CSE - 4255 Data Mining and Warehousing Lab

Comparison Between the Performance of Decision Tree and Naive Bayes Classifier in Classification

Saif Mahmud
Roll: SH - 54

M. Tanjid Hasan Tonmoy
Roll: SH - 09

Submitted To:

Dr. Chowdhury Farhan Ahmed
Professor

&

Abu Ahmed Ferdaus
Associate Professor

Department of Computer Science and Engineering
University of Dhaka

September 30, 2019

1 Problem Definition

In this experiment, we have implemented two different classification algorithms, namely Decision tree and Naive Bayes. The algorithms utilize discrete and continuous features to predict class labels. Comparative analysis of these two algorithms have been conducted using various evaluation metrics for both balanced and imbalanced datasets of varied sizes.

2 Theory

2.1 Decision Tree

Decision tree learning is a supervised machine learning technique for inducing a decision tree from training data. A decision tree (also referred to as a classification tree or a reduction tree) is a predictive model which is a mapping from observations about an item to conclusions about its target value. In the tree structures, leaves represent classifications (also referred to as labels), non-leaf nodes are features, and branches represent conjunctions of features that lead to the classifications.

The algorithm is invoked with three parameters: D, attribute list, and Attribute selection method. We refer to D as a data partition. Initially, it is the complete set of training tuples and their associated class labels. The parameter attribute list is a list of attributes describing the tuples. Attribute selection method specifies a heuristic procedure for selecting the attribute that 'best' discriminates the given tuples according to class. This procedure employs an attribute selection measure such as information gain or the Gini index. Whether the tree is strictly binary is generally driven by the attribute selection measure. Some attribute selection measures, such as the Gini index, enforce the resulting tree to be binary. Others, like information gain, do not, therein allowing multi-way splits (i.e., two or more branches to be grown from a node).

However, When a decision tree is built, many of the branches will reflect anomalies in the training data due to noise or outliers. Tree pruning methods address this problem of overfitting the data. Such methods typically use statistical measures to remove the least-reliable branches. Pruned trees tend to be smaller and less complex and, thus, easier to comprehend.

2.2 Naive Bayes Classifier

The Naive Bayesian classifier is based on Bayes theorem with the independence assumptions between predictors. A Naive Bayesian model is easy to build, with no complicated iterative parameter estimation which makes it particularly useful for very large datasets. Despite its simplicity, the Naive Bayesian classifier often does surprisingly well and is widely used because it often outperforms more sophisticated classification methods. Naive Bayesian classifiers assume that the effect of an attribute value on a given class is independent of the values of the other attributes. This assumption is called class-conditional independence. It is made to simplify the computations involved and, in this sense, is considered 'naive'.

The goal of any probabilistic classifier is, with features x_0 through x_n and classes c_0 through c_k, to determine the probability of the features occurring in each class, and to return the most likely class. Therefore, for each class, we want to be able to calculate P(c_i — x_0, , x_n). In order to do this, we use Bayes rule. It may be noted that Bayes rule is the following:

$$P(A \mid B) = \frac{P(B \mid A)\, P(A)}{P(B)}$$

2.3 K - Fold Cross Validation

In k-fold cross-validation, the initial data are randomly partitioned into k mutually exclusive subsets or 'folds', D_1 , D_2 , . . . , D_k , each of approximately equal size. Training and testing is performed k times. In iteration i, partition D_i is reserved as the test set, and the remaining partitions are collectively used to train the model. That is, in the first iteration, subsets D_2 , . . . , D_k collectively serve as the training set to obtain a first model, which is tested on D_1 ; the second iteration is trained on subsets D_1 , D_3 , . . . , D_k and tested on D_2 ; and so on. Unlike the holdout and random subsampling methods, here each sample is used the same number of times for training and once for testing. For classification, the accuracy estimate is the overall number of correct classifications from the k iterations, divided by the total number of tuples in the initial data.

3 Experimental Setup

3.1 Implementation

For the implementation of decision tree, two different attribute selection methods (entropy and Gini index) have been used for both discrete and continuous attributes and is available as option for training models. hen there is a large number of distinct values for a continuous attribute, the training time increases significantly due to the fact that all possible splitting points have to be considered. The tree is stored using a dictionary structure in python and built recursively. Prepruning of the tree based on a threshold given as input has been used to prevent over-fitting.

