Improving Naive Bayesian Classifier by Discriminative Training

Kaizhu Huang
Information Technology Lab
Fujitsu R&D Center CO.,LTD
Chaoyang District, Beijing, P.R.China
Email: kzhuang@frdc.fujitsu.com

Zhangbing Zhou
Bell Labs China
Lucent Technologies
Hai Dian Nan Lu, Beijing, P.R.China
Email: zzb@lucent.com

Irwin King, Michael R. Lyu
Dept. of Computer Science and Engineering
The Chinese Univ. of Hong Kong
Shatin, N.T. Hong Kong
Email:{king, lyu}@cse.cuhk.edu.hk

***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 the Bayes rule to construct a posterior classifier. In general, generative classifiers are not as accurate as discriminative classifiers. However generative classifiers provide a principled way to deal with the missing information problem, which discriminative classifiers cannot easily handle. To achieve good performance 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 the Discriminative Naive Bayesian classifier. We provide theoretic justifications, outline the algorithm, and perform a serious of experiments on benchmark real-world datasets to demonstrate our model's advantages. Its performance outperforms NB in classification tasks and outperforms SVM in handling missing information tasks.**

I. Introduction

Generative classifiers have shown their advantages in many classification tasks, even though their overall performance is not as good as discriminative classifiers such as Support Vector Machines [18][9]. A typical example of generative classifiers is the Naive Bayesian classifier [5][12]. 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 point x into the class with the highest posterior probability as shown in Eq. (1), where A_i, A_j represent the attributes or variables, C denotes the class variable, n is the number of the attributes. In Eq. (2), this posterior classification rule can be transformed into a joint probability classification rule, since $P(A_1, A_2, \ldots, A_n)$ for a given data point is a constant with respect to C. Finally, by combining the independency assumption, the classification rule is changed in 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 applications, NB first partitions the dataset into several sub-datesets by the class label. Then, in each sub-dataset labelled by C_i, $P(A_j = a_{jk}|C_i)$ can be 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 according to the Maximum Log-likelihood (ML) criterion. Here n_i is the number of the samples in sub-dateset C_i.

This above simple scheme achieves surprising success in many classification tasks [5][12][6]. Importantly, a great advantage of NB is its immediate ability to deal with the missing information problem. 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 or missing, 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 calculated in training NB.

However, there are still shortcomings in NB. More specifically, this approach models the joint probability in each subset separately and then applies the Bayes rule to construct the posterior classification rule. This framework appears to be incomplete, since this construction procedure actually discards important discriminative information for classification. Without considering the other classes of data, 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 the 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 solutions is to directly learn a posterior probability model rather than a joint probability model. However within the framework of Bayesian network classifier, this kind of approaches is often computationally hard to perform the optimization. Even for the simple Naive Bayesian classifier, the corresponding posterior learning, known as the Logistic Regression (LR) [11], will encounter problems in order to deal 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, the ML criterion can be used to find the parameters β and θ. When the values of a subset of attribute set T are unknown, the marginalization on T is obtained 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. First, $P(A-T, T)$ varies from T, thus it cannot be omitted. Second, $P(c = C_0|A)$, 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. We call this model the Discriminative Naive Bayesian (DNB) classifier. Beginning with modelling the joint probabilities for the data, we plug into the optimization function a penalty item which describes the divergence between two classes. On one hand, the optimization of the new function 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 inherits the NB's ability in handling the missing information problem.

This paper is organized as follows. In the next section, we present a short review on related work. In Section III, we describe the Discriminative Naive Bayesian classifier in detail. We then in Section IV evaluate our algorithm on four benchmark datasets. In Section V, the relationship between our algorithm and other approaches, such as SVM, and Fisher Discriminant Analysis, is discussed. Finally, in Section VI, we set out the conclusion.

II. Related work

Combining generative classifiers and discriminative classifiers has been one of active topics in machine learning. A lot of work [1][7][17][2] has been done in this area. However nearly all of these methods are designed for Gaussian Mixture Model [14] or Hidden Markov Model [15]. By contrast, our discriminative approach is developed for one of Bayesian network classifiers, the Naive Bayesian classifier. On the other hand, Jaakkola et al. develop a method to explore generative models from discriminative classifiers [10]. Different from this approach, our method performs a reverse way to use discriminative information in generative classifiers. In [8], a discriminative training is performed on a kind of tree belief networks, Chow-Liu Tree; however, it appears hard to prove the convergence of the algorithm.

