**ASSIGNMENT-1**

**Sentiment Analysis using Naïve Bayes**

GROUP MEMBERS:

VATSAL GUPTA: 2018A7PS0198H

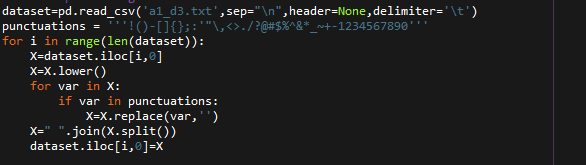
MIHIR BANSAL: 2018A7PS0215H

BANDARU ABHINAV: 2018A7PS0236H

**Design Procedure**

The procedure applies naïve Bayes method to classify the string into two classes. The whole code is divided into 4 main parts.

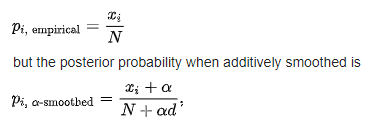
* Firstly, the pre-processing which converts the text into lower case and also removes punctuations. Since removal of there punctuations will also lead to multiple spaces, this is also handled using X=” ”.join(X.split())



* Secondly, finding the unique words from the training data. Since we are using 5-fold cross validation method, we will iterate with our model 5 times, each having a new set of Unique word (as each word is taken from 0.8 of the total dataset).
* Variable uniqueword is a Dictionary which stores the unique word in the training data with the frequency of their occurrence.
* Variable Listofdicti is a list of those unique words.



* Thirdly, we will calculate the probability of it being a 1 or a 0 by our standard naïve Bayes algorithm. (We are considering that all the features are independent of each other). Here we have also put to the test Laplacian smoothing

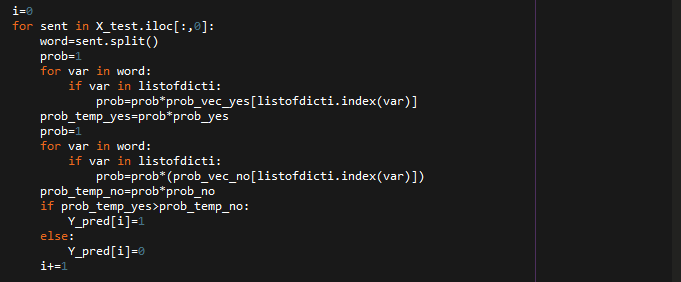


We have taken α to be 1 and d to be 2 (where d generally represents the no of classes in the dataset).



* The variable X\_t is a matrix which has the shape length of the training data \* no of unique words. It stores weather that string contains that particular word or not.
* The vector prob\_vec\_yes stores the probability of occurrences of a word given that the target variable is 1. similarly, prob\_vec\_no stores the occurrences of a word given that the target variable is 0.

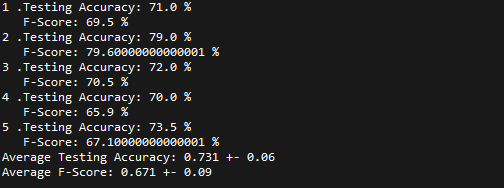
With the help of all this the following code does the required job for us storing the predicted value in Y\_pred.



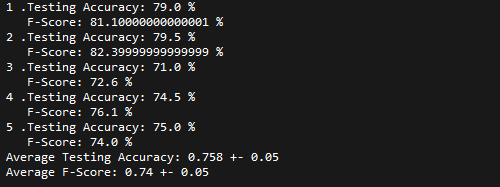
* Fourthly, the accuracy and F-score, which were calculated by matching Y\_pred and Y\_test, and the results were printed in the console.

**Results**

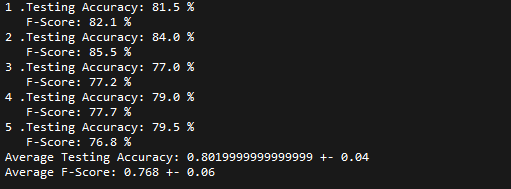
1. Results Without applying Laplacian Smoothing and Without removal of Stopwords



1. With the removal of Stopwords another model can be created. Its accuracy and F-Score is as follows: (here there is smoothing applied)



1. Considering Laplacian Smoothing and no Stopwords Removal.



**Conclusion**

As it can be seen from the results above shown the smoothing has helped us to improve our accuracy. This is mainly because many times some words might not occur at all in one sentiment but this does not mean its probability will be 0, yes it would be really less. In order to fix this error smoothing has helped, which increases the probability of some 0 probabilities to some very low value. It can also be seen that removal of Stopwords has decreased the probability, this could be because of the fact that removal of Stopwords also remove some words such as “no”, ”not”, ”should”, “i”, etc ; which cold be related to a sentiment and helps in predicting it. As these words occur very frequently in the text, its affect is also seen in the results.

Hence a model with smoothing and not removal of Stopwords helps us to reach an accuracy of about 80%.