Selective Bayesian Classifier: Feature Selection for the Naïve Bayesian Classifier Using Decision Trees

Chotirat "Ann" Ratanamahatana, Dimitrios Gunopulos
Department of Computer Science,
University of California, Riverside, USA.

Abstract

It is known that Naïve Bayesian classifier (NB) works very well on some domains, and poorly on some. The performance of NB suffers in domains that involve correlated features. C4.5 decision trees, on the other hand, typically perform better than the Naïve Bayesian algorithm on such domains. This paper describes a Selective Bayesian classifier (SBC) that simply uses only those features that C4.5 would use in its decision tree when learning a small example of a training set, a combination of the two different natures of classifiers. Experiments conducted on eleven datasets indicate that SBC performs reliably better than NB on all domains, and SBC outperforms C4.5 on many datasets of which C4.5 outperform NB. SBC also can eliminate, on most cases, more than half of the original attributes, which can greatly reduce the size of the training and test data, as well as the running time. Further, the SBC algorithm typically learns faster than both C4.5 and NB, needing fewer training examples to reach high accuracy of classification.

1 Introduction

Two of the most widely used and successful methods of classification are C4.5 decision trees [9] and Naïve Bayesian learning (NB) [2]. While C4.5 constructs decision trees by using features to try and split the training set into positive and negative examples until it achieves high accuracy on the training set, NB represents each class with a probabilistic summary, and finds the most likely class for each example it is asked to classify.

Several researchers have emphasized on the issue of redundant attributes, as well as advantages of feature selection for the Naïve Bayesian Classifier, not only for induction learning. Pazzani [8] explores the methods of joining two (or more) related attributes into a new compound attribute where the attribute dependencies are present. Another method, Boosting on Naïve Bayesian classifier [3] has been experimented by applying series of classifiers to the problem and paying more attention to the examples misclassified by its predecessor. However, it was shown that it fails on average in a set of natural domain [7]. Augmented Bayesian Classifiers [5] is another approach where Naïve Bayes is augmented by the addition of correlation arcs between attributes. Langley and Sage [6], on the other hand, use a wrapper approach for the subset selection to only select relevant features for NB.

It has been shown that Naïve Bayesian classifier is extremely effective in practice and difficult to systematically improve upon [1]. In this paper, we show that it is possible to reliably improve this classifier by using a feature selection method. Naïve Bayes can suffer from oversensitivity to redundant and/or irrelevant attributes. If two or more attributes are highly correlated, they receive too much weight in the final decision as to which class an example belongs to. This leads to a decline in accuracy of prediction in domains with correlated features. C4.5 does not suffer from this problem because if two attributes are correlated, it will not be possible to use both of them to split the training set, since this would lead to exactly the same split, which makes no difference to the existing tree. This is one of the main reasons C4.5 performs better than NB on domains with correlated attributes.

We conjecture that the performance of NB improves if it uses only those features that C4.5 used in constructing its decision tree. This method of feature selection would also perform well and learn quickly, that is, it would need fewer training examples to reach high classification accuracy.

We present experimental evidence that this method of feature selection leads to improved performance of the Naïve Bayesian Classifier, especially in the domains where Naïve Bayes performs not as well as C4.5. We analyze the behavior on ten domains from the UCI repository. The experimental results justify our expectation. We also tested SBC on another sufficiently large synthetic dataset and our algorithm appeared to scale nicely. Our Selective Bayesian Classifier always outperforms NB and performs as well as, or better than C4.5 on almost all the domains.

2 Naïve Bayesian Classifier

2.1 Description and Problems

The Naïve Bayesian classifier is a straightforward and frequently used method for supervised learning. It provides a flexible way for dealing with any number of attributes or classes, and is based on probability theory (Bayes' rule). It is the asymptotically fastest learning algorithm that examines all its training input. It has been demonstrated to perform surprisingly well in a very wide variety of problems in spite of the simplistic nature of the model. Furthermore, small amounts of bad data, or “noise,” do not perturb the results by much.

