DECISION BY MAXIMUM OF POSTERIOR PROBABILITY AVERAGE WITH WEIGHTS: A METHOD OF MULTIPLE CLASSIFIERS COMBINATION

PENG-TAO JIA, HUA-CAN HE, WEI LIN

School of Computer Science, Northwest Polytechnical University, Postbox 757, 710072 Xi'an, P.R.China
E-MAIL: jiapengtao@xust.edu.cn,hehuac@nwpu.edu.cn,linwei2002@sina.com

Abstract:

In this paper, a new multiple classifiers combining algorithms, that is Maximum of posterior probability Average with Weight (MAW rule), is introduced. We adopt the same methods with Bagging to train single classifier in this algorithm, but we amend integration rule, which the result lies on maximum in average with weight for every class rather than majority vote. This algorithm bases on parallel integration, and naïve Bayesian classification is used to construct single classifier. Besides our combining algorithm, we also select other algorithms, which are Max rule, Min rule, Majority vote rule, Sum rule, and Product rule as comparing objects. According to experiments on KDD99 dataset and the letter dataset of UCI, MAW rule lead to less error than other combining algorithms and better performance.

Keywords:

Multiple classifiers combination; Integration rule; Naïve Bayesian classification; MAW rule

1. Introduction

Traditional pattern classification usually bases on single classifier. Therefore the classifier must have divisional ability on all features of examples. Although many scholars perform some efforts on single classifier using in pattern classification for improving recognition capability, these investigations mainly focus on exploring new classification algorithm and improving old classification algorithm, and do not overcome disadvantage of single classification[1]. One of the important directions in improvement of the pattern classification is the integration of multiple classification techniques. An integration technique should estimate and then select the most appropriate component classifiers from an ensemble of classifiers. By using integration of multiple classifiers, we can reduce recognition error rate and improve robust of classification[2]. Thus the research of integration of multiple classifiers becomes a hotspot. At present, recognition based on integration of multiple classifiers was applied in many fields, such as handwritten and text recognition[3], face recognition[4], time-series prediction[5], etc.

Using the system of multiple classifiers can improve classification capability. The strong emphasis on research of multiple classifiers is how to increase the performance of classification system by making use of complementarity of classification algorithm. We have reached great academic achievement on aspect of integration of multiple classifiers, and put forward a lot of integration methods of multiple classifiers, such as boosting and bagging[6]. But these integration methods always have not good capability in different datasets. The challenge of the integration problem is to decide which classifiers to rely on or how to combine results of several classifiers.

In this paper, we introduce an integration classifier system, which will be introduced by next section. We use different datasets to train and test our model. Then the results of classification are given. After comparing our algorithm with other algorithms, the result shows the performance of classification is improved. Finally we conclude the performance of our system.

2. Integration of Multiple Classifiers

2.1. Basic Model of Multiple Classifiers Combination

A system of multiple classifiers is made up of four components[7]: input system, design of single classifier, a system of multiple classifiers structure, and integration rules. Input system is indication manner of input and single classifier; design of single classifier is study algorithm of every classifier and the definition of correlative parameters; a system of multiple classifiers structure has two manners: cluster structure and parallel structure; integration rules are combination manner of classification judgment. If we want to construct a multiple classifiers system, we must confirm these components firstly. Because input system is a certain problem, we omit it in this paper, and we pay attention to the design of other components.

2.2. Algorithm of Naïve Bayesian Classification

An important issue in combining classifiers is that this is particularly useful if they are different[1]. This can be achieved by using different feature sets, as well as by different training sets, randomly selected, or based on different classifiers. We choose the same model, naïve Bayesian classifier to construct sub classifiers and different training sets to train them to gain our ends.

In order to design single classifier of multiple classifiers, we select naïve Bayesian classification to construct every component classifier. Although the assumption that the predictor (independent) variables are independent is not always accurate, it does simplify the classification task dramatically, since it allows the class conditional densities $p(x_k \mid C_j)$ to be calculated separately for each variable, i.e., it reduces a multidimensional task to a number of one-dimensional ones. In effect, naïve Bayesian classification reduces a high-dimensional density estimation task to one-dimensional kernel density estimation. Furthermore, the assumption does not seem to greatly affect the posterior probabilities, especially in regions near decision boundaries, thus, leaving the classification task unaffected. Algorithm of Naïve Bayesian Classification describes as follows[8]:

1) Given a training dataset $D_1 = (X_1, X_2, ..., X_l)^T$, and $X_i = (x_{i1}, x_{i2}, ..., x_{in}, c)^T, i = 1,2,...,l$ depicting n measurements made on the sample from n attributes and 1 class label, respectively, $A_1, A_2, ..., A_n, C$.