In case of implementing Naive Bayes classifier, the prior and posterior probability is stored in the dictionary data structure in the training phase in order to ensure constant time retrieval during the prediction phase. The maximum likelihood estimates of probability for Bayesian rule is based on count in the case of categorical data and on the other hand, probability computed from Gaussian distribution in case of continuous data. To eradicate the zero probability due to sparse data, Laplace correction has been deployed in posterior probability calculation.

4 Result

4.1 Evaluation Metric

The standard accuracy metric defined in (1) is used to evaluate the models used in this experiment.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

where, TP, TN, FP and FN are defined as True Positive, True Negative, False Positive and False Negative respectively. In this regard, true positive rate (TPR) or Recall or Sensitivity is defined as per the expression defined in (2).

$$TPR/Recall/Sensitivity = \frac{TP}{TP + FN} \tag{2}$$

where, TP and FN are defined as True Positive and False Negative respectively.

On the other hand, Specificity and False Positive Rate (FPR) is specified as per (3) and (4) respectively.

$$Specificity = \frac{TN}{TN + FP} \tag{3}$$

$$FPR = 1 - Specificity = \frac{TP}{TP + FN} \tag{4}$$

Table 1: Accuracy : Decision Tree

Dataset	**No of tuples**	**Accuracy % [Entropy]**
iris	150	92.6607
mushroom	8124	98.5227
breast cancer	286	67.5955
chess	3196	66.0425
connect-4	67557	65.8303
pendigits	7494	19.469
poker	1000000	50.1209
creditcarddefault	30000	81.9945
annealing	798	77.2503

Table 2: Accuracy : Naive Bayes Classifier

Dataset	**Dataset Size**	**k-Fold Cross Validation (k = 5) Accuracy (%)**	**Avg. Accuracy (%)**
Adult	32561	82.9879	83.35124
		83.2463	
		83.4459	
		83.6302	
		83.4459	
Breast Cancer	286	86.2069	73.38174
		82.4561	
		64.9123	
		64.9123	
		68.4211	
Census-Income	199523	85.6939	85.57612
		85.255	
		85.5654	
		85.8235	
		85.5428	
Chess	3196	89.8438	87.63982
		86.25	
		88.1064	
		88.7324	
		85.2665	
Chess - II	28056	36.2989	36.2274
		36.545	
		36.1255	
		36.3361	
		35.8315	
Connect-4	67557	72.1137	72.15832
		72.7353	
		72.2617	
		71.7786	
		71.9023	
Credit Card Default	30000	67.2055	65.12984
		70.4167	
		53.0167	
		73.2167	
		61.7936	
Iris	150	93.3333	95.33334
	6	90	
		96.6667	
		96.6667	
		100	

Dataset	Dataset Size	k-Fold Cross Validation (k = 5) Accuracy (%)	Avg. Accuracy (%)
Mushroom	8124	95.326	95.37192
		95.0154	
		94.4	
		95.6281	
		96.4901	
Pendigits	7494	78.7333	78.0755
		78.6	
		77.7333	
		77.5183	
		77.7926	
Poker	1000000	50.119	50.1187
		50.114	
		50.12	
		50.1205	
		50.12	
Breast Cancer - Wisconsin	699	65.035	62.78208
		63.8298	
		62.4113	
		60.8696	
		61.7647	

Table 3: Dataset : Adult

Class Label	Precision	Recall	F1-Score
<=50K	0.85	0.93	0.89
>50K	0.7	0.5	0.58
Avg	0.82	0.83	0.82

Table 4: Dataset : Breast Cancer

Class Label	Precision	Recall	F1-Score
no-recurrence-events	0.75	0.82	0.79
recurrence-events	0.46	0.35	0.4
Avg	0.66	0.68	0.67