III. Discriminative Naive Bayesian Classifier

In this section, we first develop the discriminative Naive Bayesian classifier in a two-category classification task. Then in Section III-B we emploit a voting scheme to extend our method into multi-category classification tasks.

A. Two-category Discriminative Naive Bayesian Classifier

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 represent the data with the class label C_1 and C_2 respectively. Then in each sub-dataset, the ML or the cross entropy criterion can 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 the Kullback–Leibler function shown in the following:

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

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

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

$\hat{p_1}$ and $\hat{p_2}$ represent the empirical distribution for the 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

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

$Div(p_1, p_2)$ 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

$$Div(p_1, p_2) = \frac{1}{\sum_x p_1 \log \frac{p_1}{p_2}} \ . \tag{8}$$

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 plugging 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 in a complete form:

$$\begin{aligned} &\min_{\{p_1,p_2\}} \sum_{c=1}^{2} \sum_{j=1}^{n} \sum_{A_j} [\hat{p_c}(a_{jk}) \log \frac{p_c(\hat{a}_{jk})}{p_c(a_{jk})}] \\ &+W \cdot \frac{1}{\sum_{j=1}^{n} \sum_{A_j} p_1(a_{jk}) \log(p_1(a_{jk})/p_2(a_{jk}))}, \quad (9) \\ &s.t. \quad 0 \le p_c(a_{jk}) \le 1, \quad (10) \\ &\sum_{A_j} p_c(a_{jk}) = 1, \quad c = 1, 2; j = 1, 2, \ldots n. \end{aligned}$$

$p_c(a_{jk})$ 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 gradients of the optimization function with respect to p_1 and p_2. 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 the Quadratic Interpolation method [13]. The process is repeated until a local minimal is obtained. We write down the detailed steps as follows:

1: Calculate the gradient according to Eq. (13-15).
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 p_1, p_2 by the following Equations.

$$p_1(a_{jk})^{new} = p_1(a_{jk})^{old} - \alpha \nabla f^M_{1jk} \ ; \tag{11}$$
$$p_2(a_{jk})^{new} = p_1(a_{jk})^{old} - \alpha \nabla f^M_{2jk} \ . \tag{12}$$

5: Goto step 1 until p_1 and p_2 converge.

$$\begin{aligned} \frac{\partial f}{\partial p_1(a_{jk})} &= -\hat{p_1}(a_{jk})/p_1(a_{jk}) \\ &- \frac{W}{Z}[1 + \log(p_1(a_{jk})/p_2(a_{jk}))]; \quad (13) \\ \frac{\partial f}{\partial p_2(a_{jk})} &= -\hat{p_2}(a_{jk})/p_2(a_{jk}) \\ &+ \frac{W}{Z} p_1(a_{jk})/p_2(a_{jk}); \quad (14) \\ Z &= \sum_{i=1}^{n} \sum_{A_j} \log \frac{p_1(a_{jk})}{p_2(a_{jk})}; \quad (15) \end{aligned}$$

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

where A is the coefficient matrix for the constraint.

B. Multi-category Discriminative Naive Classifier

We use a partly-connected 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 vote 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. 1, we illustrate a four-category committing machine. In Fig. 1, 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.

IV. Evaluations

In this section, we implement the DNB algorithm to evaluate its performance on four benchmark datasets from Machine Learning Repository in UCI [3]. The detailed information for these datasets are listed in Table I. We intentionally choose these datesets which vary in the variable number and the sample size. As observed in Table I, the variable number ranges from 4 to 60 and the sample size varies from 150 to 6435. The diversity in choosing the datesets will make the evaluations on the algorithms more reliable. For the Iris dateset, which has a small number of samples, we use a five-fold Cross-Validation method (CV5) to test the performance. We compare our model's performance with NB and a 3-order polynomial kernel 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 II.

