

LBR-Meta: An Efficient Algorithm for Lazy Bayesian Rules

Zhipeng Xie

School of Computer Science

Fudan University

220 Handan Road, Shanghai 200433, PR. China

xiezp@fudan.edu.cn

Abstract

LBR is a highly accurate classification algorithm, which lazily constructs a single Bayesian rule for each test instance at classification time. However, its computational complexity of attribute-value pair selection is quadratic to the number of attributes. This fact incurs high computational costs, especially for datasets of high dimensionality. To solve the problem, this paper proposes an efficient algorithm LBR-Meta to construct lazy Bayesian rules in a heuristic way. It starts with the global classifier trained on the whole instance space. At each step, the attribute-value pair that best differentiates the performance of the current local classifier is selected and used to reduce the current subspace to a further smaller subspace for the next step. The selection strategy used has a linear computational complexity with respect to the number of attributes, in contrast to the quadratic complexity in LBR. Experimental results manifest that LBR-Meta has achieved comparable accuracy with LBR, but at a much lower computational cost.

Keywords: naïve Bayesian classifiers, lazy Bayesian rules, classification, decision trees.

1 Introduction

Classification is a core problem in machine learning and data mining. A variety of approaches such as naïve Bayes, nearest neighbours, decision trees, and neural networks, have been proposed to deal with it. However, none of these approaches can beat the others on all domains. Each approach has its own strong and weak points. Recently, a hotspot is to combine different learning paradigms together for higher overall predictability.

Among these numerous existing classification methods, the naïve Bayesian classifier (NB) (Duda & Hart 1973) is simplest and computationally most efficient. It is also robust to noise and irrelevant attributes, and has outperformed many complicated methods in varied application domains. However, all these advantages are obtained through its strong (or sometimes impractical) independence assumption that the attributes are conditionally independent given the class label. As a result, its performance may degrade significantly in the domains where the assumption does not hold. To alleviate this

problem, Kohavi (1996) proposed a hybrid algorithm NBTree to integrate decision tree algorithm and naïve Bayesian classifier together. Although NBTree has achieved higher accuracies than either naïve Bayesian classification method or decision tree learning algorithm in most datasets, it suffers from the small disjunct problem due to the tree structure. To deal with it, Zheng and Webb (2000) proposed the lazy Bayesian rule learning algorithm (LBR) which lazily induces at classification time a single Bayesian rule for each instance to be classified. The antecedent of a Bayesian rule is a conjunction of conditions in the form of attribute-value pairs, while the consequent is a local naïve Bayesian classifier trained on an instance subspace which is used to make the decision for the test unlabelled example. The objective of LBR is to grow the antecedent of a Bayesian rule that ultimately decreases the errors of the local naïve Bayesian classifier in the consequent of the rule. The LBR algorithm adopts a hill-climbing strategy in determining which attribute-value pair should be added to the antecedent at each step. It always makes the choice such that the local Bayesian classifier trained on the reduced subspace can obtain the highest reduction in classification error. Thus, for each candidate (attribute-value) pair, LBR has to train a local Bayesian classifier on the corresponding reduced subspace, and then estimate its error rate. This process requires substantial computational cost whose complexity is quadratic to the number of attributes. It is not desirable, especially for the datasets of high dimensionalities.

This paper proposes a heuristic strategy for attribute-value pair selection, which is based on the meta-information about the performance of the current local naïve Bayesian classifier. The attribute-value pair selected partitions the current subspace into two reduced smaller subspaces such that the current local naïve Bayesian classifier has significantly different performances on these reduced subspaces. Based on this selection strategy, an algorithm LBR-Meta is designed and implemented. LBR-Meta selects an attribute-value pair and adds it to the antecedent of the current Bayesian rule at each step. This process terminates only when the size of the reduced training subset is smaller than a threshold parameter. This simple criterion guarantees that the final subspace generated by LBR-Meta is quite small, when compared with the one generated by LBR.

This paper is organized as follows: Section 2 introduces naïve Bayesian classifier and local accuracy estimation. Section 3 describes the LBR-Meta algorithm in details. The experimental results are shown in Section 4. Finally, section 5 concludes the whole paper and points out the future work.

