SNNB: A Selective Neighborhood based Naïve Bayes for Lazy Learning

Zhipeng XIE Wynne HSU Zongtian LIU Mong Li LEE

[1]School of Computing
National University of Singapore
Lower Kent Ridge Road, Singapore, 119260
{xiezp, whsu, leeml}@comp.nus.edu.sg

[2]School of Computing
Shanghai University of China
Shanghai, P.R.China, 200072
ztliu@mail.shu.edu.cn

Abstract. Naive Bayes is a probability-based classification method which is based on the assumption that attributes are conditionally mutually independent given the class label. Much research has been focused on improving the accuracy of Naïve Bayes via eager learning. In this paper, we propose a novel lazy learning algorithm, Selective Neighbourhood based Naïve Bayes (SNNB). SNNB computes different distance neighborhoods of the input new object, lazily learns multiple Naïve Bayes classifiers, and uses the classifier with the highest estimated accuracy to make decision. The results of our experiments on 26 datasets show that our proposed SNNB algorithm outperforms Naïve Bayes, and state-of-the-art classification methods NBTree, CBA, and C4.5 in terms of accuracy as well as efficiency.

Key words: Naïve Bayes, Classification, Lazy Learning

1 Introduction

Naive Bayes [5] is a probability-based classification method which is based on the assumption that attributes are conditionally mutually independent given the class label. Although simple, Naïve Bayes has surprisingly good performance in a wide variety of domains, including many domains where there are clear dependencies between the attributes. Naïve Bayes is also robust to noise and irrelevant attributes and the learnt theories are easy for domain experts to understand. As a result, Naïve Bayes has attracted much attention from researchers. Research work to extend the Naïve Bayes can be broadly divided into three main categories.

The first category aims to improve Naïve Bayes by transforming the feature space such as feature subset selection and constructive feature. Kononenko's semi-naïve Bayesian classifier [10] performs exhaustive search by iteratively joining pairs of attribute values to generate constructive features based on statistical tests for independence. The constructive Bayesian classifier [14] employs a wrapper model to find the best Cartesian product attributes from existing nominal attributes, and

possible deletion of existing attributes. Langley and Sage [11] use the Forward Sequential Selection (FSS) method to select a subset of the available attributes, with which to build a Naïve Bayes classifier. It is shown that such attribute selection can improve upon the performance of the Naïve Bayes classifier when attributes are inter-dependent, especially when some attributes are redundant.

The second category of research extends Naïve Bayes by relaxing the attribute independence assumption. This covers many classification methods based on Bayesian network [2]. Friedman and Goldszmidt [8] explore the Tree Augmented Naïve Bayes (TAN) model for classifier learning, which belongs to a restricted sub-class of Bayesian network by inducing a tree-structure network structure.

The third category employs the principle of local learning to extend Naïve Bayes. It is well-established that large, complex databases are not always amenable to a unique global approach to generalization. This is because there may exist different models specific to a data point. A typical example in this category is the Naïve Bayes tree, NBTree [9], which uses decision tree techniques to partition the whole instance space (root node) into several subspaces (leaf nodes), and then trains a Naïve Bayes classifier for each leaf node. [17] presents the Lazy Bayesian Rule (LBR) classification method to solve the small disjunct problem of NBTree.

In addition, Zheng Zijian [16] presents a method to generate Naïve Bayes Classifier Committees by building individual naïve Bayes classifiers using different attribute subsets in sequential trials. Majority vote of the committees was applied in the classification stage. It has been shown that this method is able to improve the accuracy of Naïve Bayes by a wide margin.

On the other hand, we can divide classification methods into two types: eager learning and lazy learning, depending on when the major computation occurs [1]. Lazy learning is distinguished by spending little or no effort during training and delaying computation until classification time. On the other hand, eager learning replaces the training inputs with an abstraction expression such as rule set, decision tree, concept lattice or neural network) and use it to process queries. The majority of the methods to extend Naïve Bayes are eager, except for LBR. We observe that most existing techniques for improving the performance of the Naïve Bayesian classifier require complex induction processes.

