Discriminative Naive Bayesian Classifiers

Kaizhu Huang
Department of Computer Science and Engineering
The Chinese University of Hong Kong
Shatin, N.T. Hong Kong SAR
kzhuang@cse.cuhk.edu.hk

April 22, 2003

Abstract

Discriminative classifiers such as Support Vector Machines (SVM) directly learn a discriminant function or a posterior probability model to perform classification. On the other hand, generative classifiers often learn a joint probability model and then use Bayes rules to construct a posterior classifier from this model. In general, generative classifiers are not as accurate as discriminative classifiers. However generative classifiers provide a principled way to deal with the missing information problems, which discriminative classifiers cannot easily handle. To achieve good performances in various classification tasks, it is better to combine these two strategies. In this paper, we develop a method to train one of the popular generative classifiers, the Naive Bayesian classifier (NB) in a discriminative way. We name this new model as Discriminative Naive Bayesian classifier. We give theoretic justification, outline of the algorithm, discussion of the extension to the general Bayesian classifiers, and perform a serious of experiments on benchmark real-world datasets to demonstrate our model's advantages. The Discriminative Naive Bayesian Classifier is shown to have a similar optimization form as SVM. Its performance outperforms NB in classification tasks and outperforms SVM in handling missing information tasks.

1 Introduction

Generative classifiers have showed their advantages to deal with missing information problems in many classification tasks, even though their overall performances are not as good as discriminative classifiers such as Support Vector Machines [19]. A typical example of generative classifiers is the Naive Bayesian classifier[6] [10]. Under a conditional independency assumption, i.e., $P(A_i, A_j|C) = P(A_i|C)P(A_j|C)$, for $1 \leq i \neq j \leq n$, NB classifies a

new data x into the class with the largest posterior probability as shown in Eq. (1), where A_i, A_j represent the attributes or variables, C is the class variable, n is the number of the attributes. Further as in Eq. (2) this posterior classification rule can be transformed into joint probability classification rule, since $P(A_1, A_2, \ldots, A_n)$ for a given data is a constant w.r.t C. Finally, combining the independency assumption, the classification rule is changed into a decomposable form as Eq. (3).

$$\begin{aligned} c &= \arg\max_{C_i} P(C_i|A_1, A_2, \ldots, A_n) && (1) \\ &= \arg\max_{C_i} \frac{P(C_i)P(A_1, A_2, \ldots, A_n|C_i)}{P(A_1, A_2, \ldots, A_n)} \\ &= \arg\max_{C_i} P(C_i)P(A_1, A_2, \ldots, A_n|C_i) && (2) \\ &= \arg\max_{C_i} P(C_i)\prod_{j=1}^{n} P(A_j|C_i) && (3) \end{aligned}$$

When used in real application, NB firstly partitioned the dataset into several sub-datesets by the class label. According to Maximum Log-likelihood criterion, in each sub-dataset labelled by C_i, $P(A_j = a_{jk}|C_i)$ is easily estimated by the frequency n_{ijk}/n_i, n_{ijk} is the number of the occurrences of the event $\{A_j = a_{jk}\}$ in sub-dateset C_i, n_i is the number of the samples in sub-dateset C_i.

This kind of simple scheme achieves a surprising success in many classification tasks [6] [10] [7]. Importantly, a great advantage for NB is its immediate ability to deal with missing information problems. Assume the attributes set $\{A_1, A_2, \ldots, A_n\}$ as A, when the information of a subset of A, for example T, is unknown, the marginalization inference can be obtained immediately as follows:

$$\begin{aligned} c &= \arg\max_{C_i} \sum_{T} P(C_i)P(A-T, T|C_i) \\ &= \arg\max_{C_i} P(C_i)P(A-T|C_i) \\ &= \arg\max_{C_i} P(C_i)\prod_{j\in A-T} P(A_j|C_i). && (4) \end{aligned}$$

No further computation is needed in handling this missing information problem, since each term $P(A_j|C_i)$ has been obtained in training the NB.

However, problems exist for NB. This model separately model the joint probability in each subset and then apply Bayes rule to construct posterior classifiers. This kind of framework seems to be incomplete, since this construction procedure actually discards some important discriminative information for classification. With no consideration of the other data with different class label, this method only tries to approximate the information in each sub-dataset. On the other hand, the discriminative classifiers preserver this information well by directly constructing decision rules among all the data. Therefore, for Naive Bayesian classifier, it is not enough to approximate the date in each sub-dateset separately. It should provide a global scheme to preserve the discriminative information among all the data.

One of the remedy method is to directly learn a posterior probability model rather than a joint probability model. However in the framework of Bayesian network classifier, this kind of approaches are often computationally hard to perform the optimization. Even for the simple Naive Bayesian classifiers, the corresponding posterior learning, known as Logistic Regression (LR) will encounter difficult problems for dealing with missing information tasks. In a two-category classification problem, LR defines the posterior probability as $P(c = C_0|A_1, A_2, \ldots, A_n) = 1/(1 + exp(-\sum_{j=1}^{n} \beta_j A_j - \theta))$, $P(c = C_1|A_1, A_2, \ldots, A_n) = 1 - P(c = C_0|A_1, A_2, \ldots, A_n)$. Similarly, ML criterion can be used to find the parameters β and θ. When the values of a subset of attribute set T is unknown, the marginalization on T as shown in Eq.(5):

$$P(c = C_0|A - T) = \frac{\sum_T P(c = C_0|A)P(A - T, T)}{\sum_T P(c = C_0|A)P(A - T, T) + \sum_T P(c = C_1|A)P(A - T, T)} \quad (5)$$

The right hand side will be hard to calculate. Firstly, $P(A - T, T)$ varies from T, thus it cannot be omitted. Secondly, $P(c = C_0|A)$, the logistic form will be at least calculated $r^{|T|}$ times , where r is the minimum number of values of attributes, $|T|$ represents the cardinality of set T. This calculation will be computationally intractable when the number of missing attributes is big.