However, there are two central assumptions in Naïve Bayesian classification. First, the classification assumes that the elements of each class can be assigned on probability measurement, and that the measurement is sufficient to classify the element into exactly one class. This assumption entails that the classes can be differentiated only by means of the attribute values. The dependence on this type of differentiation is related to the idea of linear separability; therefore, Naïve Bayesian classification may not easily learn or predict complicated Boolean relations.

The other assumption is that given a particular class membership, the probabilities of particular attributes having particular values are independent of each other. However, this assumption is often violated in reality.

A plausible assumption of independence is computationally problematic. This is best described by redundant attributes. If we posit two independent features, and a third which is redundant (i.e. perfectly correlated) with the first, the first attribute will have twice as much influence on the expression as the second has, which is a strength not reflected in reality. The increased strength of the first attribute increases the possibility of unwanted bias in the classification. Even with this independence assumption, Hand and Yu illustrated that Naïve Bayesian classification still works well in practice [4]. However, this paper shows that if those redundant attributes are eliminated, the performance of Naïve Bayesian classifier can significantly increase.

3 C4.5 Decision Trees

Decision trees are one of the most popular methods used for inductive inference. They are robust for noisy data and capable of learning disjunctive expressions. A decision tree is a k-ary tree where each of the internal nodes specifies a test on some attributes from the input feature set used to represent the data. Each branch descending from a node corresponds to one of the possible values of the feature specified at that node. And each test results in branches, which represent different outcomes of the test.

The algorithm starts with the entire set of tuples in the training set, selects the *best* attribute that yields maximum information for classification, and generates a test node for this attribute. Then, top down induction of decision trees divides the current set of tuples according to their values of the current test attribute. Classifier generation stops, if all tuples in a subset belong to the same class, or if it is not worth to proceed with an additional separation into further subsets, i.e. if further attribute tests yield only information for classification below a pre-specified threshold.

Decision tree algorithm uses an entropy-based measure known as “information gain” as a heuristic for selecting the attribute that will best split the training data into separate classes. Its algorithm computes the information gain of each attribute, and in each round, the one with the highest information gain will be chosen as the test attribute for the given set of training data. A well-chosen split point should help in splitting the data to the best possible extent.

After all, a main criterion in the greedy decision tree approach is to build *shorter* trees. The best split point can be easily evaluated by considering each unique value for that feature in the given data as a possible split point and calculating the associated information gain.

A simple decision tree algorithm only selects one decision tree given an example set, though there may be many different trees consistent with the data. The information gain measure (implemented in ID3 decision trees) is biased in that it tends to prefer attributes with many values rather than those with few values. C4.5 suppresses this bias by using an alternative measure called *Information Gain Ratio* , which considers the probability of each attribute value. This removes the bias of information gain towards features with many values.

3.1 Tree Pruning

C4.5 builds a tree so that most of the training examples are classified correctly. Though this approach is correct when there is no noise, accuracy for unseen data might degrade in cases where there is a lot of noise associated with the training examples and/or the number of training examples is very small. To alleviate this problem, C4.5 uses the post-pruning method. This approach allows C4.5 to grow a complete decision tree first, and then *post-prune* the tree. It tries to shorten the tree in order to overcome overfitting. This generally involves removal of some of the nodes or subtrees from the original decision tree. Its goal is to improve (by pruning) the accuracy on the unseen set of examples. As a result, C4.5 achieves further elimination of features through pruning. It uses rule -post pruning to remove some of the *insignificant* nodes (and hence, some *not so relevant* features) from the tree.

4 Selective Bayesian Classifier

Our purpose is to improve the performance of the Naïve Bayesian classifier by removing redundant and/or irrelevant attributes from the dataset, and only choosing those that are most informative in classification task, according to the decision tree constructed by C4.5.

