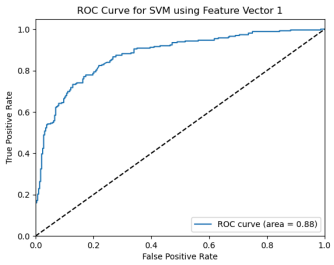
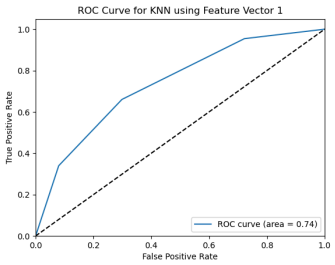
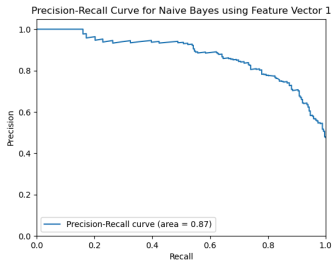
Result

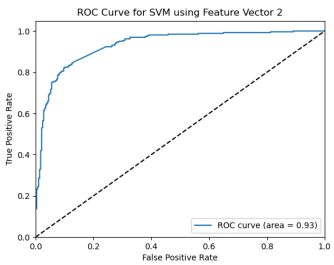
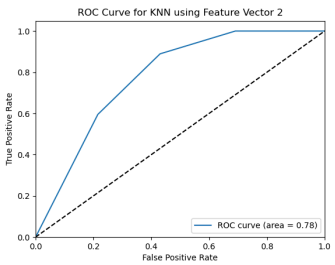
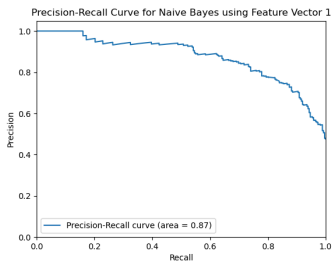
Feature Vector	Classifier	Accuracy	Precision (0)	Precision (1)	Recall (0)	Recall (1)	F1-Score (0)	F1-Score (1)	Support (0)	Support (1)	Training Time (s)	Prediction Time (s)	AUC (ROC)
1	KNN	0.68	0.69	0.67	0.70	0.66	0.70	0.66	288	262	0.0020	0.0554	0.74
	SVM	0.80	0.82	0.78	0.79	0.81	0.80	0.79	288	262	1.1283	0.0426	0.88
	Naive Bayes	0.78	0.80	0.75	0.76	0.79	0.78	0.77	288	262	0.0021	0.0002	NA
2	KNN	0.72	0.85	0.65	0.57	0.89	0.68	0.75	288	262	0.0010	0.0645	0.78
	SVM	0.86	0.85	0.87	0.89	0.83	0.87	0.85	288	262	0.4109	0.0191	0.93
	Naive Bayes	0.83	0.94	0.76	0.73	0.95	0.82	0.84	288	262	0.0020	0.0002	NA

ROC and Precision Recall Curve

For FV1 - Original feature vector

		
ROC_Curve_for_SVM_using_Feature Vector 1	ROC_Curve_for_KNN_using_Feature Vector 1	Precision_Recall_Curve_for_Naive Bayes_using_Feature Vector 1

For FV2 - pared-down feature vector by using TF-IDF with 512 vector size

		
ROC_Curve_for_SVM_using_Feature Vector 2	ROC_Curve_for_KNN_using_Feature Vector 2	Precision_Recall_Curve_for_Naive Bayes_using_Feature Vector 2

Additional Comment

1. During the initial stage of Text Processing and Feature Extraction for Sentiment Analysis, I removed punctuation and digits, which further enhanced my feature vector FV1. This preprocessing step helped to refine the quality of the features and contributed to the overall effectiveness of the sentiment analysis model.
2. I attempted to implement the KNN algorithm, but encountered challenges due to its feasibility with large datasets. For instance, when dealing with a dataset comprising 3,000 test samples and 3,000 training samples, the sheer volume of distance calculations required renders KNN impractical. With approximately 9,000,000 individual computations needed, the computational complexity becomes prohibitive. Consequently, I opted to utilize the SKlearn API as a more efficient alternative. The related K-NN code is included in the lab1.ipynb.

Summary

In the comparative study between different classifiers using two sets of feature vectors, we have observed notable differences in both the efficiency of training/prediction times and the effectiveness of the classifiers based on accuracy and other performance metrics.

Efficiency Analysis

When considering the offline efficiency cost, which refers to the time taken to build the classifier model, we see a clear distinction between the classifiers. The KNN classifier demonstrates the fastest training times across both feature vectors, which is expected due to the simplicity of its training process — essentially just storing the training data. On the other hand, SVM takes longer to train, particularly with Feature Vector 1, due to the complexity of finding the optimal hyperplane. Naive Bayes consistently shows low training times, slightly faster than KNN, which can be attributed to the simplicity of its probability-based learning mechanism.

For the online efficiency cost, or the time taken to classify a new tuple, KNN has longer prediction times compared to the other classifiers, which is a characteristic of lazy learning algorithms that defer computation until classification. SVM and Naive Bayes provide very fast predictions, with Naive Bayes slightly edging out due to the straightforward computation of posterior probabilities.

Effectiveness Analysis

The accuracy and ROC curves provide insight into the effectiveness of each classifier. With Feature Vector 1, SVM leads with an accuracy of 0.80, followed closely by Naive Bayes at 0.78 and KNN at 0.68. The ROC curves support these findings, with SVM achieving an AUC of 0.88, indicating a high true positive rate across various thresholds. KNN's AUC of 0.74 is respectable but suggests room for improvement.

Feature Vector 2, which underwent feature selection, shows an interesting shift. SVM still performs the best with an improved accuracy of 0.86 and an impressive AUC of 0.93. Naive Bayes follows with an accuracy of 0.83 and KNN improves to 0.72. The Precision-Recall curve for Naive Bayes using Feature Vector 2 exhibits a high area under the curve (AUC) of 0.92, indicating that the classifier maintains a high precision across different recall levels.

Feature Selection Impact

The adoption of feature selection methodology has evidently impacted the classifiers' performance. For SVM and Naive Bayes, there is a clear improvement in accuracy when moving from the original feature vector (FV1) to the pared-down feature vector (FV2). This

suggests that the removal of less informative features helped the classifiers focus on the most relevant aspects of the data, thus improving the decision-making process.
For KNN, while there is an improvement in accuracy from FV1 to FV2, the change is less dramatic. This could imply that KNN is less sensitive to the presence of irrelevant features or that the nature of the feature interaction in the dataset benefits more from the comprehensive feature set provided in FV1.

Conclusion

The experiments suggest that feature selection can lead to better classifier performance, as seen with SVM and Naive Bayes. KNN, while less affected by feature selection, still shows improvement. In terms of efficiency, KNN is preferred for faster model building but not for prediction tasks, where SVM and Naive Bayes excel. The visualizations provide compelling evidence that SVM is the most effective classifier in this study, with the highest AUC and consistent improvement in accuracy after feature selection.