2 Naïve Bayesian Classification and Local Accuracy Estimation

Consider a domain where instances are represented as instantiations of a vector $A=\{a_1, a_2, \dots, a_m\}$ of m nominal variables. Here, each instance x takes a value $a_i(x)$ from $domain(a_i)$ on each a_i . Further, an example (or instance) x is also described by a class label $c(x)$ from $domain(c)$. Let $D=\{(x_i, c(x_i)) \mid 1 \leq i \leq n\}$ denote the training dataset of size n . The task of classification is to construct a model (or classifier) from the training set D , which is a function that assigns a class label to a new unlabelled example.

The underlying assumption of naïve Bayesian classifiers (NB) is that attributes are conditionally mutually independent given the class label. According to the Bayes Theorem, the probability of a class label i for a given unlabelled instance $x=(v_1, \dots, v_m)$ consisting of m attribute values is given by

$$P(c=i|x) = \frac{P(c=i) \times P(x|c=i)}{P(x)}.$$

It follows from the independent assumption that $P(x|c=i) =$

$\prod_{k=1}^m P(a_k = v_k | c=i)$ holds. Thus, the class label with the highest probability given the instance x , is used as the predicted class. Note that we do not need to compute the value of $P(x)$. This is because $P(x)$ is a constant for a given x . To put it formally, the naïve Bayesian classifier trained on D can be expressed as

$$NB(x, D) = \arg \max_i \left(P(c=i) \times \prod_{k=1}^m P(a_k = a_k(x) | c=i) \right)$$

Hence, the construction of naïve Bayesian classifier on D is to estimate the probabilities $P(c=i)$ and conditional probabilities $P(a_k=v_k|c=i)$, which are done in the following way:

$$P(a_k = v_k | c=i) = \frac{|\{x \in D : a_k(x) = v_k \text{ and } c(x) = i\}| + 0.5}{|\{x \in D : c(x) = i\}| + 0.5 \times |domain(a_k)|}$$

and

$$P(c=i) = \frac{|\{x \in D : c(x) = i\}| + 0.5}{|D| + 0.5 \times |domain(c)|}.$$

Local Accuracy Estimation

To determine which classification method is more suitable for a given data set, we often need to estimate the prediction accuracy of a classifier. Popular techniques of accuracy estimation include hold-out, cross-validation, and leave-one-out. As naïve Bayesian classifiers have been adopted as base local classifiers in this paper, which is easy to be updated incrementally, leave-one-out can be implemented easily and efficiently for accuracy estimation.

For the naïve Bayesian classifier $NB(D_1)$ trained on a training set D_1 , leave-one-out method can be used to estimate its accuracy as follows:

$$ACC_G(NB(D_1)) = |\{x \in D_1 | NB(x, D_1 - \{x\}) = c(x)\}| / |D_1|.$$

However, the estimated accuracy above is actually a kind of global accuracy, it is averaged over the whole instance space (or the whole training set). A classifier usually

performs differently in different regions (or subspaces). For example,

For a subset D_2 of D_1 , the *local accuracy* of $NB(D_1)$ on D_2 is estimated by

$$ACC_L(NB(D_1), D_2) = |\{x \in D_2 | NB(x, D_1 - \{x\}) = c(x)\}| / |D_2|.$$

It is evident that $ACC_G(NB(D_1)) = ACC_L(NB(D_1), D_1)$. This local accuracy plays an important role in classifier selection, because what we are interested in is actually which classifier performs best in the local subspace surrounding the target test example.

3 LBR-Meta: A heuristic algorithm for lazy Bayesian rules based on meta-information

A Bayesian rule r takes the form of “*antecedent(r)* \rightarrow *consequent(r)*”, where the antecedent of r is a conjunction of attribute-value pairs, and the consequent of r is a local naïve Bayesian classifier. An instance x satisfies a attribute-value pair (a, v) if and only if the value of attribute a on x equals to v (that is, $a(x)=v$). The instance subspace defined by r consists of all the instances that satisfy all the attribute-value pairs in r 's antecedent. The subset of all training examples that satisfy the antecedent of r is called the local training set of r . The original LBR algorithm requires that the local naïve Bayesian classifier as the consequent should be trained on the local training set of r . This requirement is relaxed by the LBR-Meta algorithm of this paper. It is only required that the training set of the local naïve Bayesian classifier should be no less than the local training set of the Bayesian rule.

