Journal of the Korean
Data & Information Science Society
2001, Vol. 12, No. 2, pp. 83 ~ 93

A Kernel Approach to Discriminant Analysis for Binary Classification[1]

Yangkyu Shin[2]

Abstract

We investigate a kernel approach to discriminant analysis for binary classification as a machine learning point of view. Our view of the kernel approach follows support vector method which is one of the most promising techniques in the area of machine learning. As usual discriminant analysis, the kernel method can discriminate an object most likely belongs to. Moreover, it has some advantage over discriminant analysis such as data compression and computing time.

Key Words and Phrases: discriminant analysis, kernel method, support vector method, machine learning.

1. Introduction

Linear discriminant analysis is a traditional statistical method to classify cases into the values of a categorical dependent, usually a dichotomy. The procedure, however, fails for a nonlinear problem. Recently, various approaches have been proposed to address this problem. Mika, Rätsch and Weston(1999) propose a nonlinear classification technique based on Fisher's discriminant. Their technique uses the kernel trick which allows the efficient computation of Fisher discriminant in feature space.

[1]This research was supported by a grant from Kyungsan University, Kirin Foundation in 2000.

[2]Professor, Faculty of Information Science, Kyungsan University, Jumchon-dong, Kyungsan, Korea

Baudat and Anouar(2000) present a generalized discriminant analysis to deal with nonlinear discriminant analysis using kernel function operator. The key idea of these two proposals is using a kernel method, which has also addressed in Campbell(2000).

The kernel method, however, goes back to Vapnik(1995) in his *support vector machine*(SVM), which is not a machine but a computing method. The support vector method transforms the input using a nonlinear mapping so that a straight line in the new space does not look straight in the original instance space. Thus, a linear model constructed in the new space can represent a nonlinear decision boundar in the original space. Guyon and Stork(2000) state that support vector method can be understood in traditional linear discriminant techniques. In this paper, we investigate linear discriminant analysis can be described in support vector method as a nonlinear extension for binary classification task.

2. Binary Classification

Suppose we are given a set $\{(x_1, y_1), ..., (x_l, y_l)\}$ of empirical data, also called training examples, where $x_i \in X$ and $y_i \in \{+1, -1\}$ for each $i = 1, ..., l$. Here, the domain X is some nonempty set that the patterns x_i are taken from and the y_i are called labels or targets. The goal of general binary classification task is to find a classifier with the decision function $f(x)$, such that $y = f(x)$, where y is the class label for x. The performance of the binary classifier is measured in terms of classification error which is defined in (1).

$$E(y, f(x)) = \begin{cases} 0 & \text{if } y = f(x) \\ 1 & \text{otherwise} \end{cases} \tag{1}$$

Suppose there is a learning method with adjustable parameters λ. Given the binary classification task, the method will tune its parameters λ to learn the mapping $x \to y$. This will result in a possible mapping $x \to f(x, \lambda)$ which defines this particular learning method. The performance of this method can be measured by the expectation of test error as follows:

$$R(\lambda) = \int E(y, f(x, \lambda)) dP(x, y) \tag{2}$$

This is called expected risk or actual risk. It requires at least an estimate of $P(x, y)$, which is not available for most classification tasks. Hence, one must settle for the empirical risk measure which is defined in (3). This is just a measure of the mean error over the available training data.

$$R_{emp}(\lambda) = \frac{1}{l} \sum_{i=1}^{l} E(y, f(x, \lambda)) \tag{3}$$

Most training algorithms for learning methods implement *empirical risk mimization* (ERM), i.e. minimize the empirical error using maximum likelihood estimation for the parameters λ. These conventional training algorithms do not consider the capacity of the learning method and this can result in over fitting, i.e. using a learning method with too much capacity for a particular problem.

In contrast with ERM, the goal of *structural risk minimization* (SRM) is to find the learning method that yields a good trade-off between low empirical risk and small capacity. There are two major problems in achieving this goal.

- The SRM requires a measure of the capacity of a particular learning method or at least an upper bound of this measure.

- An algorithm to select the desired learning method according to SRM's goal is needed. One can divide the entire class of methods into nested subsets with decreasing capacity. Then one can train a series of methods, one for each subset, using the ERM principle. Finally the method that gives the best trade-off can be selected. This can be a very difficult task. An alternative is to define a learning method with variable capacity and a corresponding training algorithm that minimizes both the empirical error and capacity of that method.

To address these two problems, Vapnik(1995) proposed the concept of "Vapnik Chervenenkis (VC) confidence" and support vector methods. These will be described in sequence.