2) Suppose there are class label set $C = \{C_1, C_2, ..., C_m\}$, and $c \in C$. Given an unknown data sample, X (i.e., having no class label), the classifier will predict that X belongs to the class having the highest posterior probability, conditioned on X. That is, the naive Bayesian classifier assigns an unknown sample X to the class C_i if and only if :

$$P(C_i \mid X) > P(C_j \mid X) \; for \; 1 \le j \le m, j \ne i \qquad (1)$$

Thus we maximize $p(C_i \mid X)$. The class C_i for which $p(C_i \mid X)$ is maximized is called the maximum posteriori hypothesis. By Bayes theorem,

$$P(C_i \mid X) = \frac{P(X \mid C_i)P(C_i)}{P(X)} \qquad (2)$$

3) As $p(X)$ is constant for all classes, only $P(X \mid C_i)p(C_i)$ need be maximized. If the class prior probabilities are not known, then it is commonly assumed that the classes are equally likely, i.e. $P(C_1) = P(C_2) =,...,= P(C_m)$, and we would therefore maximize $p(X \mid C_i)$. Otherwise, we maximize $P(X \mid C_i)p(C_i)$. Note that the class prior probabilities may be estimated by $P(C_i) = \frac{s_i}{s}, i = 1,2,...,m$, where S_i is the number of training samples of class C_i, and S is the total number of training samples.

4) Given datasets with many attributes, it would be extremely computationally expensive to compute $p(X \mid C_i)$. In order to reduce computation in evaluating $p(X \mid C_i)$, the naive assumption of class conditional independence is made. This presumes that the values of the attributes are conditionally independent of one another, given the class label of the sample, i.e., that there are no dependence relationships among the attributes. Thus,

$$P(X|C_i) = \prod_{k=1}^{n} P(x_k|C_i) \qquad (3)$$

The probabilities $P(x_1 \mid C_i), P(x_2 \mid C_i), \cdots, P(x_n \mid C_i)$ can be estimated from the training samples, where:

(a) If A_p is categorical (discrete-valued), then

$$P(x_k|C_i) = \frac{s_{jk}}{s_i} \qquad (4)$$

where s_{jk} is the number of training samples of class C_i having the value x_k for attribute A_p, and s_i is the number of training samples belonging to C_i.

(b) If A_p is continuous-valued, then the attribute is typically assumed to have a Gaussian distribution so that

$$P(x_k|C_i) = g(x_k, \mu_{C_i}, \sigma_{C_i}) = \frac{1}{\sqrt{2\pi}\sigma_{C_i}} e^{-\frac{(x_k-\mu_{C_i})^2}{2\sigma_{C_i}^2}} \quad (5)$$

where $g(x_k, \mu_{C_i}, \sigma_{C_i})$ is Gaussian (normal) density function for attribute A_p, while μ_{C_i} and σ_{C_i} are the mean and standard deviation, respectively, given the values for attribute A_p for C_i.

5) In order to classify an unknown sample X, $P(X \mid C_i)p(C_i)$ is evaluated for each class C_i. Sample X is then assigned to the class C_i if and only if :

$$p(X \mid C_i)P(C_i) > P(X \mid C_j)P(C_j)$$
$$for \quad 1 \le j \le m, j \ne i \quad (6)$$

where

$$P(X|C_i) = \prod_{p=1}^{n} P(x_k|C_i) \quad (7)$$

2.3. Multiple Classifiers Structure

In this paper, we introduce a model based on parallel integration rule. Figure 1 shows this model. We divide the train dataset into k subsets with same scale, and use every subset to train every component classifier. Then we will get k classifiers, and given different weight for every component classifier according by error rate. We obtain a decision rule as follows. In order to classify an unknown sample X, we firstly calculate $P(X \mid C_i)p(C_i)$ is evaluated for each class C_i, then