In this paper, we develop a novel method to train the Naive Bayesian classifier, in a discriminative way. Beginning with modelling the joint probabilities for the data, we add

into the optimization function a penalty item which describes the divergence between two classes. On one hand, the optimization on the new function not only tries to approximate the dataset as accurately as possible. On the other hand, it also tries to enlarge the divergence among classes as big as possible. Importantly, when improving the accuracy, this model also inherit the NB's ability in handling missing information problems.

This paper is organized as follows. In next section, we present a short review on the related work. In Section 3 we describe the discriminative Naive Bayesian classifiers in detail. We then in Section 4 evaluate our algorithm on five benchmark datasets. In section 5, the relationship between our algorithm and other approaches, such as SVM, and Fisher Discriminant, are discussed. In this section, we also give an algorithm to extend our algorithm to a tree-like Bayesian network. Finally, in Section 6, we conclude this paper with remarks.

2 Related work

Combining generative classifiers and discriminative classifiers has been one of an active topics in machine learning. A lot of work [1, 8, 18, 2] has been done in this area. However nearly all of these methods are designed for Gaussian Mixture Model [12] and Hidden Markov Model [15]. By contrast, our discriminative approaches are developed for one of Bayesian network classifiers, Naive Bayesian classifiers. On the other hand, Jaakkola et. al. developed a method to explore generative models from discriminative classifiers [17]. Different with this approach, our method performs a reverse way to use discriminative information in generative classifiers.

3 Discriminative Naive Bayesian Classifiers

In this section, we first develop the discriminative Naive Bayesian classifier in a two-category classification tasks. Then we in Subsection 3.2, based on a voting scheme, we present an extension of our method into multi-category classification tasks.

3.1 Two-category Discriminative Naive Classifiers

The NB firstly partitions the dataset into several sub-datasets by the class variable. Typically, in a two-category classification problem, two sub-datasets S_1 and S_2 will be created respectively for class label C_1 and C_2. Then in each sub-dataset, an ML or cross entropy criterion will be used to find the optimal values for the parameters, namely, $P(A_j|C_1)$ and $P(A_j|C_2)$, $1 \leq j \leq n$. The cross entropy between a distribution p and a reference distribution q is defined as a Kullback–Leibler form shown in the following:

$$KL(q,p) = \sum q \log \frac{q}{p} \tag{6}$$

In the framework of Bayesian learning, the reference distribution is normally the empirical distribution. Therefore for NB, the optimization function in a two-category classification problem can be written:

$$\{P_1, P_2\} = \arg\max_{\{p_1,p_2\}}(KL(p_1,\hat{p_1}) + KL(p_2,\hat{p_2})) \tag{7}$$

$\hat{p_1}$ and $\hat{p_2}$ represent the empirical distribution for Sub-dataset 1 and Sub-dataset 2 respectively. The first term and second term on the right hand side of Eq. (7) describe how accurately the joint distributions p_1 and p_2 approximate the sub-dataset 1 and sub-dataset 2. It is observed again that this function is incomplete, since only the inner-class information is preserved. The important inter-class information, namely, the divergence information,between class 1 and class 2 is discarded actually. To fix this problem, we add into the optimization function an interactive term, which represents the divergence between classes.

$$\begin{aligned} \{P_1, P_2\} &= \arg\min_{\{p_1,p_2\}} f(p_1,p_2) \\ &= \arg\min_{\{p_1,p_2\}}(KL(p_1,\hat{p_1}) + KL(p_2,\hat{p_2}) + W \cdot Div(p_1,p_2)) \end{aligned} \tag{8}$$

$Div(p_1,p_2)$ is a function of the divergence between p_1 and p_2. This function value needs to go up as the divergence goes down. W is a penalty parameter. In this paper, we use the reciprocal of the Kullback–Leibler measure to represent the function.

$$Div(p_1,p_2) = \frac{1}{\sum_x p_1 \log \frac{p_1}{p_2}} \tag{9}$$

Optimization on this function will make the inner-divergence described in the first two terms on the right hand side as small as possible while the inter-class divergence among classes will be as big as possible, which will benefit the classification greatly. Different from the discriminative classifiers such as the LR, the discriminative information is finally incorporated into the joint probability p_1 and p_2. Thus the advantages of using joint probabilities will be naturally inherited into the discriminative Naive Bayesian classifier.

However, the disadvantage of adding into this interactive item is that we cannot optimize p_1 and p_2 as in NB separately in the sub-dataset 1 and sub-dataset 2. To clarify this problem, we combine the NB assumption to expand the optimization function into a complete form:

$$\min_{\{p_1,p_2\}} \quad \sum_{c=1}^{2}\sum_{j=1}^{n}\sum_{A_j}[\hat{p_c}(a_{jk})\log\frac{p_c(\hat{a_{jk}})}{p_c(a_{jk})}] + W\cdot\frac{1}{\sum_{j=1}^{n}\sum_{A_j}p_1(a_{jk})\log(p_1(a_{jk})/p_2(a_{jk}))}, \tag{10}$$

$$s.t. \quad 0 \le p_c(a_{jk}) \le 1, \tag{11}$$

$$\sum_{A_j} p_c(a_{jk}) = 1, \quad c = 1, 2; j = 1, 2, \ldots n. \tag{12}$$

$p_c(a_{jk})$ is the short form of $p_c(A_j = a_{jk})$. So does $\hat{p_c}(a_{jk})$. p_1 and p_2 are a set of parameters, namely, $p_1 = \{p_1(A_j), 1 \le j \le n\}$, $p_2 = \{p_2(A_j), 1 \le j \le n\}$. This is a nonlinear optimization problem under linear constraints. p_1 and p_2 are interactive variables. It is clear that they cannot be separately optimized as in Eq.(7).