3. VC dimension and VC confidence

VC confidence is defined as the second term on the right hand side of (4).

$$R(\lambda) \leq R_{emp}(\lambda) + \sqrt{(\frac{h(log(2l/h)+1) - log(\eta/4)}{l})} \tag{4}$$

In (4), η is chosen such that $0 \leq \eta \leq 1$, and the bound will hold with probability $1-\eta$. h is the VC dimension and a measure of the notion of capacity of a learning method.

VC diemnsion can be defined for various classes of learning methods. It is defined as the maximum number of points that can be "shatter" by the method. Considering the binary classification problem, a set of l points have 2^l possible label assignments. If a learning method with the mapping $f(x, \lambda)$ can correctly assign all possible labels, then that set of points is shattered by this learning method. It can be shown that a method with oriented hyper-planes in R^N as its mapping function, has a VC dimension of $N+1$.

It is clear that a learning method with large capacity (hence large VC dimension) will give low empirical risk but the VC confidence interval is also large for that learning method, i.e. the method does not generalize well. By measuring this bound, one can select a learning method that give the lowest expected risk. The selected method will give better generalization.

In Vapnik(1995), it is stressed that the VC dimension and hence the capacity does not depend on the number of free parameters directly, i.e. a learning method with a larger number of free parameter need not have larger capacity. And the reverse is also true. The method with mapping function $f(x, \lambda) = \theta(sin(\lambda x))$, where $x, \lambda \in R$ has an infinite VC dimension.

3.1 Linear Support Vector Method: A Maximum Margin Classifier

Consider a binary classifier with oriented hyper-planes. The decision function of the classifier is $f(x, \lambda) = sgn(w \cdot x + b)$, where $sgn(\alpha)$ is the sign of α and $x, w \in R^N$. If the training data is linearly separable, then a set of $\{w, b\}$ pairs can be found such that the constraints in (5) are satisfied.

$$\forall i, \; y_i(x_i \cdot w + b) - 1 \geq 0 \tag{5}$$

Notice that there is ambiguity in the magnitude of w and b. They can be arbitrary scaled such that $|f(x_p, \{w, b\})| = 1$, where x_p is the training data nearest to the decision plane.

Intuitively, the classifier with the largest margin will give lower expected risk (2), i.e. better generalization. Comparing the two different decision planes, the classifier with smaller margin will have higher expected risk. The margin for this linear classifier is just $2/ \parallel w \parallel$. Hence to maximize the margin, one needs to minimize the $\parallel w \parallel$ with the constraints in (5). In short, the training of this classifier is achieved by solving a linearly constrained optimization problem.

This optimization problem is solved using the Lagrangian formulation. The Lagrangian for minimizing $\parallel w \parallel$ is shown in (6).

$$L_p = \frac{1}{2} \parallel w \parallel^2 - \sum_{i=1}^{l} \lambda_i y_i (w \cdot x_i + b) + \sum_{i=1}^{l} \lambda_i \tag{6}$$

This needs to be minimized with respect to w and b, and it is simultaneously required that $\frac{dL_p}{d\lambda} = 0$ and (7) be satisfied.

$$\forall i, \ \lambda_i \geq 0 \tag{7}$$

Applying the prima-dual formulation, this can be achieved by maximizing L_p subject to the constraints that $\frac{dL_p}{dw} = 0$, $\frac{dL_p}{db} = 0$ and also satisfying (7).

From the constraints in the dual formulation, the condition in (8) and (9) is obtained.

$$\begin{aligned} \frac{dL_p}{dw} &= 0 \\ w - \sum_i \lambda_i y_i x_i &= 0 \\ w &= \sum_i \lambda_i y_i x_i \end{aligned} \tag{8}$$

$$\begin{aligned} \frac{dL_p}{db} &= 0 \\ \sum_i \lambda_i y_i &= 0 \end{aligned} \tag{9}$$

These can be substituted into (6) to give the new Lagrangian in (10). The training problem is transformed to maximize (10) with respect to constraints in (5), (7) and (9).

$$L_D = \sum_i \lambda_i - \frac{1}{2} \sum_{i,j} \lambda_i \lambda_j y_i y_j (x_i \cdot x_j) \tag{10}$$

This is a convex quadratic programming problem, since $\| w \|$ is convex and all constraints are linear. For this particular problem, the Karush-Kuhn-Tucker(KKT) conditions are (5), (7) and (9) with an additional condition as stated in (11).

$$\forall i,\ \lambda_i (y_i (w \cdot x_i + b) - 1) = 0 \tag{11}$$

From these KKT condition, the following conclusion can be made.