In this paper, we propose a novel Naïve Bayes classifier, the Selective Neighborhood Naïve Bayes (SNNB), for lazy classification. SNNB constructs multiple Naïve Bayes classifiers on multiple neighborhoods by using different radius values for an input new object. It then selects the most accurate one to classifier the new object. Experimental results shows that SNNB outperforms not only the Naïve Bayes and NBTree, but also several other state-of-art classification methods.

The rest of the paper is organized as follows. Section 2 briefly reviews the Naïve Bayes method. Section 3 gives an example to motivate the work of SNNB. A detailed description and analysis of SNNB is given in Section 4. Section 5 gives the results from the performance study of SNNB. Section 6 discuss some related work and we conclude in Section 7 by highlighting our contributions.

2 Naïve Bayes and Accuracy Estimation

For simplicity, we shall assume that the dataset is a relational table with only nominal attributes and consists of the descriptions of n objects in the form of tuples. These n objects have been classified into q known classes, $C_1, C_2, \ldots, C_q$. Each object in the database is described by m distinct attributes, $Attr_1, ..., Attr_i, ..., Attr_m$. In an instantiation of object description, an attribute $Attr_i$ takes on the value $v_{ij} \in domain(Attr_i)$. Let $U=\{x_1, x_2, \ldots, x_n\}$ denote the set of objects and $A=\{Attr_1, \ldots, Attr_m\}$ denote the set of attributes. Various kinds of classification method have been developed to induce classifiers on a dataset, and the classifier can be thought as a function assigning a class label to an unclassified object.

Naïve Bayes is a probability-based classification method, built on the assumption that all the attributes are mutually independent within each class. Given an unlabelled instance $x=<v_1, \ldots, v_m>$ consisting of m attribute values, the classification technique of Naïve Bayes classifier will assign the object x to the class C_i such that the value of $P(C_i|x)$ is maximal. $P(C_i|x)$ can be calculated via Bayesian Theorem and the independence assumption as follows:

$$P(C_i|x)=\frac{P(x \mid C_i)\times P(C_i)}{P(x)} \;\mu\; P(x|C_i)\times P(C_i)=\prod_{j=1}^{m} P(A_j = v_j \mid C_i)\times P(C_i)\,. \qquad (1)$$

Note that $P(x)$ is fixed for a given x. To estimate $p(C_i)$, the simplest probability estimate, or occurrence frequency, is used. That is,

$$p(C_i) = N(C_i)/N$$

where N is the number of the training examples, and $N(C_i)$ is the number of the training examples with class C_i. To estimate the conditional probability $p(Attr_k=v_k|C_i)$, we adopt the Laplace-corrected estimate, which leads to

$$p(Attr_k=v_k|C_i)=(N(Attr_k=v_k, C_i)+f)/(N(C_i)+fn_j)$$

where n_j is the number of values of the k-th attribute, and f is a multipicative factor (default value as $1/N$) [4]

For Naïve Bayes classifier, Leave-One-Out is used to get the accuracy on training set. This can be implemented efficiently, and is linear to the number of objects, number of attributes, and number of label values [9]. For Naive Bayes classifier cls_{NB}, $acc(cls_{NB})$ is used to denote the accuracy computed through Leave-One-Out method.

3 Motivating Example

Before presenting the details of our SNNB algorithm, let us look at a simple example, for which Naïve Bayes fails while one possible solution succeeds.

Example: Suppose we are given a small dataset that comprises of 300 objects which are described by two conditional attributes, A and B. These objects are divided into two classes, d=0 and d=1. Table 1 shows that among the 50 objects with description (A=0, B=0), 45 are classified as (d=0), while 5 are classified as (d=1). Obviously, an ideal classifier should be able to classify an object (A=0, B=0) to class

(d=0), object (A=0, B=1) to class (d=1), object (A=1, B=0) to class (d=1), and object (A=1, B=1) to class (d=0). Note that for simplicity, all the probabilities are estimated with occurrence frequencies.

A	B	d=0	d=1
0	0	45	5
0	1	5	95
1	0	5	45
1	1	95	5

Table 1. Example of a dataset.