To solve this problem, we use a modified Rosen's Gradient Projection Method [16]. We firstly calculate the gradient of the optimization function w.r.t p_1 and p_2 as Eq. (13). We then project this gradient on the constraint plane. In our problem the projection matrix can be written as Eq. (16). The optimal step length α is searched in the projected gradient direction by using Quadratic Interpolation method [11]. The process is repeated until a local minimal is obtained. We write down the detailed steps as follows:

$$\frac{\partial f}{\partial p_1(a_{jk})} = -\hat{p_1}(a_{jk})/p_1(a_{jk}) - \frac{W}{Z}[1 + \log(p_1(a_{jk})/p_2(a_{jk}))]; \tag{13}$$

$$\frac{\partial f}{\partial p_2(a_{jk})} = -\hat{p_2}(a_{jk})/p_2(a_{jk}) + \frac{W}{Z}p_1(a_{jk})/p_2(a_{jk}); \tag{14}$$

$$Z = \sum_{i=1}^{n}\sum_{A_j}\log\frac{p_1(a_{jk})}{p_2(a_{jk})}; \tag{15}$$

$$M = I - A(A'A)^{-1}A', \tag{16}$$

where, A is the coefficient matrix for the constraint.

1: Calculate the gradient according to Eq. 13.

2: Project the gradient into the constraint plane:$\nabla f^M = \nabla f \cdot M$.

3: Search the optimal step length α by Cubic Interpolation method.

4: Update the p_1, p_2 by the following Equations.

$$p_1(a_{jk})^{new} = p_1(a_{jk})^{old} - \alpha \nabla f^M_{1jk}; \tag{17}$$

$$p_2(a_{jk})^{new} = p_1(a_{jk})^{old} - \alpha \nabla f^M_{2jk}; \tag{18}$$

5: goto step 1 until p_1 and p_2 converge.

3.2 Extended to Multi-category Discriminative Naive Classifiers

We use a partly-connect committing machine scheme to extend the two-category classification problem into the multi-category one. We construct a two-category classifier for each pair of classes. For an m-category problem, in total, $m(m-1)/2$ classifiers will be constructed. Each classifier will output a probability on how confident its voting is. We then sum up the voting probabilities for each class and return the class with the highest probabilities as the final decision. In Fig. (3.2), we illustrate a four-category committing machine. In Fig. (3.2), totally $4 \times 3/2 = 6$ DNB two-category classifiers are constructed. Then these classifiers output the confidence on the class they are voting for. These confidences or probabilities are summed up for each class. Finally, the class with the maximum confidence is outputted as the classification result.

4 Evaluations

In this section, we implement the DNB algorithm to evaluate its performance on five benchmark datasets from Machine Learning Repository in UCI [13]. The detailed information for

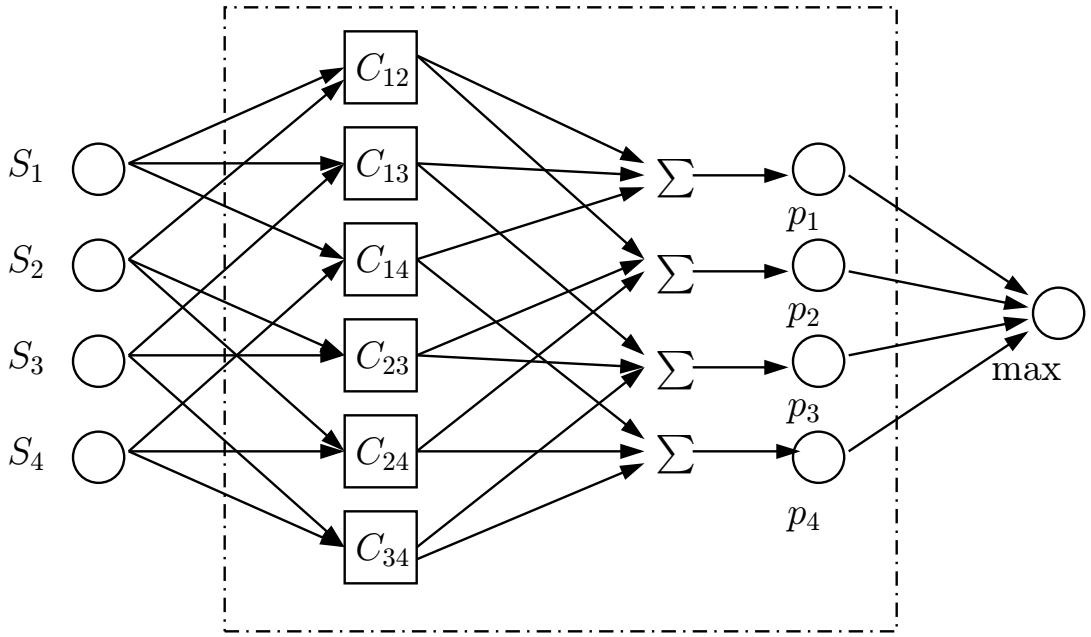

Figure 1: Discriminative Naive Bayesian Classifier Committing Machine for a four-category problem. S_i, $1 \leq i \leq 4$ represent the sub-dataset for category i respectively. $C_l m$, $1 \leq l \leq 3$, $l < m$ means the the two-category Discriminative Naive Beyesian classifier for category l and category m.