4.1 Description

As described in section 3, the features that C4.5 selected in constructing its decision tree are likely to be the ones that are most descriptive in terms of the classifier, in spite of the fact that a tree structure inherently incorporates dependencies among attributes, while Naïve Bayes works on a conditional independence assumption. C4.5 will naturally construct a tree that does not have an overly complicated branching structure if it does not have too many examples that need to be learned. As the number of training examples increases, the attributes that are considered will usually be the ones that are not correlated. This is mainly because C4.5 will use only one of a set of correlated features for making good splits in training set. However, sometimes many of the branches may reflect noise or outliers (overfitting) in the training data. "Tree pruning" procedure in C4.5 attempts to identify and remove those least reliable branches, with the goal of improving classification accuracy on unseen data. Even after pruning, if the result decision tree is still too *deep* or grown into too many levels, our algorithm only picks attributes contained in the first few levels of the tree as the most representative attributes. This is supported by the fact that by the selection of attributes that split the data in the best possible way at every node, C4.5 will try to ensure that it encounters a leaf at the very earliest possible point, i.e. it prefers to construct shorter trees. And by its algorithm, C4.5 will find trees that have attributes with higher information gain nearer to the root. We conjecture that this simple method of feature selection would help Naïve Bayesian classifier perform well and learn quickly, that is, it would need fewer training examples to reach high classification accuracy.

4.2 Algorithm

1. Shuffle the original data.
2. Take 10% from the original data as training data.
3. Run C4.5 on data from step 2.
4. Select a set of attributes that appear only in the first 3 levels of the simplified decision tree as *relevant* features.
5. Repeat 10 times (step 1-4)
6. Union the sets of attributes obtained from all 10 rounds.
7. Run Naïve Bayesian classifier on the training and test data using *only* the final features selected in step

Figure 1. Selective Bayesian Classifier Algorithm:
Feature Selection Using C4.5

Figure 1 shows the algorithm for the Selective Bayesian classifier. We first shuffle the training data and use 10% of that to run C4.5 on. This is to make sure that all the subsamples are not biased toward any particular classes. We find 10% of the training to be a good size for feature selection process. Once we run C4.5 and obtain the decision tree, we only pick attributes that only appear in the first 3 levels of the decision trees as the most relevant features. We hypothesize that if a feature in the deeper levels on any one execution of C4.5 is *relevant* enough, it will finally rises up and appear in one of the top levels of the tree in some other executions of C4.5. It is important to note that in the 10 different iterations, C4.5 may give slightly different decision trees, i.e. it uses different attributes to produce decision tree for different training sets, even when the number of training examples is the same across these training sets. We union all the attributes from each run, and finally, run the Naïve Bayesian classifier on the training and test data using only those features selected in the previous step.

5 Experimental Evaluation

5.1 The Datasets

We used 10 datasets from the UCI repository and one synthetic dataset, shown in Table 1. The Synthetic Data, created with Gaussian distribution, contains 1,200,000 instances with 20 attributes and 2 classes. We chose 10 datasets from the UCI databases, 5 of which Naïve Bayes outperforms C4.5 and the other 5 of which C4.5 outperforms Naïve Bayes.

Table 1. Descriptions of domains used

Dataset	# Attributes	# Classes	# Instances
Ecoli	8	8	336
GermanCredit	20	2	1,000
KrVsKp	37	2	3,198
Monk	6	2	554
Mushroom	22	2	8,124
Pima	8	2	768
Promoter	57	2	106
Soybean	35	19	307
Wisconsin	9	2	699
Vote	16	2	435
SyntheticData	20	2	1,200,000

5.2 Experimental Design

1. Each dataset is shuffled randomly.
2. Produce *disjoint* training and test sets as follows.
   10% training and 90% test data
   20% training and 80% test data
   … … … …
   90% training and 10% test data
   99% training and 1% test data
3. For each set of training and test data, run
   - Naïve Bayesian Classifier (NBC)
   - C4.5, and
   - Selective Bayesian Classifier (SBC)
4. Repeat 15 times

The classifier accuracy is determined by *Random Subsampling* method. The overall accuracy estimate is the mean of the accuracies obtained from all iterations. This will give us information about both the learning rates, as well as the asymptotic accuracy of the learning algorithms used.