The pseudo-code of LBR-Meta algorithm is listed in figure 1, which will be fully explained in this section. For any given unlabelled test example x_{test} , LBR-Meta starts from the global Bayesian rule where the antecedent is empty and the consequent is trained on the whole training set D . At each step, an attribute-value pair is selected and added into the antecedent to reduce the current instance subspace, and hence reduce the corresponding local training subset. There are two key problems to be solved in the LBR-Meta algorithm. The first key problem is how to select an attribute-value pair. After an attribute-value is added to the antecedent and the subspace is refined to a further smaller subspace, the second problem appears: how to determine which local classifier is best suited for this reduced subspace. The following two subsections are devoted to the detailed solutions to these two problems.

3.1 A Heuristic Criterion for Attribute-Value Pair Selection

The first problem to be dealt with is: which attribute-value pair should be selected to reduce the current instance subspace? The original LBR algorithm makes the decision so that the local classifier trained on the reduced subspace has the lowest estimated error rate. With this selection criterion, one classifier has to be induced for each attribute-value pair, which leads to high computational overhead. In order to improve computational efficiency, this paper takes a heuristic way: it selects the attribute-value pair that can differentiate the performance of the current local classifier on the current subspace. The details go as follows:

Let r be the current Bayesian rule, D_{local} be the current local training set, and $NB(D_{current})$ be the current local naïve Bayesian classifier associated with r . As stated above, this local naïve Bayesian classifier is trained on $D_{current}$ which does not necessarily equal to D_{local} . However, it is required that $D_{local} \subseteq D_{current}$. A boolean attribute c_{meta} is appended to each training example, whose value denotes whether or not the current local naïve Bayesian classifier $NB(D_{current})$ can classify the corresponding training example correctly with leave-one-out method. Put it formally, for each $x \in D_{local}$:

$$c_{meta}(x) = \begin{cases} true & \text{if } NB(x, D_{current} - \{x\}) = c(x); \\ false & \text{otherwise.} \end{cases}$$

According to the values of c_{meta} , the current local training set D_{local} is partitioned into two subsets:

$$(D_{local})_{true} = \{x \in D_{local} | c_{meta}(x) = true\},$$

and

$$(D_{local})_{false} = \{x \in D_{local} | c_{meta}(x) = false\}.$$

Furthermore, each attribute-value pair (a, v) can also partition the current local training set D_{local} into two subsets:

$$(D_{local})_0 = \{x \in train | a(x) = v\}$$

and

$$(D_{local})_1 = \{x \in train | a(x) \neq v\}.$$

The subset $(D_{local})_0$ consists of the training examples in D_{local} that take value v on attribute a ; while $(D_{local})_1$ consists of the training examples in D_{local} that do not take value v on a . The current local naïve Bayesian classifier may have different performance (or accuracies) on these two subsets. Its local accuracy on $(D_{local})_i$, $i \in \{0, 1\}$, is estimated by

$$ACC_L(NB(D_{current}), (D_{local})_i) = \frac{|(D_{local})_{i,true}|}{|(D_{local})_i|},$$

$$\text{where } (D_{local})_{i,true} = (D_{local})_i \cap (D_{local})_{true}.$$

The idea used for heuristic attribute-value pair selection is: the more the difference between the local accuracies of these two subsets is, the more likely can we expect to get good performance by zooming the local naïve Bayesian classifier into the corresponding subspace of $(D_{local})_0$. Using information gain, the goodness of an attribute pair (a, v) is measured by LBR-Meta as follows:

$$\begin{aligned} Goodness(a, v) = & Info(|(D_{local})_{true}|, |(D_{local})_{false}|) \\ & + \sum_{i=0}^1 \frac{|(D_{local})_i|}{|D_{local}|} Info(|(D_{local})_{i,true}|, |(D_{local})_{i,false}|), \end{aligned}$$

$$\text{where } (D_{local})_{i,false} = (D_{local})_i \cap (D_{local})_{false},$$

$$\text{and } Info(n_1, n_2) = -\log \frac{n_1}{n_1 + n_2} - \log \frac{n_2}{n_1 + n_2}.$$

The information gain has also been used for attribute selection in a famous decision tree algorithm ID3 (Quinlan 1986, Quinlan 1993). The difference is that it is concerned about the Boolean attribute a_{meta} , while it is concerned about the class attribute c in ID3.