these datasets are listed in Table 1. We intentionally choose these datesets which vary in variable number and sample number. As observed in Table 1, the variable number ranges from 4 to 60 and the sample number varies from 150 to 6435. The diversity in choosing the datesets will make the evaluations on the algorithms more reliably. For the datesets with a small number of samples such as Iris and Vote, we use a five-fold Cross-Validation method [9] to test the performance. We compare our model's performance with NB and SVM in two cases, namely the case without information missing and the case with information missing. The parameters for DNB and SVM are used in the experiments are listed in Table 2.

4.1 Without information missing

We first implement our model in the case without information missing. The experimental results are demonstrated in Table 3. It can be observed that DNB outperforms NB in all of the five datesets, which implies incorporating discriminative information in training

Table 1: Description of data sets used in the experiments

Dataset	♯Variables	♯Class	♯Train	♯Test
Iris	4	3	150	CV-5
Vote	15	2	435	CV-5
Segment	19	7	2310	30%
Satimage	36	6	4435	2000
DNA	60	3	2000	1186

Table 2: Parameters used in the experiments

Method	Penalty parameter	Kernel function
DNB	1000	N/A
SVM	1000	3-order polynomial

the generative models benefits the classification greatly. When compared with SVM, DNB wins in two of the datesets while it loses in three of the others. Especially in Segment and Setimage, SVM performs significantly better than DNB. This demerit of DNB roots in the inner scheme of generative classifiers. Actually this kind of demerit is traded off with its merit when handling the missing information cases. In Section 5.1. We will present a detailed discussion on this issue.

Table 3: prediction accuracy without information missing(%)

Dataset	NB	DNB	SVM
Iris	93.33	**97.33**	95.33
Vote	90.11	93.33	**94.77**
Segment	88.44	90.88	**95.96**
Satimage	80.65	82.65	**87.90**
DNA	94.44	**94.52**	94.35

4.2 With information missing

It is important to discuss the ability of DNBs in handling the missing information issues, since one of the main advantages for generative classifiers lies in this point. Gradually, we increase the percentages of the number of unknown or missing attributes. We then test the recognition rate on these datasets with different percentages. As mentioned previously for DNB and NB, a principled way to handle missing information is using marginalization

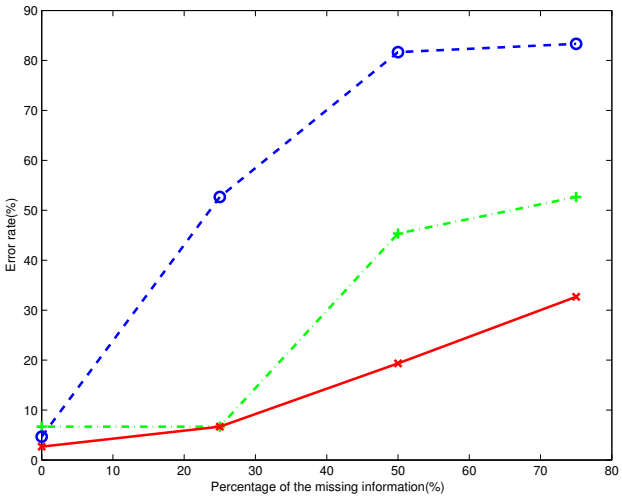

Figure 2: Error rates in Iris with information missing.

inference Eq. (19) under uncertainty. For SVM, a norm way to force its application in missing information tasks is simply setting zeros values for the missing attributes. The experiment results for the five datasets are shown in Fig. 2 to Fig. 6.

$$c = \arg\max_{C_i} P(C_i) \prod_{j \in A-T} P(A_j|C_i). \tag{19}$$

It is shown that NB demonstrates a robust ability to handle the missing information problem. In five experiments, the error rate curves do not go up until 40% attributes are unknown. Further, the DNB shows a similar resistance ability while its accuracy is higher than NB. This superiority is especially prominent in small datasets such as Iris and Vote datasets. In the datasets with small number of samples, the ML approximation is not reliable, since the distribution from insufficient training data perhaps cannot represent the true distribution. Therefore, the discriminative item will contribute more in constructing the classifiers. On the other hand, the SVM's performance gradually runs down as the missing extent goes up.

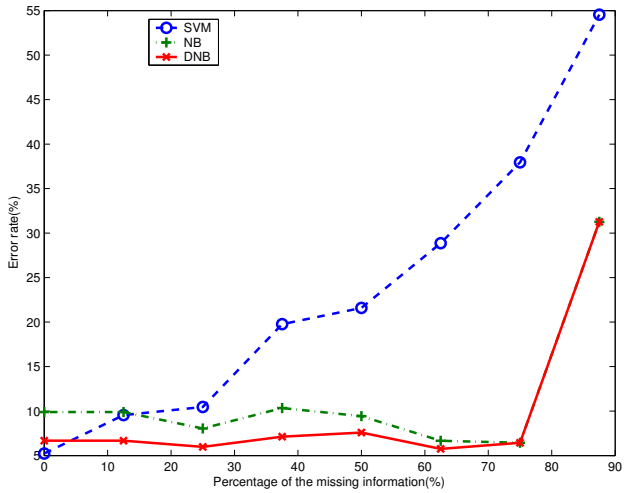

Figure 3: Error rates in Vote with information missing.