Let us first consider the classifier generated by Naïve Bayes. We have the following probabilities:

p(A=0|d=0)=1/3, p(A=1|d=0)=2/3, p(B=0|d=0)=1/3, p(B=1|d=0)=2/3
p(A=0|d=1)=2/3, p(A=1|d=1)=1/3, p(B=0|d=1)=1/3, p(B=1|d=1)=2/3
p(d=0)=p(d=1)=1/2

According to Equation (1) (in Section 2), the following results will be obtained:

(A=0, B=0) will be classified as d=1,
(A=0, B=1) will be classified as d=1,
(A=1, B=0) will be classified as d=0, and
(A=1, B=1) will be classified as d=0,

Clearly, Naïve Bayes can't produce the ideal results.

Now, let us consider the situation where we construct a naïve bayes trained for an input test example on its neighborhood including all the objects with distance no larger than 1. Thus, for an test object with description (A=0, B=0), only (A=0, B=0), (A=0, B=1), (A=1, B=0) in Table 1 will be considered as being in the neighborhood. The following probabilities can be computed on these training objects:

p(A=0|d=0)=50/55, p(A=1|d=0)=5/55
p(B=0|d=0)=50/55, p(B=1|d=0)=5/55
p(A=0|d=1)=100/145,p(A=1|d=1)=45/145
p(B=0|d=1)=50/145, (B=1|d=1)=95/145
p(d=0)=55/200, p(d=1)=145/200

Hence,

p(A=0|d=0)*p(B=0|d=0)*p(d=0) = (50/55)*(50/55)*(55/200) = 0.227
p(A=0|d=1)*p(B=0|d=1)*p(d=1) = (100/145)*(50/145)*(145/200) = 0.172

Therefore, (A=0, B=0) will be classified as d=0.

Similarly, for each other input object, through constructing a naïve bayes classifier on its 1-neighborhood, and applying this classifier to make decision for the input object, an ideal classification result can be achieved.

4 Selective Neighborhood based Naïve Bayes

We will now present the details of *SNNB*, a Selective Neighborhood based Naïve Bayesian classifier. The basic idea is to construct multiple classifiers on multiple

neighborhoods with different radius, then select out the classifier with the highest estimated accuracy to classify the new object.

For any two objects x and y, the distance between them normally can be defined as the number of the attributes on which x and y take on the different values, that is,

$$distance(x, y)=|\{Attr_i \in A | Attr_i(x) \neq Attr_i(y)\}|.$$

For an input new object x, its k-Neighborhood consists of all the objects in U with the distance to x not larger than k, denoted as

$$NH_k(x)=\{x_i \in U | distance(x_i, x) \leq k\}$$

We also call the Naïve Bayes classifier, k-NB_x, *trained* on $NH_k(x)$ as the k-th local Naïve classifier. Clearly, for any input object x, m-NB_x is trained on the whole object set U, so it is also called the global Naïve Bayes classifier. The pseudo-code for SNNB is given below, where OWD is an array of sets of objects, and OWD[*dist*], $0 \leq dist \leq m$, stores all the objects in U that has distance dist to the input object x.

```
Algorithm SNNB

input: Training set T,
       the trained global NB classifier CLS_global,
       unknown new object x;
output: the predicted class of x;

begin
1  Add t into OWD[distance(t,x)] for each t∈T;
2  k=|A|; j=0;
3  total=|T|;
4  k-NB=CLS_global; NH_k=T;
5  Candidates={k-NB};
6  while (true)
7    count=0;
8    while(count<(1-θ)×total)//θ is set as 0.5 defaultly
9      count+=|OWD[k]|;
10     k--;
11   endwhile;
12   if (count<ϕ×|T|) break; endif;
13   j++;
14   NH_k=OWD[0]∪OWD[1]∪…∪OWD[k];
15   total=|NH_k|;
16   Train a Naïve Bayes classifier k-NB on NH_k;
17   Add k-NB into Candidates;
18 endwhile;
19 Select out the classifier q-NB with the maximal
   value of acc(q-NB) from Candidates;
20 return q-NB(x);
end;
```