A. Without Information Missing

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

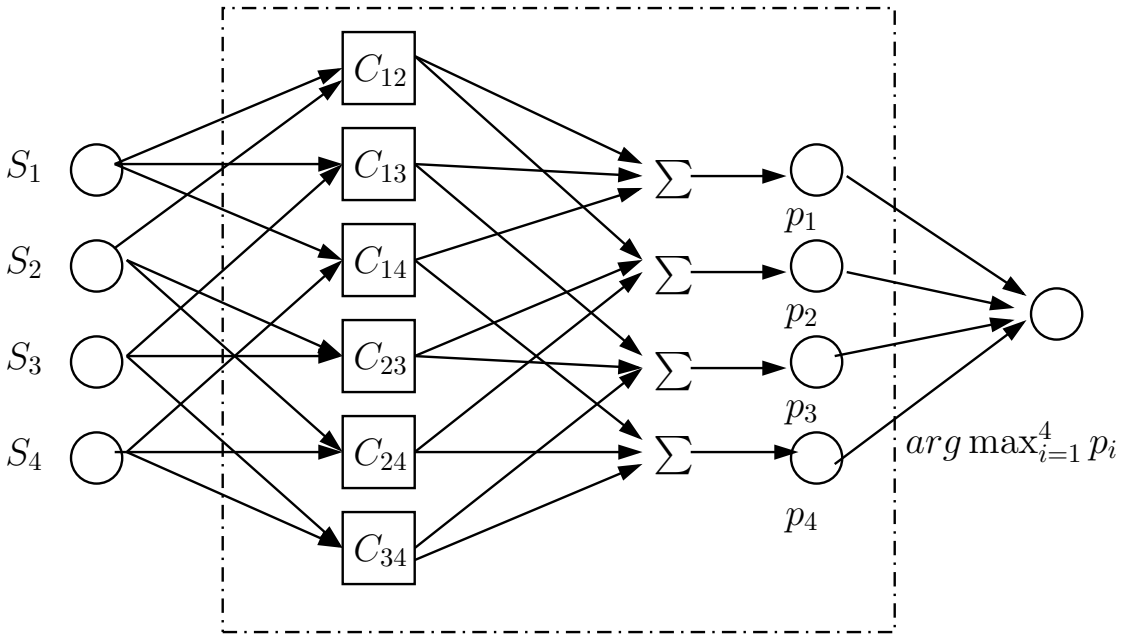

Fig. 1. Discriminative Naive Bayesian Classifier Committing Machine for a four-category problem. S_i, $1 \leq i \leq 4$ represent the sub-dataset for category i respectively. $C_l m$, $1 \leq l \leq 3$, $l < m$ means the two-category Discriminative Naive Beyesian classifier for category l and category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

TABLE I

DESCRIPTION OF DATA SETS USED IN THE EXPERIMENTS

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

TABLE II

PARAMETERS USED IN THE EXPERIMENTS

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

information in training the generative models benefits the classification greatly. When compared with SVM, DNB wins in two of the datesets while it loses in the other two. We note that in these two datasets, i.e., Segment and Setimage, SVM performs significantly better than DNB. This demerit of DNB roots in the inner scheme of generative classifiers. In Section V-A, we will present a detailed discussion on this issue.

TABLE III

PREDICTION ACCURACY WITHOUT INFORMATION MISSING(%)

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

B. With Information Missing

It is important to discuss the ability of the DNB in handling the missing information problem, since one of the main advantages for generative classifiers lies at this point. Gradually, we increase the percentages of the number of unknown or missing attributes randomly. We then test the recognition rate on these datasets with different percentages. As mentioned previously for DNB and NB, a principled way to handle the missing information problem is to use inference under uncertainty: $c = \arg\max_{C_i} P(C_i) \prod_{A_j \in A-T} P(A_j|C_i)$. For SVM, a normal way to force its application in missing information tasks is simply setting zero values for the missing attributes. Since in implementing SVM, the values of attributes are pre-normalized to the range [-1,1], the zero value can be considered as the average value of the attributes. Thus in a sense, this method can be regarded as replacing the missing values with the corresponding average value. The experiment results for the four datasets are shown in Fig. 2.

It is shown that NB demonstrates a robust ability to handle the missing information problem. In four datasets, the error rate curves of NB maintain a flat trend when the information does not decay too much. Furthermore, DNB shows a similar resistance ability while its accuracy is higher than NB. In comparison, SVM's performance gradually runs down as the number of the missing variables goes up. The superiority of DNB over NB and SVM is especially prominent in the Iris dataset. In the Iris dataset the number of samples is relatively small. Thus the the distribution from insufficient training data may not represent the real distribution. Therefore, the discriminative item will contribute more in constructing the classifiers.