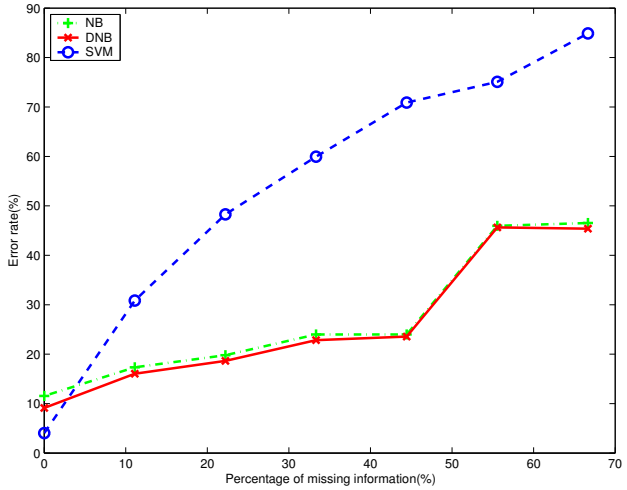

Figure 4: Error rates in Segment with information missing.

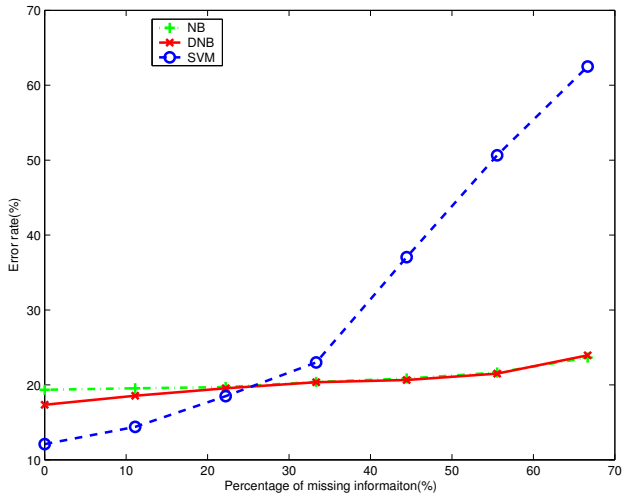

Figure 5: Error rates in Satimage with information missing.

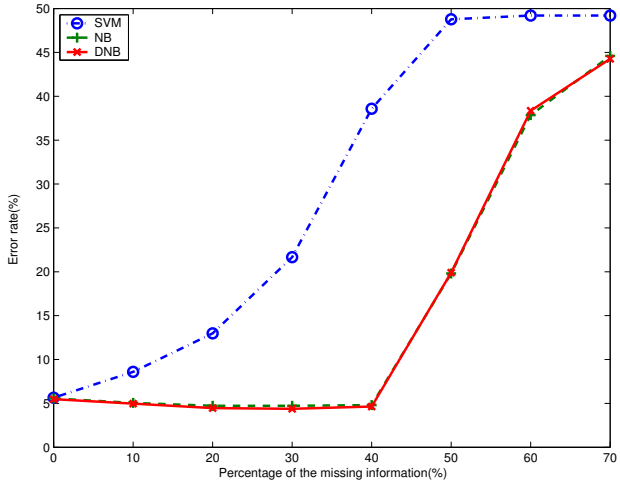

Figure 6: Error rates in DNA with information missing.

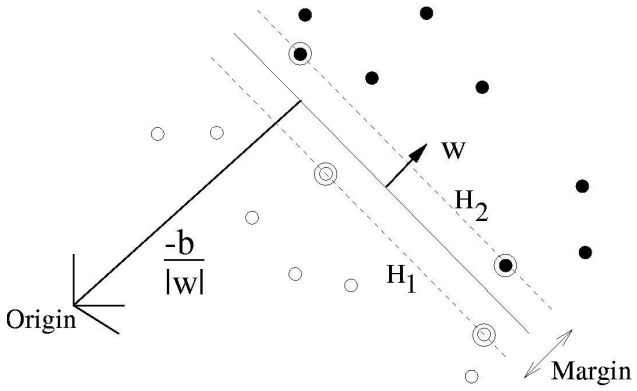

Figure 7: SVM

5 Discussion

5.1 Why DNB performs not as well as SVM when no information is missing?

In SVM, a linear classifier $y = w \cdot x + b$ with the maximum margin between two classes is searched by minimizing the following function as Eq. (20).

$$\tau(w, \xi) = \frac{C}{N} \sum_{i=1}^{N} \xi_i + \frac{1}{2} \parallel w \parallel^2 \tag{20}$$
$$s.t. \quad y_i \cdot ((w \cdot x_i) + b) \geq 1 - \xi_i, \quad \xi_i \geq 0$$

To handle the non-linear problem, usually the so-called kernel trick will be used to map the input into a high-dimension feature space, where a linear classifier can be found. This function consists of two parts. Since $\frac{2}{\|w\|}$ represents the margin between two classes, the second part on the right hand of Eq. (20), namely $\frac{1}{2} \parallel w \parallel^2$ describes the extent on how far two classes are from each other. The first term can be considered as the loss function in the training dataset, i.e., how accurate the sample in the training dateset can be classified into the corresponding class. Interestingly, we note this optimization function for SVM is similar as the one for the DNB. In the DNB model, two terms form the optimization function as Eq (8). The second term represents a similar meaning as the one in SVM. The first term in the DNB also tries to approximate the training dataset as accurately as possible. The difference is that in SVM, the first part directly minimizes the recognition error rate while in the DNB, this part minimizes an intermediate term not directly related to the classification error rate. As Box says, all models are wrong (but some are useful) [3]. Thus, using a generative model to approximate the dateset and then making predictions need always pay something. This may explain why SVM perform better than DNB when no information is missing.