The algorithm SNNB consists of three main steps. The first step calculates the distance between the input new object x and each training object t in training set T, and stores all the training objects according to their distances (line 1). The second step constructs a series of NB classifiers on different subsets of training objects (line 2-line 18). The last step classifies the new object with the most accurate NB classifier (line 19-line 20). There are two parameters in the algorithm: one is the support difference threshold, θ, with 0.5 as the default value; the other is the support threshold ϕ, with

0.03 as the default value. The support threshold is to ensure the generalizing ability of learnt model, while the support difference threshold is mainly for controlling the speed of the algorithm.

Complexity Analysis

We will now give an analysis of the complexity of the algorithm. Let m be the number of attributes and n be the number of objects. It is obvious that the complexities of the first step and the third step are O(m×n) and O(m) respectively. We will now examine the complexity of the second step.

Fact: Given a set TS of training objects, the complexity of inducing a NB classifier CLS and estimating its accuracy with Leave-1-out is O(|TS|×m).

Suppose the generated series of NB classifiers is CLS_1, CLS_2, …, CLS_p, which are trained on TS_1, TS_2, …, TS_p respectively. From lines 8-10, we know that

(1) $|TS_1| \le \theta \times n$, and

(2) $|TS_{i+1}| \le \theta \times |TS_i|$ for i = 1, 2, …, p–1

That is, $|TS_i| \le \theta^i \times n$ for i=1, 2, …, p–1. According to line 12, we also have $|TS_p| = \theta^p \times n \ge \phi \times n$, that is $p \le \log_\theta \phi$. Given the values of θ and φ, $\log_\theta \phi$ is a constant value. Hence, the complexity of the second step is also O(m×n).

5 Experimental Results

Dataset	No. Attrs	No. Classes	Size	Dataset	No. Attrs	No. Classes	Size
anneal	38	6	798	australian	14	2	690
auto	25	7	205	breast-w	10	2	699
cleve	13	2	303	crx	15	2	690
diabetes	8	2	768	german	20	2	1000
glass	9	7	214	heart	13	2	270
hepatitis	19	2	155	horse	22	2	368
hypo	25	2	3163	ionosphere	34	2	351
iris	4	3	150	labor	16	2	57
led7	7	10	3200	lymph	18	4	148
pima	8	2	768	sick	29	2	2800
sonar	60	2	229	tic-tac-toe	9	2	958
vehicle	18	4	846	waveform	21	3	5000
wine	13	3	178	zoo	16	7	101

Table 2. Datasets used in the experiments.

We carried out an empirical comparison of the algorithm SNNB by using the 26 datasets from UCI Machine Learning Repository [13]. The characteristics of these datasets are listed in Table 2. Since the current version of SNNB can only deal with nominal attributes, the entropy-based discretization algorithm [6] is used for pre-processing.

5.1 Error-rate comparison

We first compare the accuracy results of SNNB with Naïve Bayes, and three other state-of-art classification methods[1]:

- NBTree in [9] (a state-of-art hybrid classification method to improves the accuracy of Naïve Bayes),
- CBA in [12] (a classification method based on association rules), and
- C 4.5 Rules (Release 8) [15].

The error rates of the different algorithms on the experimental domains are listed in Table 3. All the error rates are obtained through 10-fold cross validation. We use the same train/test set split for different classification methods in the experiments. Throughout the experiment, the parameters of SNNB are set as default values without adjusting.

