Code Description:

Dataset Used:

We used the datasets given i.e naïve_bayes_data.txt file.

Pre-processing:

We read the text file line by line in Python. Each line has a label i.e **pos** or **neg.** So, every line is stored in a 2D array. Every entry of this array is an array containing the label of the lin and words in that line.

Procedure for Sentiment Analysis Using Naïve Bayes Algorithm:

Naive Bayes algorithm is based on **Bayes' Theorem**. It is not a single algorithm but a family of algorithms where all of them share a common principle, i.e. every pair of features being classified is independent of each other.

First we read data line by line from the text file and stored it in a 2D array. Now, we split the data i.e the 2D array into 2 parts i.e Testing and Training Data. We select random entries and put them into training and testing dataset.
After this, we consider training data set for building our model. We iterate through all the entries in the training dataset and get the count of positive and negative sentences. While storing data in 2D array, we created a dictionary named bag_of_words to keep track of all the unique datasets in the in the training dataset.
Now, for each and every word in the training dataset, we get the count of number of times that word occurs in the positive examples and the number of times it occurs in the negative examples. If this count is less than 10, we consider this as less frequent word and remove that word from the bag_of_words.
Applying Bayes formula for multiple events, we get probabilities of the words in Training dataset.
Now, we read inputs from the Testing dataset. For each and every word in a sentence, we calculate POS and NEG i.e
POS= (Number of times word occurs in positive examples)/(Number of positive words+ Number of words in bag_of_words) and

NEG= (Number of times word occurs in negative examples)/(Number of negative words+ Number of words in bag_of_words).
Now, if we get POS>NEG, we classify that example as **pos** and if NEG>POS, its classified as **neg.**
After getting the predicted and actual results, we make comparisons and generate the Confusion Matrix and get the values of necessary scores.

Results:

This is one of the observations:

Confusion Matrix:

	Actual	pos	neg
Predicted			
pos		**482**	**122**
Neg		**656**	**1123**

True Positive=482

True Negative=1123

False Positive=122

False Negative=656

We calculated Accuracy, Precision, Recall and F1 Score using below formulas:

$Accuracy = (TP + TN) \div (TP + TN + FP + FN)$ = 0.67

$Precision = (TP) \div (TP + FP)$ = 0.798

$Recall = (TP) \div (TP + FN)$ = 0.42

$F1\ Score = 2 * (Precision * Recall) \div (Precision + Recall)$ = 0.55