Comparison with LBR algorithm: The LBR algorithm tries to add each candidate attribute-value pair to the antecedent of the current Bayesian rule, then reduces the

local training set, and then trains a local classifier accordingly. The total time spent is $O(|A| \times |A| \times |D_{local}|)$.

The LBR-Meta algorithm calculates the goodness of each candidate attribute-value pair. The time needed in total is $O(|A| \times |D_{local}|)$.

3.2 Local Naïve Bayesian Classifier Selection

After an attribute-value pair has been selected and added to the antecedent of the rule, the current local subspace is reduced to a further smaller subspace. The current local training set D_{local} is also reduced to a smaller local training subset D_{red} . The aim of local classifier selection is to select the local classifier that has the highest local accuracy on the reduced smaller subspace that is measured on the smaller local training subset D_{red} . LBR-Meta uses the variable *LocalClassifiers* to represent the set of all qualified local naïve Bayesian classifiers that have already been generated. The global naïve Bayesian classifier is assumed to be qualified and added into *LocalClassifiers* (line 2 in figure 1). Once a local naïve Bayesian classifier *SubLocalNB* is trained on the reduced training subset D_{red} at each step, it is compared with all the qualified local classifiers in *LocalClassifiers*. If *SubLocalNB* has the highest estimated local accuracy on D_{red} , it is qualified and added into *LocalClassifiers* (lines 17-18 in figure 1); otherwise, *SubLocalNB* will be discarded.

- If D_{red} contains too few training examples, that is, the size of D_{red} is less than a threshold *Thresh* (line 9 in figure 1), the local accuracy estimated on D_{red} can not provide reliable information about the accuracy on the corresponding subspace, and thus the current local naïve Bayesian classifier *CurrentLocalNB* is used directly to make the decision for x_{rest} . Note: the default value of *Thresh* is set to 5.
- If the size of D_{red} is larger than or equal to the threshold *Thresh*, we first estimate the local accuracy on D_{red} for each local classifier in *LocalClassifiers* (lines 10-12 in figure 1). The one with the highest estimated local accuracy is selected and denoted by *BestSupNB* with the estimated local accuracy stored in the variable *BestSupAcc* (lines 13-14 in figure 1). Then we compare the size of D_{red} with another parameter *MinNumObj* which should be set greater than *Thresh* (Note that *MinNumObj* has default value 15 in this paper):

A. If the size of D_{red} is less than *MinNumObj*, the decision made by the local classifier *BestSupNB* is returned as the result (line 15 in figure 1).

B. If D_{red} contains enough training examples to train a local naïve Bayesian classifier *SubLocalNB* (line 16 in figure 1), or in another words, the size of D_{red} is no less than *MinNumObj* (line 15 in figure 1), the classifier *SubLocalNB* will compete with other local classifiers previously generated for the domination of the subspace corresponding to the training subset D_{red} . If *SubLocalNB* is more accurate on D_{red} than *BestSupNB*, the classifier *SubLocalNB* is added into *LocalClassifiers*, and the *CurrentLocalNB* is set to be *SubLocalNB* (lines 17-19 in figure 1); otherwise, the *CurrentLocalNB* is set to be *BestSupAcc* (lines 20-21 in figure 1).