5.2 Relation between DNB and Fisher Discriminant Classifier

It is also interesting that Fisher Discriminant also uses a similar idea as ours to separate two classes. Fisher Discriminant function is defined as:

$$J_F(w) = \frac{(\mu_1 - \mu_2)^2}{D_1^2 + D_2^2}, \tag{21}$$

$$where \quad \mu_i = \frac{1}{N}\sum_{x \in S_i} x, D_i^2 = \sum_{x \in S_i}(x - \mu_i)^2. \tag{22}$$

S_i are the sub-dateset for class i. By maximizing $J_F(w)$, Fisher Discriminant minimizes the inner-class divergence described by the denominator and maximizes the inter-class divergence describe by the numerator. The classifier is given as a linear form: $y = sign(w \cdot x + b)$. We argue that using the difference between the mean values as the divergence between two class is not an informative way as Kullback-Leibler divergence. This may partly explain why Fisher Discriminant is often used as the dimension reduction method rather than a classification method. Since DNB can also be considered as a linear classifier, to some extent, it may be regarded as an informative counterpart of the Fisher Discriminant.

5.3 Extension to Bayesian Network Classifier

As a special case of Bayesian Network classifiers (BN) [14], NB can be depicted as a dependence graph as Fig. 8. A more general BN does not assign the fixed dependence relationship as in NB. Instead, it tries to search the dependence relationship among the variables.

An interesting question is proposed naturally. Can the discriminative training for the NB be extended to the general Bayesian Network classifiers. However, difficulties will be encountered in trying to give a positive answer to the question. It has been reported that searching an general Bayesian network structure is an NP-hard problem [5]. Introducing an interactively discriminative term will make this problem more difficult to be solved. Thus a popular way is to develop restricted BN, in which, the dependence relationship can be found relatively easier.

Usually, restricted Bayesian networks confine the dependence structure in a specific fam-

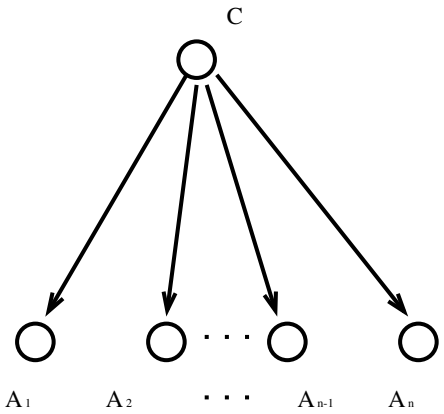

Figure 8: Naive Bayesian Network Classifier. $A_i, 1 \leq i \leq n$ is the attribute. This figure means each attribute is independent on the other attributes, given the class label C.

ily, for example, a tree family. Optimization on restricted BN often goes into two parts sequently, namely, structure learning and then parameter fitting when given the structures. However, even for the restricted BN, the number of possible restricted structures is normally huge and thus make the optimization on them intractable. Two exceptions are the NB and tree-like BN. In NB, the structure is fixed, since no other structures are possible when the independency assumption is given. In tree-like BN, The joint probability under the tree-like structure assumption can written as a decomposed form as

$$P(A_1, A_2, \ldots, A_n) = \prod_{j=1}^{n} \hat{P}(A_j | pa(A_j)), \tag{23}$$

where $pa(A_j)$ means the parent variable of variable A_j under the tree assumption. Then a Chow-Liu algorithm [4] can be used to optimize the structure and the parameter simultaneously in an $O(n^2)$ time. Here n is the number of the variables. Even for tree-like BNs, discriminative training encounter problems. Different from DNB, a structure learning problem is involved. The heuristic gradient method will be hard to be integrated to the optimization.

$$\{p^*_{B_1}, p^*_{B_2}, B_1, B_2\} = \arg\max_{\{p_{B_1}, p_{B_2}, B_1, B_2\}} \{KL(p_{B_1}, \hat{p_1}) + KL(p_{B_2}, \hat{p_2}) + W \cdot Div(p_{B_1}, p_{B_2})\}, \tag{24}$$

To be detailed, the optimization as Eq. (24) in tree-like BN is involved into the learning problem of structures B_1 and B_2. Without the discriminative item, B_i, P_{B_i} can be separa-

tively optimized by Chow-Liu algorithm. When added into the interactive term, Chow-Liu algorithm is not directly available to solve the problems. The heuristic gradient method in DNB cannot be given as a close-form function of the structure, thus it is also not applicable in such a task..

In the following we develop an iterative algorithm to train the tree-like BN in a discriminative way. We call this algorithm Discriminative Chow-Liu Tree algorithm (DCLT). Instead performing optimization on one objective function, we optimize two functions sequently in an iterative way:

$$f_1 = KL(p_{B_1}, \hat{p_1}) + W \cdot Div_1(p_{B_1}, p_{B_2}) \tag{25}$$

$$f_2 = KL(p_{B_2}, \hat{p_2}) + W \cdot Div_2(p_{B_1}, p_{B_2}) \tag{26}$$

$where,$

$$Div_1(p_{B_1}, p_{B_2}) = -\sum p_{B_1} \log \frac{p_{B_1}}{p_{B_2}} \tag{27}$$

$$Div_2(p_{B_1}, p_{B_2}) = -\sum p_{B_2} \log \frac{p_{B_2}}{p_{B_1}} \tag{28}$$

When B_1, p_{B_1} is fixed, f_2 can be minimized by a modified Chow-Liu tree algorithm. So does f_1 when B_2, p_{B_2} is fixed. The details can be seen in Appendix. The solvability implies us an iterative way as Fig 9 to perform discriminative training. The detailed algorithm is written as follows.

