Name – PRABHAKAR KUMAR

Roll – IRM2017008

Assignment 5 (Naïve Bayes Spam Classifier)

Niave Bayes algorithm is a well-known Supervised classification algorithm based on Bayes Theorem. It is very fast and well suited for purpose of text classification. For text classification some well known applications may include, classification of product reviews into positive and negative reviews, classification of emails into spam or ham, which is done as a part of this assignment.

The Naïve Bayes algorithm, works around calculation of conditional probability which can be done using several methods because of which we can have majorly three different types of implementations of Naïve Bayes :

1. Multinomial Naïve Bayes
2. Bernoulli Naïve Bayes
3. Gaussian Naïve Bayes

Multinomial Naïve Bayes

In my first program I have implemented the Multinomial Naïve Bayes which is generally used when we have discrete values of features, which here I took as the count of a word in a sentence. It is used when the classification classes are more than or equal to two.

Here as per the lectures, I have also applied Laplase Smoothing, to count for words that are not there in any email, since without that our predictions may go completely wrong. The overall performance of the Multinomial Naïve Bayes algorithm is shown in the image below:

```
        if actual[i]!=predicted[i]and actual[i]==1:
            fn+=1

    print("CONFUSION MATRIX")
    print(tp," ",fp)
    print(fn," ",tn)
```

```
CONFUSION MATRIX
1043   401
0   228
```

```
[ ] print("Accuracy : ",str((tp+tn)/(tp+fp+tn+fn)))
    print("Precision : ",str((tp)/(tp+fp)))
    print("Sensitivity : ",str((tp)/(tp+fn)))
    print("Specificity : ",str((tn)/(fp+tn)))
```

```
Accuracy :  0.7601674641148325
Precision :  0.7222991689750693
Sensitivity :  1.0
Specificity :  0.3624801271860095
```

```
[ ]
```

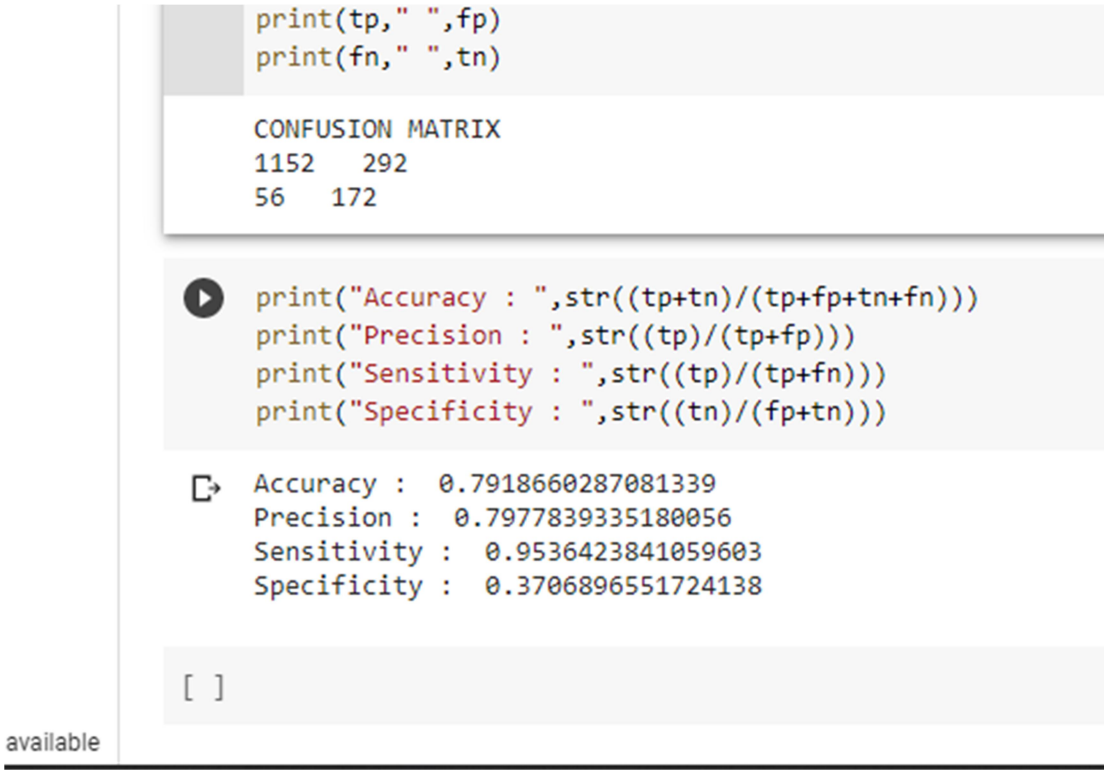
Bernoulli Naïve Bayes

In my second program I have implemented the Bernoulli Naïve Bayes algorithm. Here we assume that the input dataset is following a Bernoulli Distribution, which means that the values are either 0 or 1. Hence to convert the emails to a Bernoulli dataset, I first took all the words present in the training dataset, removed all the stop words, as their presence does not affect the model rather just slows it down, and then for each of the left word I mark whether the word is there in the email or not. If present then it is marked as 1 else 0, which then follows the Bernoulli distribution.

$$P(x_i \mid y) = P(i \mid y)x_i + (1 - P(i \mid y))(1 - x_i)$$

The above formula is used for calculation of word probability given a class, and unlike the Multinomial where we just ignore the non-presence of a word in an email, in Bernoulli we explicitly penalize the non-occurrence of a word in the

email. Here also I have used Laplase Smoothing to avoid the probabilities to get to 0. The Bernoulli is especially useful for text classification, where there are only two classes, like here in our example spam and ham, and cant be used for text classification where there are more than two classes, like good, bad , average etc. The following figure shows the output of the model performance:



Gaussian Naïve Bayes

Gaussian Naïve Bayes is built on the assumption that the input parameters follow a continuous range and follow a normal distribution of probabilities. Since the email which are lists of tokens of words, it could not be transformed into any Gaussian Distribution, hence it was not feasible to make a Gaussian Naïve Bayes algorithm for the purpose of Email Spam and Ham classifier.

Overall, we can see that the overall accuracy of the Bernoulli Model was better than that of the Multinomial Model, while there was only a little much that the Multinomial was better in terms of Specificity and Sensitivity. But possible reason for this can be that Bernoulli is especially useful for text classification

for Binary classes and also while using Bernoulli Naïve Bayes algorithm, we had removed all the stop words. This step resulted in reduction of the complexity of running the algorithm, without affecting the performance.