LBR-Meta**Input:** A : a set of attributes D : a set of training examples described using A and class attribute c x_{test} : a test example described using A **Output:** a predicted class for x

```
1  CurrentLocalNB:=NB(D);
2  Add NB(D) into the set of local classifiers LocalClassifiers;
3  Dlocal:=D; Alocal:= $\{a \in A \mid a(x_{test}) \text{ is not missing}\}$ ;
4  FOR each example  $x$  in  $D$  DO  $c_{meta}(x) := \begin{cases} true, & \text{if } NB(x, D - \{x\}) = c(x) \\ false, & \text{if } NB(x, D - \{x\}) \neq c(x) \end{cases}$ ; ENDFOR
5  WHILE (Alocal is not empty) DO
6    Calculate Goodness( $a, a(x_{test})$ ) for each attribute  $a \in A_{local}$ , according to the equation ();
7     $att := \arg \max_{a \in A_{local}} Goodness(a, a(x_{test}))$ ;
8     $D_{red} := \{x \in D_{local} \mid att(x) = att(x_{test})\}$ ;  $A_{local} := A_{local} - \{att\}$ ;
9    IF  $|D_{red}| < Thresh$  THEN return CurrentLocalNB( $x_{test}$ ); ENDIF
10   FOR each local classifier NB( $D_i$ ) in LocalClassifiers DO
11     estimate its local accuracy on  $D_{red}$ :  $ACC_L(NB(D_i), D_{red})$ ;
12   ENDFOR
13    $BestSupNB := \arg \max_{NB(D_i) \in LocalClassifiers} ACC_L(NB(D_i), D_{red})$ ;
14    $BestSupAcc :=$  the estimated local accuracy of BestSupNB on  $D_{red}$ ;
15   IF  $|D_{red}| < MinObjNum$  THEN return BestSupNB( $x_{test}$ ); ENDIF
16    $SubLocalNB := NB(D_{sub})$ ;
17   IF  $ACC_G(SubLocalNB) \geq BestSupAcc$  THEN
18     add SubLocalNB into LocalClassifiers;
19     CurrentLocalNB:=SubLocalNB;
20   ELSE
21     CurrentLocalNB:=BestSupNB;
22   ENDIF
23    $c_{meta}(x) := \begin{cases} true, & \text{if } NB(x, Tr - \{x\}) = c(x) \\ false, & \text{if } NB(x, Tr - \{x\}) \neq c(x) \end{cases}$  for each  $x$  in  $D_{red}$  where  $NB(Tr) = CurrentLocalNB$ ;
24    $D_{local} := D_{red}$ ;
25 ENDWHILE
```

Figure 1. The LBR-Meta Algorithm

Finally, due to the fact that the current local naïve Bayesian classifier has possibly been changed, we need to update the attribute values of c_{meta} for all training examples in the reduced subspace (line 23 in figure 1).

3.3 Analysis of LBR-Meta

LBR uses a statistical sign-test to control tradeoff between the decreasing error by removing harmful attribute-value pair and increasing error as a result of reducing the accuracy of the probability estimations of the local naïve Bayesian classifier due to decreases in the size of the available training set. In addition, this statistical sign-test is also used as the termination condition for the repetitive process. However, the strategies adopted by LBR-Meta are different from LBR, which are described as follows.

Firstly, even if the accuracy of the probability estimations of the local naïve Bayesian classifier is decreasing with a smaller training set, it has already been reflected in its estimated local accuracy. That is to say, it is expected that the classifier should get low estimated accuracy if the probability estimations inside it are not reliable. Or if the estimated accuracy of the classifier is high, it means that the accuracy of the inside probability estimations is acceptable.

Secondly, LBR algorithm tries to add one attribute-value pair to the antecedent at each step. This process is repeated until no attribute-value pair could lead to a local naïve Bayesian classifier with statistically higher accuracy on the reduced subspace. However, in LBR-Meta, even if the attribute-value pair that is selected cannot lead to a qualified local naïve Bayesian classifier (that is, the classifier trained on the reduced training subset has lower accuracy), the process is not terminated. This strategy does not suffer from local maxima. The final subspace produced by LBR-Meta for a given test example is much smaller than that produced by LBR.

In spite of the fact that the individual attribute-value pair selected by LBR algorithm may be better than that selected by LBR-Meta, the above two strategies taken by LBR-Meta have compensated for this shortcoming in some degree. It will be shown in the experimental part that LBR-Meta has also yielded high accuracies comparable with LBR.

Furthermore, our actual objective is to maximize the local accuracy at the point of the test example, which is usually approximately by the accuracy on a subspace around the test example. There is also a tradeoff between variance and bias. If the subspace is too large, the accuracy estimated may differ significantly from the accuracy at the point of the test example. On the contrary, if the subspace is too small, the accuracy estimated is not reliable (with large variance).

4 Experimental Results

To evaluate the performance of the proposed LBR-Meta algorithm, we compare it with two other closely related algorithms: the naïve Bayesian classifier (NB), and the lazy Bayesian rule algorithm (LBR). Twenty-three data sets from UCI machine learning repository are used for the comparison, with detailed information listed in table 1. The datasets are drawn randomly, with the requirement

that each dataset should contain at least 300 examples. The numbers of attributes vary from 8 to 36, and the numbers of examples from 303 to 12960. Ten-fold cross validation is conducted on each data set, such that each fold has at least 30 examples. For LBR can only deal with discrete attributes, the continuous attributes are discretized by an entropy-based discretization algorithm (Fayyad & Irani 1993) as a preprocess.