Dataset	NBTree	CBA	C4.5Rules	NB	SNNB
anneal	**1.0 (1)**	2.1 (4)	5.2 (5)	1.6 (3)	1.4 (2)
australian	14.5 (2)	14.6 (3)	15.3 (5)	**14.1 (1)**	14.8 (4)
auto	22.8 (4)	**19.9 (1)**	**19.9 (1)**	27.7 (5)	22.5 (3)
breast-w	2.6 (2)	3.7 (4)	5.0 (5)	**2.4 (1)**	3.0 (3)
cleve	19.1 (4)	**17.1 (1)**	21.8 (5)	18.1 (2)	18.5 (3)
crx	14.2 (2)	14.6 (4)	15.1 (5)	14.5 (3)	**13.9 (1)**
diabetes	**24.1 (1)**	25.5 (4)	25.8 (5)	**24.1 (1)**	**24.1(1)**
german	**24.5 (1)**	26.5 (4)	27.7 (5)	**24.5 (1)**	26.2 (3)
glass	28.0 (2)	**26.1 (1)**	31.3 (5)	28.5 (4)	28.0 (2)
heart	**17.4 (1)**	18.1 (2)	19.2 (5)	18.1 (2)	18.9 (4)
hepatitis	**11.7 (1)**	18.9 (4)	19.4 (5)	15.6 (3)	14.3 (2)
horse	18.7 (4)	17.6 (3)	**17.4 (1)**	21.7 (5)	**17.4 (1)**
hypo	1.0 (2)	1.0 (2)	**0.8 (1)**	1.8 (4)	1.8 (4)
ionosphere	12.0 (5)	**7.7 (1)**	10.0 (2)	10.5 (3)	10.5 (3)
iris	7.3 (5)	5.3 (2)	**4.7 (1)**	5.3 (2)	5.3 (2)
labor	12.3 (3)	13.7 (4)	20.7 (5)	5.0 (2)	**3.3 (1)**
led7	26.7 (3)	28.1 (5)	**26.5 (1)**	26.7 (3)	**26.5 (1)**
lymph	17.6 (3)	22.1 (5)	**16.5 (1)**	19.0 (4)	17.0 (2)
pima	24.9 (3)	27.1 (5)	**24.5 (1)**	**24.5 (1)**	25.1 (4)
sick	22.1 (5)	2.8 (2)	**1.5 (1)**	4.2 (4)	3.8 (3)
sonar	22.6 (4)	22.5 (3)	29.8 (5)	21.6 (2)	**16.8 (1)**
tic-tac-toe	17.0 (4)	**0.4 (1)**	0.6 (2)	30.1 (5)	15.4 (3)
vehicle	29.5 (3)	31 (4)	**27.4 (1)**	40.0 (5)	28.4 (2)
waveform	**16.1 (1)**	20.3 (4)	21.9 (5)	19.3 (3)	17.4 (2)
wine	2.8 (3)	5.0 (4)	7.3 (5)	**1.7 (1)**	**1.7 (1)**
zoo	5.9 (4)	3.2 (2)	7.8 (5)	3.9 (3)	**2.9 (1)**
Average	16.02	15.19	16.29	16.33	**14.57**

Table 3. Experiment results on the error rates of classifiers

From table 3, we have observed the following facts:

[1] These classification systems are all available from Web, where NBTree is implemented in the MLC Utilities available from http://www.sgi.com/tech/mlc/, CBA is downloadable from http://www.comp.nus.edu.sg/~dm2/, and C4.5 from http://www.cse.unsw.edu.au/~quinlan/.

(1) SNNB obtains lower error rates than Naïve Bayes in 15 out of the 26 domains, and higher error rates in 6 domains. It also obtains lower error rates than CBA in 16 domain and higher error rates in 9 domains. When compared with NBTree and C4.5Rules, SNNB gets the same sore, winning in 15 domains and losing in 9 domains.

(2) To give further insight into the experimental results, for each dataset, all the classification methods can be sorted from the lowest error rate to the highest error rate. The ranking of each method is recorded in the parentheses, where number *i* (1£*i*£5) means that the method gets the *i*-th lowest error rate among the five methods. Such information has been summarized into the following table. SNNB gets the lowest error rates on 8 domains, the sencond lowest error rates on 7 domains, the third lowest error rates on 7 domains, the fourth lowest error rates on 4 domain, and doesn't get the worst error rate on any domain. We can also conclude that SNNB is better than NB, NBTee, and CBA. As to the C4.5Rules, it gets the lowest error rates on 9 domains, which is one more than SNNB, but we can see that C4.5Rules also gets the worst error rates on 15 domains.