- if $\lambda_i = 0$, then $y_i(w \cdot x_i + b) \geq 1$
- if $\lambda_i > 0$, then $y_i(w \cdot x_i + b) = 1$

At this point it would be beneficial to consider the significance of the λ value for each set of training data. Those training data with nonzero λ values will fall on the +1 or -1 plane, i.e. these are the data that contribute to defining the decision boundary. If the other data are removed and the classifier is retrained on the remaining data, the training will result in the same decision boundary. These data points are called the *support vectors*. Support vectors with larger λ are more important, since they have stronger influence on the decision boundary.

Solving (10) will give the value for all λ. From (8), w is a linear combination of all the training data x_i(only those training points with nonzero λ value contribute). b is found by using (11), by selecting any training data with nonzero λ values. It is numerically wiser to average b over all such training data.

3.2 Extending Support Vector Method to a Soft Margin Classifier

The above algorithm can be extended to non-separable data. The correct classification constraints in (5) is revised by adding a slack variable ξ to (13). This will allow some points to be misclassified. The training algorithm will need to minimize the cost function in (12), i.e. a trade-off between maximum margin and classification

error. The selection of C and k define the cost of constraint violation. Vapnik(1995) defines a more general error function.

$$W(\lambda) = \| w \| + C(\sum_i \xi_i)^k \tag{12}$$

For positive integers k, the above optimization problem is a convex programming problem (if $k = 1$ or 2, it is a quadratic programming problem). For simplicity, k is usually set to 1. Using a similar approach as in the separable case, the training results in a similar quadratic programming problem as shown in (10) but now the λ value is bound by C in (13).

$$\begin{aligned} \forall i,\ y_i(x_i \cdot w + b) - 1 + \xi_i &\geq 0 \\ \forall i,\ \xi_i &\geq 0 \\ \forall i,\ 0 \leq \lambda_i &\leq C \end{aligned} \tag{13}$$

This bound will limit the search space for the quadratic programming problem, i.e. the possible range for the λ value. A larger search space will generally slow down the quadratic programming optimizer.

It can be proved that, for any misclassified training data, x_i, the corresponding λ_i must be at the upper bound. This can be understood by imagining that a particular data point is trying to assert a stronger influence on the boundary so that it can be classified correctly, by increasing the λ value. When the λ value reaches its maximum bound, it cannot increase its influence further, hence this point will stay misclassified. This analogy is consistent with the fact that C, the upper bound for λ is the trade-off between maximum margin and classification error. A higher C value will give a larger penalty for classification error, and this means λ is allowed to have a larger value; hence, each misclassified data can assert a stronger influence on the boundary.

4. Nonlinear Support Vector Method

So far the support vector method classifier can only have a linear hyper-plane as its decision surface. This formulation can be further extended to build a nonlinear

support vector method. The motivation for this extension is that a support vector method with nonlinear decision surface can classify nonlinearly separable data. Consider a mapping Φ, (14), which maps the training data from R^N to some higher Euclidean space H, which possibly has infinite dimensions.

$$\Phi : R^N \rightarrow H \tag{14}$$

In this high dimension space, the data can be linearly separable, hence the linear support vector method formulation above can be applied to these data. In the support vector method formulation, the training data only appear in the form of dot products, $x_i \cdot x_j$, i.e. in (10), the same is true in the decision function $w \cdot x + b (w = \sum_i \lambda_i y_i x_i)$. These can be replace by dot products in the Euclidean space H, i.e. $\Phi(x_i) \cdot \Phi(x_j)$.

The dot product in the high dimension space can also be replace by a kernel function, (15).

$$K(x_i, x_j) = \Phi(x_i) \cdot \Phi(x_j) \tag{15}$$

By computing the dot product directly using a kernel function, one avoid the mapping $\Phi(x)$. This is desirable because H has possibly infinite dimensions and $\Phi(x)$ can be tricky or impossible to compute. Using a kernel function, one need not explicitly know what Φ is. By using a kernel function, a support vector method that operates in infinite dimensional space can be constructed. This also means that w cannot be precomputed, and that the decision function will be replaced by (16), where for every new test data, the kernel function for each support vector need to be recomputed.

$$f(x, \{\lambda, b\}) = \sum_i \lambda_i y_i K(x_i, y_j) + b \tag{16}$$

The question to ask next is "what kernel function can be used?" i.e. is there any constraint on the type of kernel function suitable for this task. For any kernel function suitable for support vector method, there must exist at least one pair of $\{H, \Phi\}$, such that (15) is true, i.e. the kernel function represents the dot product of the data in H. The kernel that has these properties satisfies the Mercer's condition (Vapnik, 1995), i.e. for any $g(x)$ with finite L_2 norm of (17), (18) must hold.

$$\int g(x)^2 dx < \infty \tag{17}$$

$$\int\int K(x,y)g(x)g(y)dxdy \geq 0 \tag{18}$$

Any positive definite kernel satisfies this condition. It can be proved that the Gaussian radial basis kernel function $K(x,y) = exp(- \parallel x - y \parallel^2 /2\sigma^2)$ satisfy the the condition. It is interesting to note that by choosing different kernel functions, the support vector method can emulate some well known classifiers.