$assign \quad X \rightarrow C_i \quad if$

$$\max_{i=1}^{m} P(C_i \mid X) = \max_{i=1}^{m} \frac{\sum_{j=1}^{k} P_j(X \mid C_i)P_j(C_i) * \varpi_j}{k}$$
$$for \ i = 1,2,...,m; j = 1,2,\ldots,k; \quad (8)$$

where

$$\varpi_j = \frac{1 - f_j}{\sum_{i=1}^{k}(1 - f_i)} \quad (9)$$

f_j is error rate in single classifier. And

$$\sum_{i=1}^{k} \varpi_j = 1 \quad (10)$$

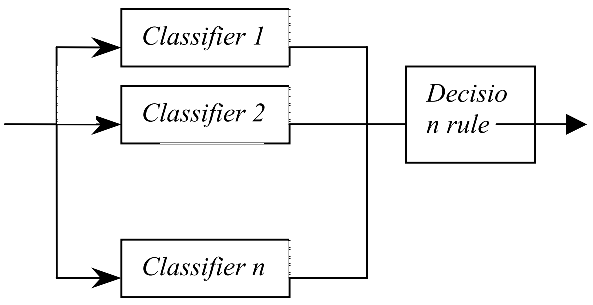

Figure 1. The structure of our combining classifiers

This method is similar to bagging algorithm, but the difference is the combining rules, which we know bagging algorithm adopt majority vote rule, and our combining rule is called Maximum of posterior probability Average with Weight (MAW Rule).

2.4. Another Combining Rules

In this section，other combining rules: Max rule, Min rule, Majority vote rule, Sum rule, and Product rule are showed as follows.

2.4.1. Max Rule

Starting from (7) and approximating the sum by the maximum of the posterior probabilities, we obtain

$assign \quad X \rightarrow C_i \quad if$

$$\max_{i=1}^{m} P(C_i \mid X) = \max_{j=1}^{k} \ \max_{i=1}^{m} P_j(C_i \mid X)$$
$$for \ i = 1,2,...,m; j = 1,2,\ldots,k; \quad (11)$$

2.4.2. Min Rules

Starting from (7) and bounding the product of posterior probabilities from above we obtain

$assign \quad X \rightarrow C_i \quad if$

$$\max_{i=1}^{m} P(C_i \mid X) = \max_{j=1}^{k} \min_{i=1}^{m} P_j(C_i \mid X)$$
$$for \;\; i = 1,2,...,m; j = 1,2,\ldots,k; \tag{12}$$

2.4.3. Majority Vote Rule

We use a posterior probabilities $P(C_i \mid X)$ to produce binary valued functions Δ_{ij} as

$$\Delta_{ij} = \begin{cases} 1 & if \; P(C_i \mid X) = \max_{j=1}^{m} P(C_j \mid X) \\ 0 & otherwise \end{cases} \tag{13}$$

Starting from (7) under the assumption of equal priors and by hardening the probabilities according to (13), we obtain

$assign \;\; X \to C_i \;\; if$

$$\sum_{j=1}^{k} \Delta_{ij} = \max_{k=1}^{m} \sum_{i=1}^{k} \Delta_{ki} \tag{14}$$

Note that for each class C_i the sum on (14) simply counts the votes received for this hypothesis from the individual classifiers. The class, which receives the largest number of votes, is then selected as the consensus (majority) decision.

2.4.4. Sum Rules

Starting from (7) and computing the sum of posterior probabilities of every classifier we obtain *assign* $X \to C_i$ *if*

$$\max_{i=1}^{m} P(C_i \mid X) = \max_{i=1}^{m} \sum_{j=1}^{k} P_j(C_i \mid X)$$
$$for \;\; i = 1,2,...,m; j = 1,2,\ldots,k; \tag{15}$$

2.4.5. Product Rule

Starting from (7) and bounding the product of posterior probabilities from above we obtain

$assign \;\; X \to C_i \;\; if$

$$\max_{i=1}^{m} P(C_i \mid X) = \max_{i=1}^{m} \prod_{j=1}^{k} P_j(C_i \mid X)$$
$$for \;\; i = 1,2,...,m; j = 1,2,\ldots,k; \tag{16}$$

In this subsection, five methods of integration rules were introduced: max rule, min rule, majority vote, sum rule and product rule. All of these rules were given the simplest forms. Weight of every classifier is not considered.

