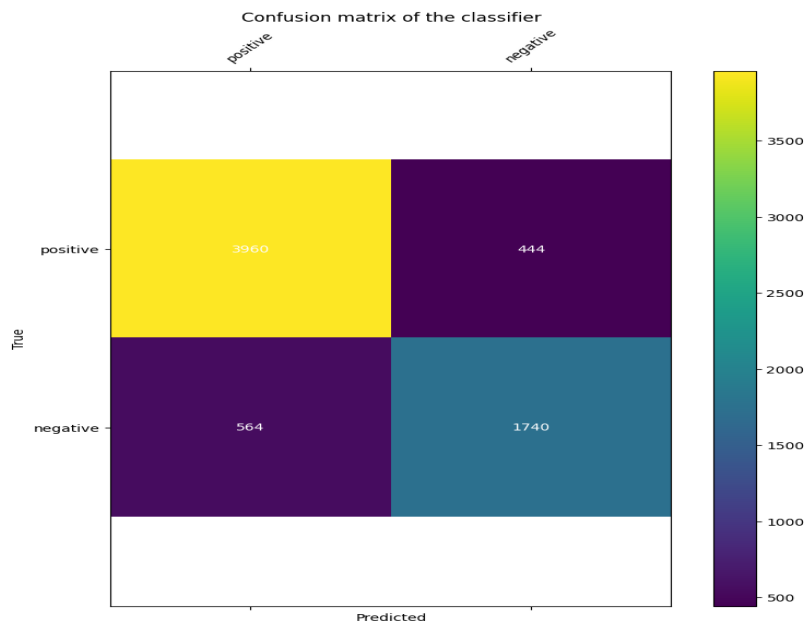
Question 5

Parse Data Line	<pre>text = data_line[2] label = data_line[1]</pre>		
First Implementation			
Preprocess:Tokenize	Preprocess:Features	Classifier	Cross Validation
Tokens obtained using split by spaces	Dictionary created by word count of each word in the sentence	Linear SVC	Precision: 0.816 Accuracy: 0.78 Recall: 0.88 F1: 0.84
Confusion Matrix:			

Cross Validation: Performed with 10 fold cross validation using 9 partitions as training set and 1 partition as testing set.												
Conclusion: The Linear SVC model performed quite well for basic processing , it was able to classify sentences as positive and negative sentiment with upto 78% accuracy.												
Second Implementation												
Preprocess:Tokenize	Preprocess:Features	Classifier	Cross Validation									
Used library word tokenize, performed lemmatization on tokens and	Dictionary created by word count of each word in the sentence	Linear SVC	Precision: 0.82 Accuracy: 0.789 Recall: 0.87 F1: 0.85									
Confusion Matrix:												
<div><p>Confusion matrix of the classifier</p><table border="1"><thead><tr><th></th><th>Actual positive</th><th>Actual negative</th></tr></thead><tbody><tr><th>Predicted positive</th><td>1642</td><td>224</td></tr><tr><th>Predicted negative</th><td>342</td><td>476</td></tr></tbody></table></div>					Actual positive	Actual negative	Predicted positive	1642	224	Predicted negative	342	476
	Actual positive	Actual negative										
Predicted positive	1642	224										
Predicted negative	342	476										
Conclusion:Observed a drastic drop in false negatives, with slight increase in the accuracy of from 0.780 to 0.789.												

Third Implementation			
Preprocess:Tokenize	Preprocess:Features	Classifier	Cross Validation
No change	Using the CountVectorizer and Tfidf Vectorizer to extract features.	Linear SVC, Naive Bayes	Count Vectorizer: Accuracy: 0.85 Tfidf Vectorizer: Precision: 0.816 Accuracy: 0.869 Recall: 0.88 F1: 0.84

Confusion Matrix:



For NaiveBayes and LinearSVC

Accuracy Naive Bayes: 0.843768, Accuracy LinearSVC: 0.84973

Conclusion: Model performed better ,Accuracy increased from 0.78 to 0.85 for CountVectorizer and 0.869 for tfidf vectorizer.

**BigramCollocationFinder used here was taken from stackoverflow code.

<https://stackoverflow.com/questions/21844546/forming-bigrams-of-words-in-list-of-sentences-with-python>

Additional Learning:

Sparse and Dense Arrays, Handling of sparse arrays with csr matrix