5.3 Experimental Results

The results confirm the initial hypotheses. It is clear that SBC does improve NBC's performance in all domains, and it does learn faster than both C4.5 and NBC on all the dataset, i.e. with small number of training data (10%), the prediction accuracy for SBC is higher.

Figure 2 – 11 depict the learning curves for the 10 UCI datasets.

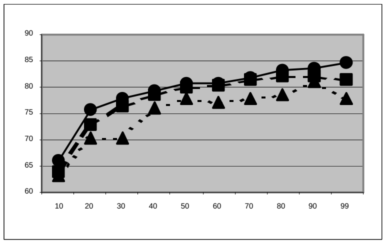

Figure 2: Ecoli. 336 instances, 8 attrib, 8 classes, 4 SBC attrib.

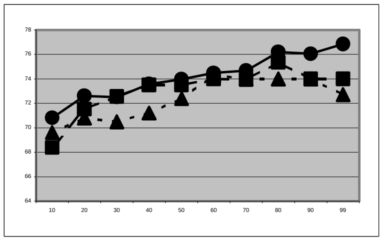

Figure 3: German. 1,000 instances, 20 attrib, 2 classes, 6 SBC attrib.

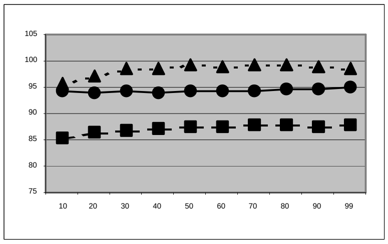

Figure 4: KrVsKp. 3,198 instances, 37 attrib, 2 classes, 4 SBC attrib.

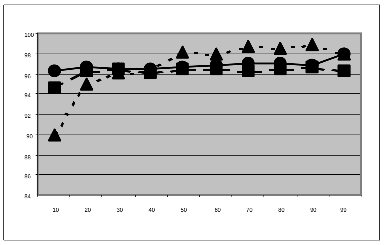

Figure 5: Monk. 554 instances, 6 attrib, 2 classes,4 SBC attrib.

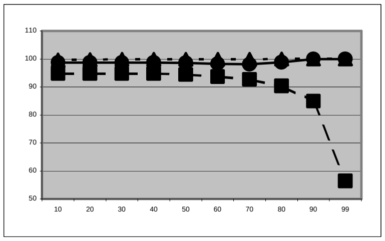

Figure 6: Mushroom. 8,124 instances, 22 attrib, 2 classes, 6 SBC attrib.

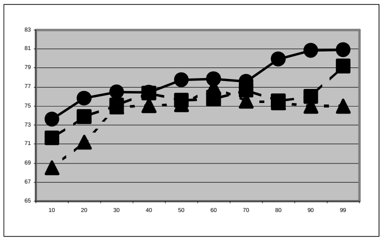

Figure 7: Pima. 768 instances, 8 attrib, 2 classes, 5 SBC attrib.

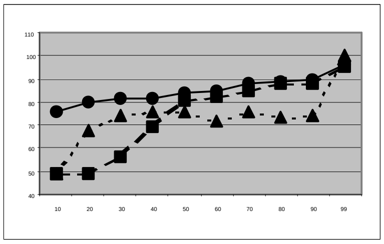

Figure 8: Promoter. 106 instances, 57 attrib, 2 classes, 5 SBC attrib.