Table 5: Dataset : PenDigit

Class Label	Precision	Recall	F1-Score
0	0.96	0.93	0.94
1	0.77	0.83	0.80
2	0.90	0.93	0.91
3	0.81	0.92	0.86
4	0.00	0.00	0.00
5	0.58	0.60	0.59
6	0.99	0.97	0.98
7	0.99	0.90	0.94
8	0.82	0.92	0.87
9	0.48	0.90	0.63
avg	0.73	0.79	0.75

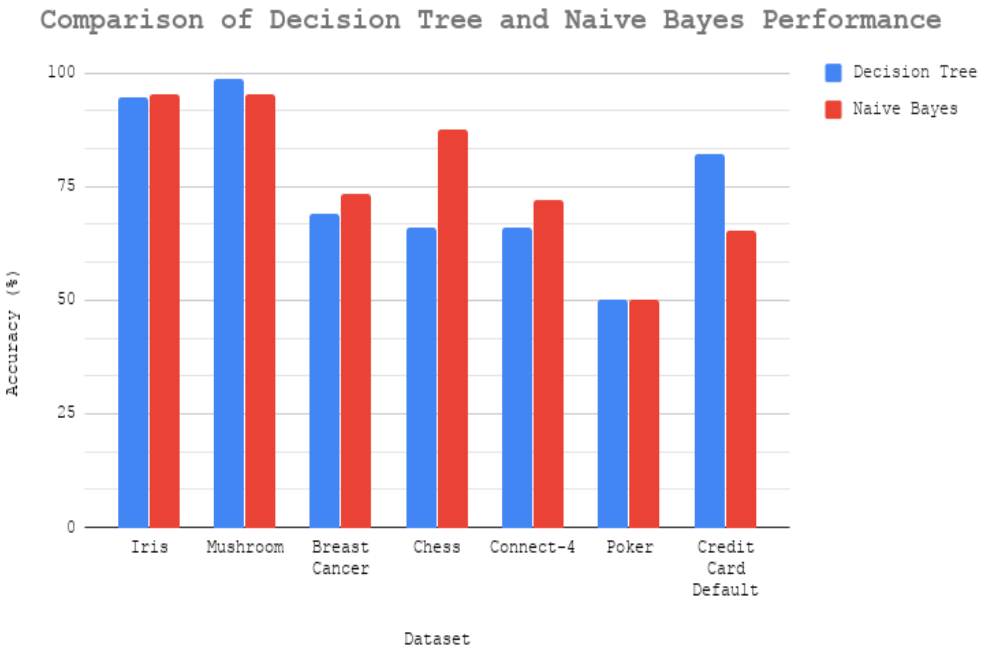

Figure 1: Comparison between Decision Tree and Naive Bayes

5 Discussion

Both of these algorithms produce reasonable performance when dealing with moderate sized datasets with close to balanced class distribution. However, in case of class imbalance, both of thee algorithms suffer. The insights we have obtained through the experiment on aforementioned classification algorithm is given below:

- In case of highly imabalanced dataset with very few positive or negative example, the classifier overfits the dominant class in most of the cases. Therefore, we can see in Figure 2 that only the dominant class has been detected correctly in the diagonal position with very few precise classification for the scarce class labels.

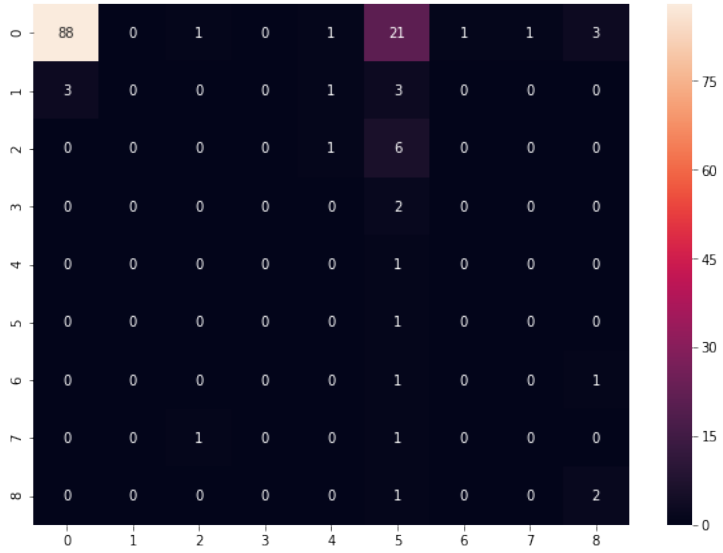

Figure 2: Confusion Matrix - Breast Cancer Wisconsin

- In the scenario of multi-class classification with many classes in comparison with the number of tuples in the dataset, both decision tree and naive bayes classifier tends to fail in segmenting with higher accuracy which is depicted in Figure 3.