To guarantee the convergence of the above scheme, we can set W to $W_0 \cdot \exp^{-\beta \cdot i}$.

We implement the DCLT algorithm on two real world datasets. The recognition accuracy is shown in Table 4. From the table, DCLT improves the accuracy of CLT in both datasets. When compared with the DNB, DCLT wins in one dataset while loses in another dataset. Further evaluations on other datasets and the comparison between DCLT and SVM is ongoing.

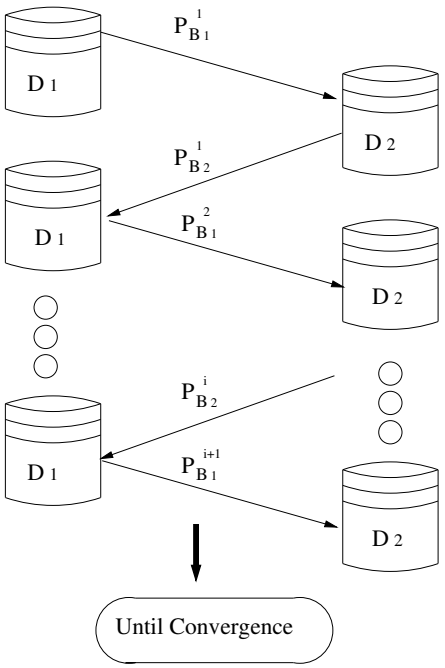

Figure 9: Iterative Training of Tree-like Bayesian Networks in a Discriminative way

Algorithm (D, W_0, β)

input(*pre-classified Dataset* $D = \{x^1, x^2, \ldots, x^N\}$,$W_0, \beta$)

Initialization: $\{B_1, p_{B_1}\} = \{B_1^0, p_{B_1}^0\}$, $i = 1$. Partition D into two sub-datasets D_1 and D_2 by class;

repeat

$\{B_1^i, p_{B_1}^i\} \leftarrow \{B_1^{i-1}, p_{B_1}^{i-1}\}$;
Find $B_2^i, p_{B_2}^i$ by minimizing f_2 in D_2;
$i \leftarrow i + 1$;
$\{B_2^i, p_{B_2}^i\} \leftarrow \{B_2^{i-1}, p_{B_2}^{i-1}\}$;
Find $B_1^i, p_{B_1}^i$ by minimizing f_1 in D_1;
$i \leftarrow i + 1$;
$W = W_0 e^{-\beta i}$;

until *convergence*;

output($B_1, B_2, p_{B_1}, p_{B_2}$)

Algorithm 1: Iterative Bayesian Multinet Optimization Algorithm

Table 4: Recognition rate

Database	NB	DNB	CLT	DCLT
Hepatitis(%)	84.95	86.02	86.03	89.25
Vote(%)	90.34	93.33	91.26	92.18

6 Conclusion

In this paper, we proposed a novel model named "Discriminative Naive Bayesian classifiers". This model combines the advantages of generative classifiers with discriminative classifiers. When handling the tasks without information missing, the DNB demonstrates a superior performance than the Naive Bayesian classifer. When handling the tasks with information missing, the DNB outperforms the Support Vector Machines. We discuss the relationship between our model and other models such as Support Vector Machines and Fisher Discriminant. We also present an extension of our model to general Bayesian Network classifiers.

References

[1] L. R. Bahl, P. F. Brown, P. V. de Souza, and R. L. Mercer. Estimating hidden markov model parameters so as to maximize speech recognition accuracy. IEEE *Transactions on Speech and Audio Processing*, 1:77–82, 1993.

[2] F. Beaufays, M. Wintraub, and Y. Konig. Discriminative mixture weight estimation for large gaussian mixture models. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pages 337–340, 1999.

[3] G. Box and G. Tiao. *Bayesian Inference in Statistical Analysis*. John Wiley & Sons,, 1992.

[4] C. K. Chow and C. N. Liu. Approximating discrete probability distributions with dependence trees. *IEEE Trans. on Information Theory*, 14:462–467, 1968.

[5] J. Dougherty, R. Kohavi, and M. Sahami. Supervised and unsupervised discretization of continuous features. In *International Conference on Machine Learning*, pages 194–202, 1995.

[6] R. Duda and P. Hart. In *Pattern classification and scene analysis*. John Wiley & Sons, 1973.

[7] N. Friedman, D. Geiger, and M. Goldszmidt. Bayesian network classifiers. *Machine Learning*, 29:131–161, 1997.

[8] T. Hastie and R. Tibshirani. Discriminant analysis by gaussian mixtures. *Journal of the Royal Statistical Society(B)*, 58:155–176, 1996.

[9] R. Kohavi. A study of cross validation and bootstrap for accuracy estimation and model selection. In *Proceedings of the 14th IJCAI*, pages 338–345. San Francisco, CA:Morgan Kaufmann, 1995.

[10] P. Langley, W. Iba, and K. Thompson. An analysis of bayesian classifiers. *In Proceedings of AAAI-92*, pages 223–228, 1992.

[11] L. S. Lasdon. *Optimization theory for large systems*. The Macmillam Company, 1970.