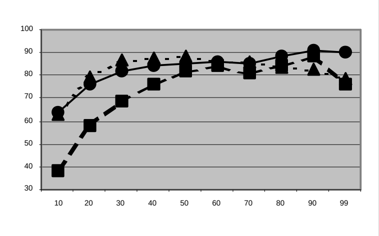

Figure 9: Soybean. 307 instances, 35 attrib, 19 classes, 12 SBC attrib.

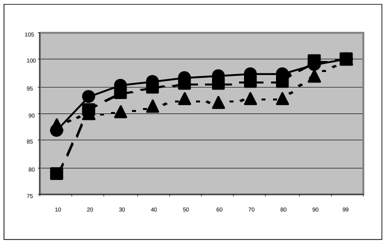

Figure 10: Wisconsin. 699 instances, 9 attrib, 2 classes, 4 SBC attrib.

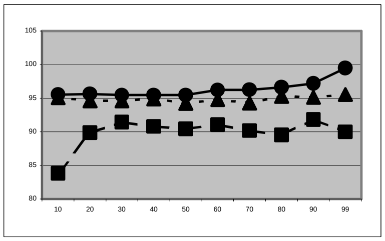

Figure 11: Vote. 435 instances, 16 attrib, 2 classes, 3 SBC attrib.

The X -axis shows the training data (%), and the Y -axis shows the accuracy on test data. SBC is represented by ● with a solid line . NBC is represented by ■ with a big dash line. And C4.5 is represented by ▲ with a small dash line.

Note that all the C4.5 accuracy considered in this experiment is based on the simplified decision tree (with pruning). This accuracy is usually higher on the unseen data, comparing with the accuracy based on unpruned decision trees.

To see a clearer picture on the SBC performance, table 2 shows the results for NBC, C4.5, and SBC algorithms using 80% of the data for training and 20% for testing. The figures shown in bold reflect the winning method on each dataset. The last two columns show the improvement of SBC over NBC and C4.5.

Table 2. Accuracy of each method using 5-fold cross-validation (15 iterations)

Dataset	**NBC**	**C4.5**	**SBC**	**SBC vs NBC**	**SBC vs C4.5**
Ecoli	81.99	78.65	**83.27**	+1.6%	+5.9%
German	75.35	74.00	**76.21**	+1.1%	+3.0%
KrVsKp	87.81	**99.12**	94.69	+7.8%	-4.5%
Monk	96.16	**98.46**	97.47	+1.4%	-1.0%
Mushroom	90.37	**99.80**	98.85	+9.4%	-1.0%
Pima	75.03	75.35	**79.94**	+6.5%	+6.1%
Promoter	87.66	66.67	**88.72**	+1.2%	+33.1%
Soybean	84.02	83.20	**88.27**	+5.1%	+6.1%
Wisconsin	95.78	92.63	**97.38**	+1.7%	+5.1%
Vote	89.54	95.29	**96.61**	+7.9%	+1.4%

From table 2, it is apparent that SBC outperforms the original NBC in EVERY domain , giving the accuracy improvement up to 9.4%. SBC also outperforms C4.5 in almost all the domain, giving the accuracy improvement up to 33.1%. Even though, SBC cannot beat C4.5 in some cases, it still gives quite big improvement over the Naïve Bayes (7.8%, 1.4%, and 9.4%).

Our experimental results demonstrate that C4.5 does pick good features for its decision tree (especially ones that are nearer to the root), which in turn asymptotically improves the accuracy of the Naïve Bayesian algorithm, when *only* those features are used in the learning process. Table 3 shows the number of features selected for Selective Bayesian classifier. On almost all the datasets, surprisingly more than half of the original attributes were eliminated. 30% or less of all attributes *selected* were shown in bold, which means that we can actually pay no attention to more than 70% of the original data and still achieve high accuracy in classification.