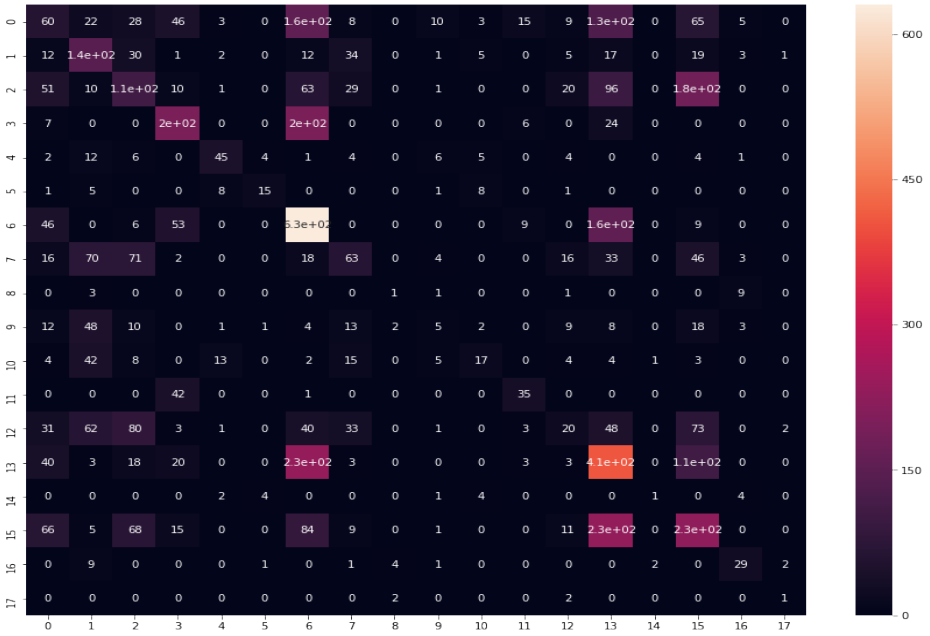

Figure 3: Confusion Matrix - Chess II

- With balanced distribution of multiple classes and reasonable amount of training tuples representing dataset size, the classification algorithms performs reasonably well which is evident in the diagonal entries of Figure 4.

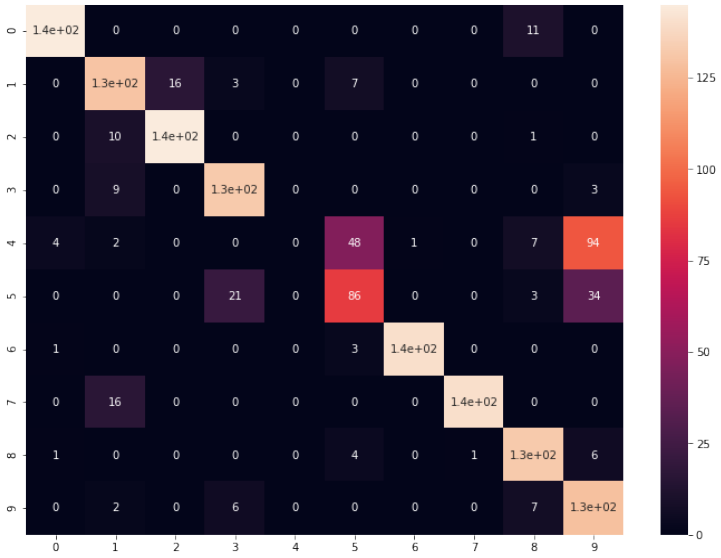

Figure 4: Confusion Matrix - PenDigit