3. Experiments and Discussion

In order to validate our decision rule, we firstly select the dataset of KDD CUP 1999[9] (ab. KDD99) to train and test our model. Then we select the letter dataset from the UCI repository of machine learning databases[10]. The reason to choose these dataset is KDD99 dataset is representation of dataset with large numbers of data, and the UCI letter dataset is representation of dataset with less data. And we give the recognition results respectively.

3.1. Datasets

For KDD99 dataset, our task is to build a predictive model (i.e. a classifier) capable of distinguishing between "bad" connections, called intrusions or attacks, and "good" normal connections. Attacks fall into four main categories: DOS, R2L, U2R, probing. There are 494021 samples we used in a 10 percent subset of KDD99 dataset, and 41 feature values in each sample. All feature values are divided into two data types: continuous and discrete. We transform discrete values into integers.

The UCI letter dataset has 26 classes with 16 dimensions. The objective is to identify each of a large number of black-and-white rectangular pixel displays as one of the 26 capital letters in the English alphabet. The character images were based on 20 different fonts and each letter within these 20 fonts was randomly distorted to produce a file of 20,000 unique stimuli. Each stimulus was converted into 16 primitive numerical attributes (statistical moments and edge counts) which were then scaled to fit into a range of integer values from 0 through 15.

3.2. Deal with Dataset

For KDD99 dataset, we select two-part of its training file as training dataset U_1, and half of residual samples as integration training set U_2, and another half of that as integration testing dataset U_3.

For UCI letter dataset, we typically train on the first 12000 samples(U_1), and 4000 samples as integration training set U_2, then use the resulting model to predict the letter category for the remaining 4000(U_3).

We divide training set U_1 into 10 subsets with equal samples randomly to train 10 classifiers like above subsection. Then we get ten classifiers. We use U_2 to train these classifiers to obtain weight of every classifier, and use U_3 to test our combining classifiers at abstract level.

3.3. Results Analysis and Discussion

Table 1 shows error rate of single classifier in U_2 and U_3, and Table 2 shows error rate of the results with combining classifiers in KDD99 dataset using different decision rules.

Table 1. Error rate of single classifier

Classifier NO.	U_2	U_3
1	0.0706	0.0690
2	0.0607	0.0623
3	0.0682	0.0628
4	0.0671	0.0659
5	0.0702	0.0710
6	0.0613	0.0623
7	0.0652	0.0612
8	0.0714	0.0702
9	0.0633	0.0652
10	0.0608	0.0614
Mean Error	0.0658	0.0651

Table 2 (a). Comparison of results

Number of Classifiers	*Max Rule*	*Min Rule*	*Majority Vote*
3	0.0524	0.0736	0.0632
5	0.0504	0.0803	0.0502
10	0.0426	0.0856	0.0438

Table 2 (b). Comparison of results

Number of Classifiers	*Sum Rule*	*Product Rule*	*MAW Rule*
3	0.0621	0.0628	0.0476
5	0.0635	0.0624	0.0481
10	0.0446	0.0440	0.0416

Table 3 shows error rate of single classifier in U_2 and U_3, and Table 4 shows error rate of the results with combining classifiers in the UCI letter dataset using different decision rules.

Table 3. Error rate of single classifier

Classifier NO.	U_2	U_3
1	0.2803	0.2760
2	0.2747	0.2783
3	0.2675	0.2650
4	0.2653	0.2635
5	0.2975	0.2802
6	0.2850	0.2828
7	0.3155	0.3225
8	0.2690	0.2657
9	0.2680	0.2613
10	0.3337	0.3297
Mean Error	0.2875	0.2825

Table 4(a). Comparison of results

Number of Classifie rs	*Max Rule*	*Min Rule*	*Majority Vote*
3	0.2348	0.2550	0.2222
5	0.2123	0.2645	0.2103
10	0.2003	0.2743	0.1903