Table 3. Number of features selected

Dataset	**# Attributes**	**# of Attributes selected**
Ecoli	8	4
German Credit	20	**6**
KrVsKp	37	**4**
Monk	6	4
Mushroom	22	**6**
Pima	8	5
Promoter	57	**5**
Soybean	35	12
Wisconsin	9	4
Vote	16	**3**
Synthetic Data	20	12

For speedup and scalability issues, we ran SBC on a large synthetic data just to see how fast it can learn. The running time for SBC on our synthetic data give **1.14** and **4.24** speedup over the original NBC and C4.5, respectively. Note that we only used **2,000** instances out of the total of 1,200,000 instances for C4.5 feature selection process, which made it a very quick operation. Hence, in practice, if the dataset is large enough, we can even sample much less than 10% of data for the feature selection process. The number of attributes selected by SBC was **12** out of the total of 20 attributes. Table 4 illustrates the mean elapsed time (user and system time) for each classifier on this synthetic data, using 1,000,000 instances for training and 200,000 instances for test data.

Table 4. Mean Elapsed time for Synthetic Dataset (sec)

NBC	**C4.5**	**SBC**
37.546	139.5	32.912

The running times of both SBC and NBC are much less than that of C4.5 because Bayesian classifier only needs to go through the whole training data once. They are also space efficient because they build up a frequency table in size of the product of the number of attributes, number of class values, and the number of values per attribute. SBC, comparing to NBC, learns faster because fewer attributes are involved in learning. However, it is obvious that most of the time spent in both algorithms was on I/O, reading the training data. That explains why SBC time did not reduce much from NBC time. If there exists a very fast way of removing unwanted features from a very large dataset, SBC would only need 25.746 seconds and give 31.4% improvement over NBC.

6 Conclusion

A simple method to improve Naïve Bayesian learning that uses C4.5 decision trees to select features has been described. The empirical evidence shows that this method is very fast and surprisingly successful, given the very different natures of the two classification methods. This Selective Bayesian classifier is asymptotically at least as accurate as the better of C4.5 and Naïve Bayes on almost all the domains on which the experiments w ere performed. Further, it learns faster than both C4.5 and NB on each of these domains.

This work suggests that C4.5 decision trees systematically select good features for Naïve Bayesian classifier to use. We believe the reasons are that C4.5 does not use redundant attributes in constructing decision trees, since they cannot generate different splits of training data. When few training examples are available, C4.5 uses the most relevant features it can find. The high accuracy of SBC achieves with few training examples is indicative of the fact that using these features for probabilistic induction leads to higher accuracy produced in each of the domains we have examined.

References

[1] Domingos, P. and Pazzani, M. On the Optimality of the Simplie Bayesian Classifier under Zero-One Loss. Kulwer Academic Publishers, Boston.

[2] Duda, R.O. and Hart, P.E. (1973). Pattern Classification and Scene Analysis. New York, NY: Wiley and Sons.

[3] Elkan, C. Bo osting and Naïve Bayesian Learning. Technical Report No. CS97 -557, Department of Computer Science and Engineering, University of California, San Diego, Spetember 1997.

[4] Hand, D. and Yu, K. (2001) Idiot's Bayes —Not So Stupid After All? International Statistical Review (2001), 69, pp.385-398.

[5] Keogh, E. and Pazzani, M. Learning Augmented Bayesian Classifiers: A comparison of distribution-based and classification -based approaches. Uncertainty 99, 7th Int'l Workshop on AI and Statistics, Ft. Lauderdale, Florida, 225-230.

[6] Langley, P. and Sage, S. Induction of Selective Bayesian Classifiers. Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence (1994). Seattle, WA: Morgan Kaufmann

[7] Ming, K. and Zheng, Z. Improving the Performance of Boosting for Naïve Bayesian Classification. In Proceedings of the PAKDD-99, pp.296-305, Beijing, China.

[8] Pazzani, M. (1996). Constructive Induction of Cartesian Product Attributes. Information, Statistics and Induction in Science. Melbourne, Australia.

[9] Quinlan, J.R. (1993). C4.5: Programs for Machine Learning, CA: Morgan Kaufmann.