5. Support Vector Method Training and Structural Risk Minimization

Finally the most important question is, "does support vector method training actually implement SRM principle?" and if yes, "how?" i.e. can one prove that support vector method training actually minimizes the VC dimension and empirical error at the same time. From section 3, the VC dimension of a nonlinear support vector method is $N+1$, where N is the dimensionality of space H. In Burges(1998), the dimensionality of space H for a p-degree polynomial and residual basis function kernel is shown to be $\binom{d_L+p-1}{p}$ and ∞ respectively, where d_L is the dimensionality of the original low dimension space. Hence support vector method could have a very high VC dimension.

Vapnik(1995) stated that the VC dimension is bounded by the inequality in (19).

$$h \leq min[\lceil R^2A^2 \rceil, N] + 1 \tag{19}$$

All training vectors are bounded by a sphere of radius R, and L_2 norm of w is bound by A in (20), i.e. $\parallel w \parallel \leq A$.

$$\begin{aligned} \parallel w \parallel &= \sqrt{w \cdot w} \\ &= \sqrt{\sum_i \lambda_i \Phi(x_i) \cdot \sum_j \lambda_j \Phi(x_j)} \\ &= \sqrt{\sum_i \sum_j \lambda_i \lambda_j K(x_i, x_j)} \end{aligned} \tag{20}$$

R can be found by first estimating the centre of the sphere that encloses all the training data. This is a convex quadratic programming problem, which can be solve using very similar techniques to those used in the support vector method training, i.e. using Lagrangian and prima-dual formulation. Burges(1998) shows that the center is given by (21) (λ is the

Lagrange multiplier). Then R is computed using (22).

$$\Phi(T) = \sum_i \lambda_i \Phi(x_i) \tag{21}$$

$$\begin{aligned} R &= \max_i \sqrt{(\Phi(x_i) - \Phi(T)) \cdot (\Phi(x_i) - \Phi(T))} \\ &= \max_i \sqrt{\Phi(x_i) \cdot \Phi(x_i) - 2\Phi(x_i) \cdot \Phi(T) + \Phi(T) \cdot \Phi(T)} \\ &= \max_i \sqrt{K(x_i, x_i) - 2K(x_i, T) + K(T, T)} \end{aligned} \tag{22}$$

By analysing the VC dimension of a gap-tolerant classifier, claimed that (19) actually gives the VC dimension of a gap-tolerant classifier but not the VC dimension of a support vector method classifier.

6. Conclusion

Discriminant analysis is a traditional statistical method to determine which variables discriminate between two or more naturally occurring groups. On the same line of research, however, support vector method is founded to be a capable learning algorithm on the basis of kernel method. It has the ability to handle various recognition problems and gives reasonable performance. Based on the principle of SRM, support vector methods have advantage over other classifiers. The problem of capacity control is embedded in the training algorithm. This allows the most generalized classifier to be found for a particular problem.

In this paper, we show the discriminant analysis can be understood in kernel approach to binary classification task. As an application to the kernel approach, support vector method has been studied. The formulation of support vector method is elegant in that it is simplified to a convex quadratic programming problem. Theoretically the training is guaranteed to converge to a global optimal. This ability to select the training data that defines the discriminant function could have many applications other than in binary classification.

References

1. Baudat, G. and Anouar, F. (2000). Generalized discriminant analysis using a kernel approach. *Neural Computation*, Vol. 12, No. 10, pages 2385–2404, MIT Press.

2. Burges, C. (1998). *A Tutorial on Support Vector Machines for Pattern Recognition.* Kluwer Academic Publishers, Boston.

3. Campbell, C. (2000). An introduction to kernel methods. *Radial Basis Function Networks: Design and Applications*, pages 1–31, Springer Verlag.

4. Gammerman, A. (1997). *Machine Learning: Progress and Prospects*, Royal Holloway, University of London.

5. Guyon, I. and Stork, D. (2000). Linear discriminant and support vector classifiers. *Advances in Large Margin Classifiers*, pages 147–169, MIT Press.

6. Kwok, J. (1999). Automated text categorization using support vector machine. *Proceedings of the International Conference on Neural Information Processing (ICONIP)*, pages 347–351.

7. Mika, S. and Rätsch, G. and Weston, J. (1999). Fisher discriminant analysis with kernels. *Neural Networks for Signal Processing IX*, pages 41–48, IEEE.

8. Vapnik, V. (1995). *The Nature of Statistical Learning Theory*, Springer Verlag, New York.