	# examples	# attributes	# classes
Australian	690	14	2
Breast	699	10	2
Chess	3196	36	2
Cleve	303	13	2
Crx	690	15	2
Diabetes	768	8	2
German	1000	20	2
Horse-Colic	368	22	2
Hypothyroid	3163	25	2
Ionosphere	351	34	2
Mushroom	8124	22	2
Nursery	12960	8	5
Pendigits	10992	16	10
Pima	768	8	2
Satimage	6435	36	6
Segment	2310	19	7
Shuttle-Small	5800	9	7
Sick	2800	29	2
Solar	323	12	6
Soybean-Large	683	35	19
Tic-Tac-Toe	958	9	2
Vote	435	16	2
Waveform-21	5000	21	3

Table 1: Datasets used for comparison

The error rates of these algorithms over all datasets are listed in table 2. The final row shows the mean error rates across all the datasets. Among the three algorithms, LBR-Meta gets the best result, which is slightly better than LBR, and greatly better than NB. When we look at individual datasets, it is found that LBR-Meta has lower error rates than LBR on 10 datasets and higher than LBR on 11 datasets.

The mean error rate is only a naive measurement of a classification method over these datasets. To evaluate the relative error rate reduction, we present a new measurement called the *relative difference* (*rdiff*). For a given a dataset, assume e_1 be the error rate of the first method, and e_2 be the error rate of the second method. The relative difference from e_2 to e_1 is defined as:

$$rdiff(e_2, e_1) = \frac{e_2 - e_1}{\max\{e_1, e_2\}}$$

When both e_1 and e_2 are zero, the value is defined to be 0. It can be seen that if e_2 equals to e_1 , the relative difference is zero; if e_2 is larger than e_1 , the relative difference is positive; and if e_2 is less than e_1 , the relative difference is negative. Therefore, the smaller the relative difference

from e_2 to e_1 is, the better is the second method relatively than the first method on the dataset. Furthermore, it is evident from the definition that $rdiff(e_2, e_1) = -rdiff(e_1, e_2)$. The relative difference may be better than the error rate ratio used by Zheng & Webb (2000), in that the error rate ratio will lead to an infinite value when some error rate appears (or approaches) zero, and in that the $rdiff$ value lies in the interval $[-1, 1]$, while the error rate ratio lies in the interval $(0, \infty]$. The last row in Table 2 gives out the mean relative difference of each algorithm to LBR-Meta over all datasets. From the fact that the mean relative difference from NB to LBR-Meta is 25.5%, conclusion can be drawn that LBR-Meta has substantially reduced the error rates of NB.

In addition, the one-tailed pairwise t -test (with significance level set at 5%) shows that LBR-Meta wins on 3 datasets (Chess, Nursery, and Tic-Tac-Toe), and loses on 1 dataset (Waveform-21) when compared with LBR. LBR-Meta also wins significantly on 11 datasets and loses on 0 when compared with NB. When comparing LBR with NB, we find that LBR significantly wins on 13 datasets and loses on 0 datasets.

These results have shown that the algorithm LBR-Meta is comparable to LBR on the experimental datasets, but the computational cost of LBR-Meta is far less than that of LBR, which is shown next.

	LBRMeta	LBR	NB
Australian	14.3%	14.2%	14.3%
Breast	3.4%	3.0%	3.0%
Chess	1.3%	2.6%	12.0%
Cleve	17.5%	16.5%	16.5%
Crx	14.3%	14.8%	14.2%
Diabetes	25.7%	25.4%	25.4%
German	26.5%	25.1%	25.2%
Horse-Colic	20.4%	17.4%	21.2%
Hypothyroid	1.1%	1.2%	1.5%
Ionosphere	10.2%	10.5%	10.0%
Mushroom	0%	0%	3.5%
Nursery	1.7%	2.3%	9.7%
Pendigits	4.3%	3.9%	12.3%
Pima	24.0%	25.5%	25.3%
Satimage	13.8%	13.6%	17.9%
Segment	6.1%	5.8%	8.8%
Shuttle-Small	0.3%	0.3%	0.7%
Sick	2.1%	2.6%	2.9%
Solar	29.5%	30.7%	30.7%
Soybean-Large	7.5%	6.7%	7.0%
Tic-Tac-Toe	11.2%	14.4%	29.8%
Vote	5.5%	6.4%	9.7%
Waveform-21	19.0%	16.2%	18.9%
Mean	11.27%	11.28%	13.93%
Mean $rdiff$ to LBR-Meta	0	0.022	0.255