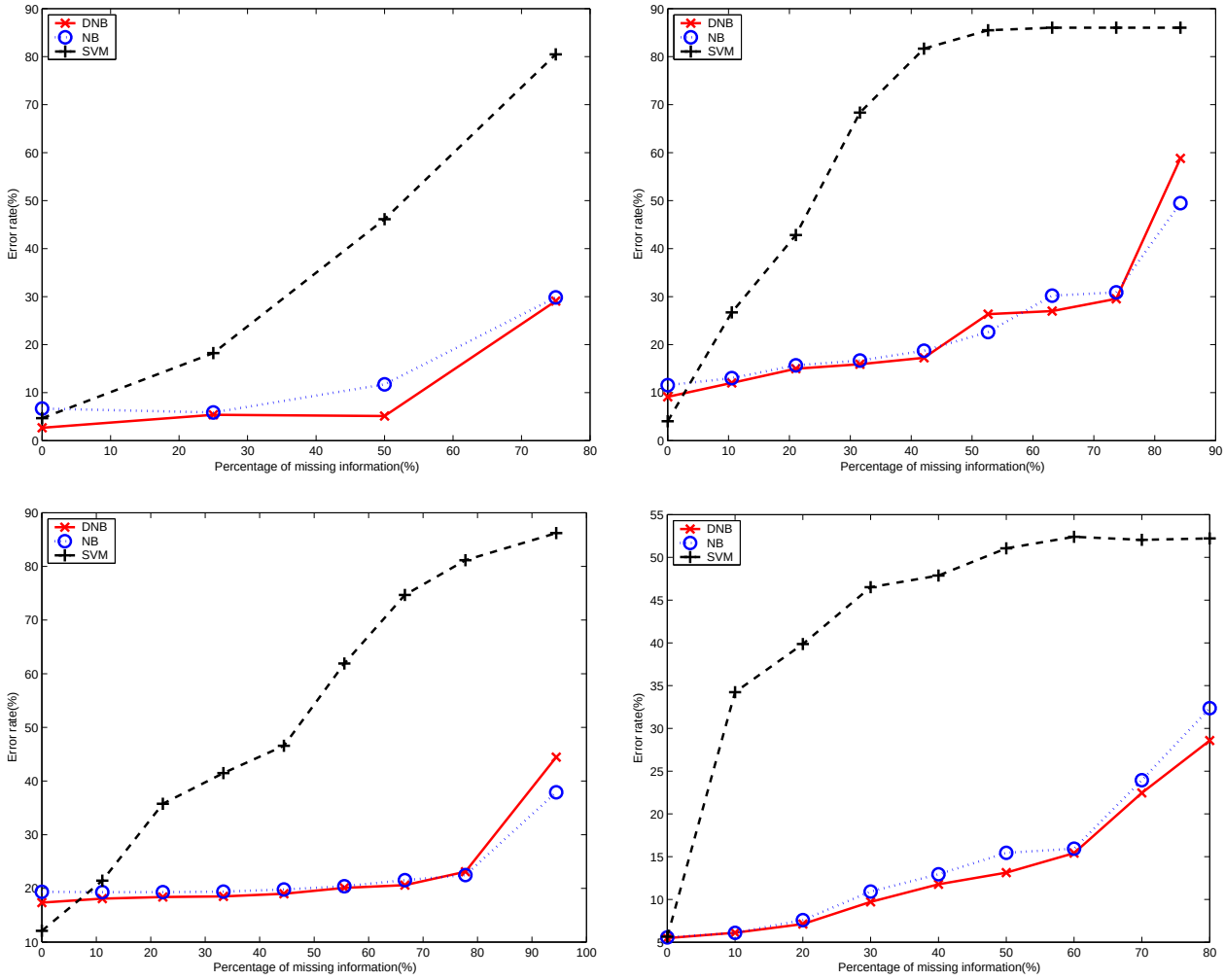

Fig. 2. Error rates with information missing for four datasets. The figures on the up-left, up-right, bottom-left,bottom-right are the curves for Iris, Segment, Satimage and DNA respectively.