	1st	**2nd**	**3rd**	**4th**	**5th**
NBTree	6	5	6	6	3
CBA	5	5	3	10	3
C4.5Rules	9	2	0	0	15
NB	6	5	7	4	4
SNNB	8	7	7	4	0

Table 4. Summary of ranking information

(3) With the comparison of the average accuracies, SNNB also produces more accurate classifiers than Naïve Bayes, NBTree, C4.5Rules, and CBA. The geometric mean error ratio shows that NB has 10.8% higher error, C4.5Rules 10.6% higher error, NBTree 9.1% higher error, and CBA 4.1% higher error than SNNB.

5.2 Computational Requirements

At the end of section 4, we have already shown the time complexity of running SNNB to classify a new object is linear with the number of training objects and the number of attributes. To get an intuitionistic idea of the computational requirements of SNNB, the table 5 records the average time of SNNB in CPU seconds on the personal computer (Pentium III 700Mhz with 128M memory) for classifying each object. We have not recorded the training time used to induce a global Naive Bayes classifier, because this process is always very fast.

In table 5, when θ set as 0.5, the three worst cases are 0.1477 seconds, 0.1471 seconds, and 0.1402 seconds in the hypo domain, sick domain and waveform domain. These three domains have 3163, 2800, and 5000 objects respectively, and they also have 25, 29, and 21 attributes respectively.

Dataset	Support-Difference Threshold q			
	θ=0.3	θ=0.5	θ=0.7	θ=0.9
anneal	0.0496	0.0607	0.0868	0.1475
australian	0.0092	0.0110	0.0148	0.0223
auto	0.0074	0.0101	0.0156	0.0324
breast-w	0.0063	0.0073	0.0102	0.0157
cleve	0.0051	0.0058	0.0070	0.0098
crx	0.0098	0.0120	0.0159	0.0249
diabetes	0.0075	0.0087	0.0105	0.0138
german	0.0203	0.0240	0.0310	0.0429
glass	0.0033	0.0042	0.0048	0.0092
heart	0.0046	0.0053	0.0070	0.0091
hepatitis	0.0038	0.0046	0.0058	0.0099
horse	0.0070	0.0088	0.0124	0.0206
hypo	0.1315	0.1477	0.1798	0.2734
ionosphere	0.0104	0.0131	0.0184	0.0347
iris	0.0007	0.0007	0.0012	0.0015
labor	0.0011	0.0019	0.0021	0.0021
led7	0.0371	0.0520	0.0812	0.1274
lymph	0.0028	0.0047	0.0081	0.0104
pima	0.0068	0.0081	0.0094	0.0123
sick	0.1315	0.1471	0.1735	0.2396
sonar	0.0164	0.0190	0.0235	0.0357
tic-tac-toe	0.0066	0.0089	0.0125	0.0176
vehicle	0.0141	0.0199	0.0312	0.0646
waveform	0.1103	0.1402	0.2057	0.3432
wine	0.0025	0.0037	0.0052	0.0067
zoo	0.0018	0.0028	0.0060	0.0113
Average	0.0234	0.0282	0.0377	0.0592

Table 5. Average time to classify a new object (in seconds)

6 Related Work

Although the underlying mechanism in SNNB is lazy, SNNB is closely related to the Naïve Bayes Tree method. Let us consider two kinds of neighborhood,

(1) Feature-based: For any object x and a feature subset F of its description, the F-neighborhood of x consists of all the objects containing the feature subset F.

(2) Distance-based: For any object x and a distance d, the d-Neighborhood of x consists of all the objects whose distances are less than d.

Now, it is clear that SNNB uses the Naïve Bayes classifier trained on a selective distance-based neighborhood of the input test object to make classification. On the other hand, NBTree uses the Naïve Bayes classifier trained on a selective feature-based neighborhood of the input test object to make classification.