Table 2: Error rate comparison

The information about the runtimes is shown in table 3. For each dataset (in the first column), the second column records the running time (in seconds) of LBRMeta, the third column records the running time of LBR, and the fourth column is the ratio of LBR's runtime to LBRMeta's runtime. The runtime values are averaged over the ten folds. If the value in the fourth column is large than 1, it means that LBR-Meta runs faster on the corresponding dataset. From the table, LBR-Meta runs about 10.4 times faster than LBR on Soybean-Large (with 35 attributes), 7.45 times faster on Chess (with 36 attributes).

	LBR-Meta	LBR	Runtime Ratio of LBR to LBR-Meta
Australian	0.5922	0.4798	0.81
Breast	0.2986	0.1955	0.65
Chess	28.9234	215.403	7.45
Cleve	0.0764	0.0627	0.82
Crx	0.6391	0.5564	0.87
Diabetes	0.2422	0.1235	0.51
German	1.0595	1.1486	1.08
Horse-Colic	0.1124	0.2388	2.12
Hypothyroid	27.9467	36.6264	1.31
Ionosphere	0.250	0.9625	3.85
Mushroom	86.9904	188.848	2.17
Nursery	69.501	71.2436	1.03
Pendigits	62.7794	183.488	2.92
Pima	0.2499	0.1173	0.47
Satimage	46.1528	193.495	4.19
Segment	5.8579	10.8002	1.84
Shuttle-Small	16.386	18.2967	1.11
Sick	24.9702	49.5563	1.98
Solar	0.1062	0.1345	1.27
Soybean-Large	1.4609	15.2015	10.41
Tic-Tac-Toe	0.4281	0.4157	0.97
Vote	0.2048	0.3374	1.65
Waveform-21	17.5639	32.7342	1.86

Table 3: Runtime Comparison (in seconds)

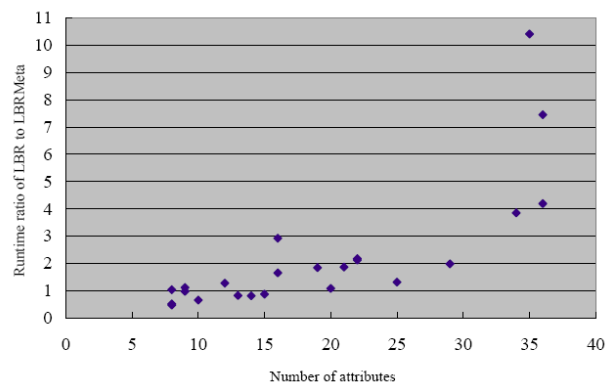


Figure 2: Scatter Plot

Furthermore, we use a scatter plot in Figure 2 to check the relationship between the runtime ratio of LBR to LBR-meta and the number of attributes. In Figure 2, each dot represents a dataset whose x coordination is the

number of attributes, and whose y coordination is the runtime ratio of LBR to LBR-Meta. A dot above the line $y=1$ means that the algorithm LBR-Meta is faster than LBR on the corresponding dataset. The higher is the y coordination of a dot, the faster and the better LBR-Meta is. Clearly, it can be observed from the figure that LBR-Meta runs faster than LBR for all the datasets with more than 15 attributes, and that LBR-Meta is much faster than LBR with increasing number of attributes in the datasets.

Finally, an ensemble technique, Bagging (Breiman 1996), is applied to LBR-Meta, LBR-Bag, and NB to check its effect in reducing error rate. As has been pointed out by Bauer & Kohavi (1999), naive Bayesian classifiers are not sensitive to the small change caused by resampling or replication of training examples. However, due to the facts that the final naive Bayesian classifiers generated by LBR-Meta and LBR are trained on a local training subset, and that the small change may change the attribute-value selected in the repetitive process, it is conjectured that LBR-Meta and LBR be more sensitive to Bagging. The experimental results are shown in Table 4, where the ensemble size is set at 10.