Table 4(b). Comparison of results

Number of Classifie rs	*Sum Rule*	*Product Rule*	*MAW Rule*
3	0.2102	0.2110	0.2078
5	0.1953	0.1942	0.1992
10	0.1902	0.1901	0.1894

According to comparing Table 1 with Table 2, and Table 3 with Table 4, we draw a conclusion that combining multiple classifiers lead to less error than single classifier in same dataset, which indicates integration of multiple classifiers can deduce error rate except min rule. We found the min rule fail this test on above datasets. The same conclusion was showed in [11]. The sum rule and the product rule have no discrepancy. The majority vote rule is better than the max rule. And MAW Rule has the best performance than other rules, and the best recognition rate can still achiever up to 95.84% in KDD99 dataset and 81.06% in letter dataset.

For the KDD99 dataset, because single classifier already has better recognition results, which the average recognition rate of single classifier is about 93.5%. After that MAW rule was applied, it is unconspicuous that the recognition rate increases about 2.3%. But for the letter dataset, it is remarkable that the recognition rate increases about 8.5% after MAW rule was used.

From Table 2 and Table 4, the model with more classifiers leads to less error. When the system has ten classifiers, five rules obtain the best recognition rate except min rule.

4. Conclusions

It is well know that naïve Bayesian classification cannot obtain good recognition rate on the letter dataset, which the average recognition rate of single naïve Bayesian classifier is about 72.5%. The result is dissatisfactory. Thus we introduce a classifier combining algorithms: MAW rule. Furthermore we introduce other algorithms: Max rule, Min rule, Majority vote rule, Sum rule, and Product rule. Then we apply these rules in KDD99 dataset and the letter dataset. According to experiments, MAW rule has the best performance. The recognition rate increases about 2.3% in KDD99 dataset. And it is remarkable that the recognition rate increases about 8.5% in letter dataset. We arrive at a conclusion that MAW rule can obviously amend recognition rate of naïve Bayesian classifier.

Acknowledgements

This paper is supported by the National Natural Science Foundation of China under Grant No. 60273087, and the Artificial Intelligence laboratory of School of Computer Science, Northwest Polytechnical University, China.

References

[1] J. Kittler, M. Hatef, Duin R. P. W., and J. Matas, “On combining classifiers”, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol 20, No. 3, pp. 226-239, Mar. 1998.

[2] Duin R. P. W., and Tax D. M. J., “Experiments with Classifier Combining Rules”, Proceedings of 1st International Workshop on Multiple Classifier System, Cagliari, Italy, pp. 16-29, Jun. 2000.

[3] Xu L, Krzyzak A, and Suen C Y, “Methods for Combining Multiple Classifiers and Their Applications to Handwriting Recognition”, IEEE Transactions on Systems，Man，and Cybernetics, Vol 22, No. 3, pp. 418-435, May. 1992.

[4] Xiaoguang Lv, Yunhong Wang and AK Jain, "Combining Classifiers for Face Recognition", Proc. ICME 2003, IEEE International Conference on Multimedia &Expo, Vol 3, No. 7, pp. 13-16, Jul. 2003.

[5] C. Dietrich, F. Schwenker, and G. Palm, “Classification of time series utilizing temporal and decision fusion”, Proceedings of Multiple Classifier Systems (MCS), Cambridge, pp. 378-387. Feb. 2001.

[6] Marina Skurichina, and Duin R. P. W., “Bagging，Boosting and the Random Subspace Method for Linear Classifiers”, Pattern Analysis & Applications，Vol 5, No.2, pp. 121-135, Feb. 2002.

[7] H. Bunke, and A.Kandel, Hybrid Methods in Pattern Recognition, World Scientific Publishing, Singapore, 2002.

[8] Richard O. Duba, Peter E. Hart, and David G. Stork, Pattern Classification(2nd Edition), John Wiley & Sons, Inc., 2001.

[9] http://kdd.ics.uci.edu/databases/kddcup99

[10] http://www.ics.uci.edu/~mlearn/MLRepository.html

[11] KOU Zhong-Bao, and Zhang Chang-Shui, “Multi-Agent Based Classifier Combination”, Chinese Journal of Computer, Vol 26, No.2, pp. 174-179, Feb. 2003.