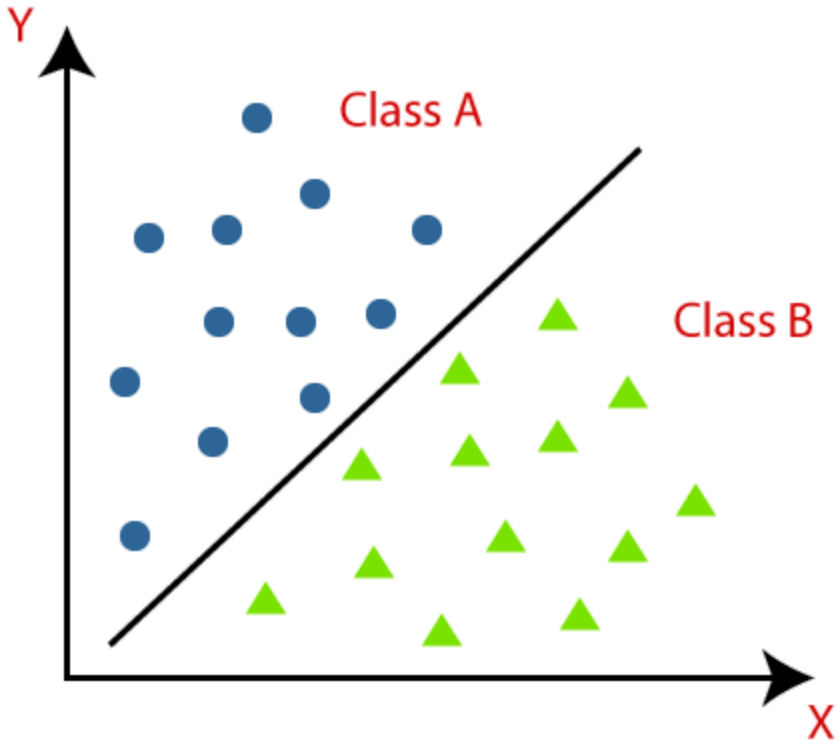
Question #1 Report

Implementation of various probabilistic classifiers



Abhinav Khanna(2017CSB1071)

CS524: Data Mining

INTRODUCTION

In the area of probabilistic classification, we have discussed three main algorithms: Full Bayes Classifier, Naive Bayes Classifier and K-Nearest Neighbors. The purpose of this project is to implement these algorithms on various datasets, analyze their space and time complexity, and conclude which algorithm is better for which dataset. The scope of this project is limited by the hardware capabilities, such as RAM, number of cores/processors, and CPU clock cycles, availability and usage of GPUs et al.

THEORETICAL FOUNDATION

Bayes classifier directly uses the Bayes theorem to predict the class for a new test instance. It uses the same theoretical foundation as used in Gaussian Mixture model classifiers. Thus, the formulas of mean, covariance matrix and prior probability stay the same as the GMM classifier.

Naive Bayesian Classification works the same way as the Full Bayesian Classifier, with the added assumption that all the attributes are independent of each other. Thus, the only difference that we need to know is that all the entries except the diagonal elements are zero in the covariance matrix.

Note that in both Full Bayesian and Naive Bayesian classifiers, our prediction is that class which has the highest probability.

K-Nearest Neighbor follows the principle of allocating the class which occurs most often in a point K nearest neighbors.

IMPLEMENTATION

Before implementing any algorithm, I have divided the dataset Credit Card fraud into 80% training and 20% testing data.

- ### Bayes Classification

  Means, Covariance matrices and Prior probabilities are calculated using the theoretical foundation mentioned above. For prediction, we need to use a gaussian function, and for that, the numpy library for multivariate gaussian

distribution has to be used.

- ### Naive Bayes Classification

  The same implementation as the Bayes Classification is done, with the only modification that all entries in the covariance matrix apart from the diagonal entries are set to zero, and the same procedure as the previous is used.

- ### K-Nearest Neighbor

  There is no prior training in this algorithm. For each point in the test dataset, its distance from all the points in the training dataset is computed. Since the dataset is huge, a library function **sklearn.metrics.pairwise.euclidean_distances** is used. This helps in speeding up the testing process. Once an array consisting of all the distances of all training points is obtained, this array is then sorted, and the index of the K nearest points is used to assign a class label to the test point, as explained earlier.

OUTPUTS AND OBSERVATIONS

It is crucial to note that the data is highly skewed. There are many more “0”s present than “1”s.

Algorithm	Misclassified “0”s %	Misclassified “1”s %
Bayes Classifier	2.36%	7.69%
Naive Bayes Classifier	3.19%	8.28%
KNN (K = 5)	0.07%	12.7%

# of training data points: 227808

# of test data points: 56999

Time Taken:

Bayes Classifier: < 10 sec

Naive Bayes Classifier: < 10 sec

KNN: ~20 mins

CONCLUSION

It is clear due to the heavily skewed data, the error rate is much higher in “1”s than “0”s. It is also evident that in these types of datasets, Bayes and Naive Bayes are much better choices than Naive Bayes. They have a lesser misclassification rate of “0”s than “1”s, and more importantly, take much less time than KNN. Another observation is that Bayes and Naive Bayes have almost the same performance, meaning that the assumption of attributes being independent is correct to a great extent.

REFERENCES

1. Data Mining and Analysis: Fundamental Concepts and Algorithms: Textbook by Mohammed J. Zaki and Wagner Meira
2. Class notes