V. Discussions

A. 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. (17).

$$\tau(w, \xi) = \frac{C}{N} \sum_{i=1}^{N} \xi_i + \frac{1}{2} \parallel w \parallel^2$$
$$s.t. \quad y_i \cdot ((w \cdot x_i) + b) \geq 1 - \xi_i, \quad \xi_i \geq 0, \quad (17)$$

where, $x_i \in \mathbb{R}^n$, n is the data dimension, $y_i \in \mathbb{R}$ is the class label corresponding to x_i and $w \in \mathbb{R}^n$, $b \in \mathbb{R}$ are the variables to be optimized. 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. (17), namely $\frac{1}{2} \parallel w \parallel^2$ describes the extent on how faraway 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 of SVM is similar to the one of DNB. In the DNB model, two terms form the optimization function as Eq. (8). The second term represents a meaning similar to the one in SVM. The first term in 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 DNB, this part minimizes an intermediate term representing the difference between the estimated joint distribution and the empirical distribution. As Box says, all models are wrong (but some are useful) [4]. The estimated distribution under the strong independency assumption may not always coincide with the real data, and therefore may fail to work in practice.

B. Relation Between DNB And Fisher Discriminant Analysis

It is interesting that Fisher Discriminant Analysis (FDA) also uses an idea similar to ours to separate two classes.

The discriminant function in FDA is defined as: $J_F(w) = \frac{(\mu_1-\mu_2)^2}{D_1^2+D_2^2}$, where $\mu_i = \frac{1}{N}\sum_{x\in S_i} x, D_i^2 = \sum_{x\in S_i}(x-\mu_i)^2$ and S_i is the sub-dateset for class i, N is the number of the data points. By maximizing $J_F(w)$, FDA minimizes the inner-class divergence described by the denominator and maximizes the inter-class divergence represented by the difference of the means between two classes. The final classification rule is provided in a linear form: $y = sign(w \cdot x + b)$. However, using the difference between the mean values as the divergence between two classes may not be an informative way as the Kullback-Leibler divergence, a distribution-based approach.

VI. Conclusion

In this paper, we have proposed a novel model named the Discriminative Naive Bayesian classifier. 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 Machine. A series of experiments has been conducted to evaluate our model. The results have demonstrated the effectiveness of our model in comparison with the Naive Bayesian classifier and the Support Vector Machine.

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] C.L. Blake and C.J. Merz. UCI repository of machine learning databases. University of California, Irvine, Dept. of Information and Computer Sciences, 1998. URL: http://www.ics.uci.edu/∼mlearn/MLRepository.html.

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

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

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

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

[8] K. Huang, I. King, and M. R. Lyu. Discriminative training of bayesian chow-liu tree multinet classifiers. In *Proceedings of International Joint Conference on Neural Network (IJCNN-2003), Oregon, Portland, U.S.A.*, volume 1, pages 484–488, 2003.

[9] K. Huang, H. Yang, I. King, and M. R. Lyu. Learning large margin classifiers locally and globally. In Russ Greiner and Dale Schuurmans, editors, *The twenty-first International Conference on Machine Learning (ICML-2004)*, pages 401–408, 2004.

[10] T. S. Jaakkola and D. Haussler. Exploiting generative models in discriminative classifiers. In *Advances in Neural Information Processing Systems (NIPS 11)*, 1998.

[11] M. I. Jordan. Why the logistic function? A tutorial discussion on probabilities and neural networks. Technical Report 9503, MIT Computational Cognitive Science Report, 1995.

[12] P. Langley, W. Iba, and K. Thompson. An analysis of bayesian classifiers. *In Proceedings of National Conference on Artificial Intelligence*, pages 223–228, 1992.

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

[14] G. J. McLachlan and K. E. Basford. *Mixture Models: Inference and Applications to Clustering*. Marcel Dekker Inc., New York, 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. B. Rosen. The gradient projection method for nonlinear programming:part I-linear constraints. *SIAM J. Appl. Math.*, (8):181–217, 1960.

[17] 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.

[18] V. N. Vapnik. *The Nature of Statistical Learning Theory*. Springer, New York, 2nd edition, 1999.