[12] G. J. McLachlan and K. E. Basford. *Mixture Models: Inference and Applications to Clustering*. Marcel Dekker Inc., New York, 1988.

[13] P. M. Murphy. UCI repository of machine learning databases. In *ftp.ics.uci.edu: pub/machine-learning-databases*.

[14] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: networks of plausible inference*. Morgan Kaufmann, CA, 1988.

[15] L. R. Rabiner and B. H. Juang. An introduction to hidden Markov models. *IEEE ASSP Magazine*, pages 4–15, January 1986.

[16] J. Rosen. The gradient projection method for nonlinear programming:part i-linear constraints. *SIAM J. Appl. Math.*, (8):181–217, 1960.

[17] J. T. S. and D. Haussler. Exploiting generative models in discriminative classifiers. In *Advances in Neural Information Processing Systems*, 1998.

[18] V. Valtchev, J. J. Odell, P. C. Woodland, and S. J. Young. Lattice-based discriminative training for large vocabulary speech recognition. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pages 605–608, 1996.

[19] V. N. Vapnik. *The nature of statistical learning theory.* Springer, New York, 2nd edition, 2000.

Appendix

Let $l = \sum_{\{a_1,\dots,a_n\}}(\hat{p_2} - W \cdot p_{B_1}) \log p_{B_2}$, when B_1, p_1 is fixed, the minimization on f_2 is equivalent to the minimization of l. Under a tree dependence structure t among the variables, l can be decomposed as:

$$\sum_{i=1}^{n} \sum_{\{a_i, a_{pa(i)}\}} [\hat{p_2}(a_i, a_{pa(i)}) - W \cdot p_{B_1}(a_i, a_{pa(i)})] \log(p_{B_2}(a_i|a_{pa(i)}) \tag{29}$$

Let:

$$p^d(a_i, a_{pa(i)}) = \hat{p_2}(a_i, a_{pa(i)}) - W \cdot p_{B_1}(a_i, a_{pa(i)}) \tag{30}$$

Thus, the optimization is changed into:

$$\max_t l_t = \max_t \{\sum_{i=1}^{n} \max_{P|t} \sum_{\{a_i, a_{pa(i)}\}} [p^d(a_i, a_{pa(i)}) \log p_{B_2}(a_i|a_{pa(i)})]\}$$

When given a specific tree t, the inner maximization can be obtained by applying Kuhn-Tucker conditions and Lagrange multiplier method under the constraint:$p_{B_2}(a_i, a_{pa(i)}) \geq \xi_2$ and $\sum_{\{a_i, a_{pa(i)}\}} p_{B_2}(a_i, a_{pa(i)}) = 1$. ξ_2 is a positive constant close to zero[1]

$$P(A_1, A_2, \dots, A_n) = \prod_{j=1}^{n} \hat{P}(A_j|pa(A_j)), \tag{31}$$

where $pa(A_j)$ means the parent vertex of vertex A_j. Each subitem $\hat{P}(A_j|pa(A_j))$ can be reliably estimated based on the empirical distribution.. The solution for the associated probabilities p_{B_2} can be written as:

$$p^t_{B_2}(a_i, a_j) = \xi_2, \quad if \quad p^d(a_i, a_j) \leq 0, i \neq j; \tag{32}$$

[1]None of the decomposed probabilities can be zero. Otherwise, the restored joint probability as Eq. (31)will be always zero whatever the other decomposed probabilities are.

$$p^t_{B_2}(a_i, a_j) = p^d(a_i, a_j)/Z,$$
$$if \quad p^d(a_i, a_j) > 0, i \neq j; \tag{33}$$

where Z is a normalization factor, which is used to guarantee $\sum_{\{a_i,a_j\}} p^t_{B_2}(a_i, a_j) = 1$;

Therefore, Eq. (31) continues to be:

$$\begin{aligned}
&\max_t l_t = \\
= \ & \{\sum_{i=1}^{n} \sum_{\{a_i, a_{pa(i)}\}} p^d(a_i, a_{pa(i)}) \log p^t_{B_2}(a_i|a_{pa(i)})\} \\
= \ & \max_t \{[\sum_{i=1}^{n} \sum_{\{a_i, a_{pa(i)}\}} [p^d(a_i, a_{pa(i)}) \\
& \log(\frac{p^t_{B_2}(a_i, a_{pa(i)})}{p^t_{B_2}(a_i) p^t_{B_2}(a_{pa(i)})})]] - \sum_{i=1}^{n} H(A_i)\}
\end{aligned} \tag{34}$$

Where, $H(A_i) = -\sum_{i=1}^{n} p^d_{B_2}(a_i) \log p^t_{B_2}(a_i)$, $1 \leq i \leq n$. $-\sum_{i=1}^{n} H(A_i)$ is a constant for any tree structure. Thus we can remove this item from Eq. (34). Further, we define the *discriminative mutual information* for a pair of variable a_i, a_j as:

$$I^d(A_i, A_j) = \sum_{\{a_i,a_j\}} p^d(a_i, a_j) \log \frac{p^t_{B_2}(a_i, a_j)}{p^t_{B_2}(a_i) p^t_{B_2}(a_j)} \tag{35}$$

Eq. (34) continues to be:

$$\begin{aligned}
&\max_t l_t \\
= \ & \max_t \sum_{i=1}^{n} I^d(A_i, A_{pa(i)})
\end{aligned} \tag{36}$$

This is a Maximum Weight Spanning Tree problem (MWST), where the weights are given by the discriminative mutual information. The MWST problem can be solved by many mature methods.