	LBRMeta-Bag	LBR-Bag	NB-Bag
Australian	14.1%	14.5%	14.3%
Breast	3.6%	2.9%	2.9%
Chess	0.8%	1.7%	13.2%
Cleve	16.5%	16.5%	16.5%
Crx	13.8%	13.9%	14.5%
Diabetes	25.1%	24.6%	25.0%
German	26.2%	24.8%	25.0%
Horse-Colic	17.1%	17.1%	20.9%
Hypothyroid	1.0%	1.1%	1.5%
Ionosphere	10.0%	9.4%	10.2%
Mushroom	0%	0%	3.6%
Nursery	1.2%	2.1%	9.8%
Pendigits	3.2%	2.9%	12.1%
Pima	23.3%	24.2%	24.6%
Satimage	12.0%	12.5%	17.8%
Segment	5.5%	4.9%	8.6%
Shuttle-Small	0.2%	0.3%	0.7%
Sick	2.2%	2.6%	3.1%
Solar	28.8%	28.2%	30.1%
Soybean-Large	6.7%	6.7%	7.6%
Tic-Tac-Toe	6.6%	10.1%	29.8%
Vote	5.3%	4.4%	9.9%
Waveform-21	16.8%	16.1%	18.9%
Mean	10.43%	10.50%	13.94%

Table 4: Bagging on LBR and LBR-Meta

From the results listed in Table 4, it can be seen that Bagging technique totally has no effect on naive Bayesian (NB) method, as NB-Bag has almost the same mean error rate as NB. By applying Bagging to LBR-Meta, the resulting LBRMeta-Bag gets 20 wins, 1 draws and only 2 loses when compared with the base LBR-Meta algorithm, while the mean error rate decreases from 11.27% to 10.43%. Applying Bagging to LBR has similar effects too.

5 Conclusion and Future Work

This paper proposes an algorithm LBR-Meta which makes use of a heuristic criterion for attribute-value pair selection in the lazy construction of a Bayesian rule. This criterion can be calculated in linear time with respect to the number of attributes, which has greatly improved the efficiency of the resulting algorithm. Experimental results have also shown that this algorithm also achieves high accuracy comparable to the classical LBR algorithm.

The future work about LBR-Meta is to enhance it with the ability to handle continuous attributes directly. This may be done as follows: Given a continuous attribute, the information gain with respect to the c_{meta} attribute is calculated for each possible split point. The best split point with the highest information gain value is selected, which can partition the current subspace into two reduced smaller subspaces. This continuous attribute together with the best split point competes with all the other attributes in the attribute-value pair selection. This straightforward process can endows the LBR-Meta with continuous-attribute handling ability.

Acknowledgements: This work was funded in part by National Natural Science Foundation of China under grant number 60503025

6 References

- Bauer, E., & Kohavi, R. (1999): An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. *Machine Learning*, 36: 105-142
- Blake, C., Keogh, E., & Merz, C.J. (1998): UCI Repository of Machine Learning Databases. <http://www.ics.uci.edu/~mlearn/MLRepository.html>. Irvine, CA: University of California, Department of Information and Computer Science.
- Breiman, L. (1996): Bagging predictors. *Machine Learning* 24: 123-140
- Duda, R. O., & Hart, P. E. (1973): Pattern classification and scene analysis. New York: John Wiley
- Fayyad, U.M., & Irani, K.B. (1993): Multi-interval discretization of continuous-valued attributes for classification learning. *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence* (pp. 1022-1027), San Mateo, CA: Morgan Kaufmann.
- Kohavi, R. (1996): Scaling up the Accuracy of Naive-Bayes Classifiers: A Decision-Tree Hybrid. *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining* (pp. 202-207), Menlo Park, CA: AAAI Press.
- Quinlan, J. R. (1986): Induction of decision trees. *Machine Learning* 1: 81-106
- Quinlan, J. R. (1993): C4.5: Programs for machine learning. Morgan Kaufmann
- Zheng, Z., Webb, G. I. (2000): Lazy learning of Bayesian Rules. *Machine Learning* 41: 53-87