Furthermore, our method is also related to the k-nearest neighbor (k-NN) algorithm [3], which is one of the most venerable algorithms in machine learning. Both SNNB and k-NN are based on distance (or similarity). The k-NN algorithm can be decomposed into two phases: Phase 1 determines a neighborhood which is formed by

the nearest *k* neighbors, while Phase 2 applies a simple classifier (Majority Voting) to classify the new object. If we replace the simple classifier (Majority voting in original *k*-NN algorithm) with Naïve Bayes, we can find that SNNB is also similar with the *k*-NN algorithm, and SNNB can be viewed as a hybrid method of Naïve Bayes and Nearest Neighborhood.

7 Conclusion

In this paper, we have developed a selective neighborhood-based Naive Bayes algorithm, SNNB. The contribution of SNNB is that it lazily constructs a set of Naïve Bayes classifiers, and chooses one of them to make decisions. Experimental results demonstrate that our proposed algorithm is not only able to improve the accuracy of Naïve Bayes, but it is also able to outperform existing state-of-art classification methods such as NBTree, CBA and C4.5. We have also shown that SNNB is computationally efficient.

Acknowledgements
We thank the anonymous reviewers for their suggestions. This work is supported by the NSTB-NUS research project R-252-000-102-112 and R-252-000-102-303. The third author's work was partly supported by National Natural Science Fund of China (No. 69985004).

References

[1] Aha, D.W. Lazy Learning. Dordrecht: Kluwer Academic, 1997
[2] Cheng, J., & Greiner, R. Comparing Bayesian network classifiers. in Proceedings of the fifteenth conference on uncertainty in artificial intelligence, 1999
[3] Cover, T.M., & Hart, P.E. Nearest neighbor pattern classification. IEEE Transactions on Information Theory, 1996, vol. 13, pp. 21-27
[4] Domingos, P., & Pazzani, M. On the optimality of the simple bayesian classifier under zero-one loss. Machine Learning, 1997, Vol. 29, pp. 103-130
[5] Duda, R.O., & Hart, P.E. Pattern Classification and Scene Analysis. New Yaork: John Wiley, 1973
[6] Fayyad, U.M, & Irani, K.B. Multi-interval discretization of continuous-valued attributes for classification learning. IJCAI-93, pp. 1022-1027
[7] Friedman, J.H., Kohavi, R., and Yun, Y. Lazy decision trees. In Proceedings of the Thirteenth National Conference on Artificial Intelligence, 1996, pp. 717—724
[8] Friedman, N., & Goldszmidt, M. Building classifiers using Bayesian networks. Proceedings of the Thirteenth National Conference on Artificial Intelligence, 1996, pp. 1277-1284
[9] Kohavi R. Scaling Up the Accuracy of Naïve-Bayes Classifiers: a Decision-Tree Hybrid. in Simoudis E. & Han J. (eds.), KDD-96: Proceedings Second International Conference on Knowledge Discovery & Data Mining, AAAI Press/MIT press, Cambridge/Menlo Park, pp. 202-207, 1996.
[10] Kononenko, I. Semi-naïve Bayesian classifier. in Proceedings of European Conference on Artificial Intelligence, 1991, pp. 206-219

[11] Langley, P., & Sage, S. Induction of selective Bayesian classifiers. in Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence, 1994, pp. 339-406
[12] Liu, B., Hsu, W., and Ma, Y. Integrating Classification and Association Rule Mining. Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining. New York, USA, 1998.
[13] Merz, C.J., and Murphy, P. UCI repository of machine learning database [http://www.cs.uci.edu/~mlearn/MLRepository.html], 1996
[14] Pazzani, M. Constructive induction of Cartesian product attributes. in Proceedings of the Conference ISIS96: Information, Statistics and Induction in Science, 1996, pp. 66-77
[15] Quinlan, J.R. C4.5: Programs for Machine Learning. Morgan Kaufmann, 1993
[16] Zheng, Z. Naïve Bayesian classifier committees. in Proceedings of European Conference on Machine Learning, 1998, pp. 196-207
[17] Zheng, Z. and Webb, G.I. Lazy Learning of Bayesian Rules. Machine Learning, 2000, Vol. 41(1), Kluwer Academic Publishers, pp. 53-84