Automatic Parameter Selection for Polynomial Kernel

Shawkat Ali and Kate A. Smith
School of Business Systems
Monash University, Victoria 3800, Australia.
E-mail: {Shawkat.Ali, Kate.Smith}@infotech.monash.edu.au

Abstract - *Kernel is the heart of kernel based learning. To choose an appropriate parameter for a specific kernel is an important research issue in the data mining area. In this paper we propose an automatic parameter selection approach for polynomial kernel. The algorithm is tested on Support Vector Machines (SVM). The parameter selection is considered on the basis of prior information of the data distribution and Bayesian inference. The new approach is tested on different sizes of benchmark datasets with binary class problems as well as multi class classification problems.*

Keywords: SVM, polynomial kernel, data distribution, Bayesian inference.

1 Introduction

Pattern recognition problems have been introduced since Fischer's theory of linear discrimination in the mid 1930's. After that, in the 1960's Rosenblatt proposed the perceptron as a new approach to machine learning [1]. In more recent times, researchers have focused on solving recognition problems with the help of different learning algorithms.

Standard learning systems, say neural networks or decision trees, operate on input data after they have been transformed into feature vectors $F_1,\ldots,F_n \in F$ living in a d-dimensional space. In such a space, the data point can be separated by a surface, clustered, interpolated or otherwise analysed. The resulting hypothesis will then be applied to test points in the same vector space, in order to make predictions.

There are many cases, however, where the input data can't readily be described by explicit feature vectors: for example biosequences, images, graphs and text documents. For such datasets, the construction of a feature extraction module can be as complex and expensive as solving the entire problem. This feature extraction process not only requires extensive domain knowledge, but also it is possible to lose important information during the process. These extracted features play a key role in the effectiveness of a system [2].

Kernel, the most important ingredient of kernel based learning, is an effective alternative to explicit feature extraction. The building block of kernel based learning methods [3,4] is a function known as the kernel function, i.e., a function returning the inner product between the mapped data points in a higher dimensional space. The learning then takes place in the feature space, provided the learning algorithm can be entirely rewritten so that the data points only appear inside dot products with other data points. Several linear algorithms can be formulated in this way, for clustering, classification and regression. The most well known example of a kernel based system is the Support Vector Machine (SVM) [5, 4], but also the perceptron, principle component analysis, Nearest Neighbour, and many other algorithms have this property.

There are quite a good number of kernels available for kernel based learning. The problem of some kernels for SVM is the difficulty in fitting the appropriate parameters values. The linear, polynomial and radial basis function (RBF) are the most classical kernels used from the beginning of SVM research. The linear kernel is suitable for linear separable cases. But unfortunately most real world problems are not linearly separable. Joachims [6] argues SVMs are universal learners. In their basic form, SVMs learn linear threshold functions. Nevertheless, by a simple 'plug-in' of an appropriate kernel function, they can be used to learn polynomial classifiers, RBF networks, and three layer sigmoid neural nets.

Previous studies [3,7-11] show that there exists no special kernel, which has the best generalization performance for all kind of problem domains. Parrado-Hernandez, *et. al.*, argue it is not clear to get priori information which kernel function is the most appropriate, and it might be desirable to train a more flexible SVM by combining different kernels to solve a given problem [12]. Selection of the kernel parameter is an important research focus in the area of SVMs [13]. However, there is no literature concerning how to choose the best parameter for polynomial kernel. So, it

0-7803-8242-0/03/ $17.00 © 2003 IEEE

is really an important research area to choose the most significant parameter value for polynomial kernel.

In this paper, we will explore a simple yet practical approach to SVM classification choosing an appropriate kernel directly from the training data. The proposed approach is based on classical statistical theory and Bayesian inference [14]. Practical validity of the proposed approach is demonstrated using several low-dimensional and high dimensional as well as binary and multi class classification problems.

This paper is organized as follows. Section 2 gives a brief introduction to SVM. Section 3 describes the kernel methods. Section 4 describes the distribution test used for the datasets and proposed approach to automatic parameter selection. Section 5 describes the experimental setup and results. Finally, summary and discussions are given in section 6.

2 Support Vector Classifications

This section reviews the main ideas behind the SVM. We mainly formulate the multi class SVM to consider all the classes at a time. SVMs are a class of algorithms that combine the principles of statistical learning theory with optimisation techniques and the idea of a kernel mapping. They were introduced in [5], and in their simplest version they learn a separating hyperplane between two sets of points so as to maximise the margin (distance between the plane and closest point). The solution has several interesting statistical properties that make it a good candidate for valid generalisation. One of the main statistical properties of the maximal margin solution is that its performance does not depend on the dimensionality of the space where the separation takes place [2]. In this way, it is possible to work in very high dimensional spaces, such as those induced by kernels, without over fitting.

Let us consider a binary classification task with the data sequence $D_m = (\mathbf{x}_1, y_1),\ldots,(\mathbf{x}_m, y_m)$ having corresponding targets $y_1,\ldots,y_m$. The data is divided into two parts. The first part $D_n = (\mathbf{x}_1, y_1),\ldots,(\mathbf{x}_n, y_n)$ is used for training, while the remaining e = m-n pairs constitute the testing sequence:

$$T_s = (\mathbf{x}_{n+1}, y_{n+1}),\ldots,(\mathbf{x}_e, y_e) \quad (1)$$

The training sequence D_n is used to design a model and the testing part is used to evaluate the model performance.

The decision function for binary SVM is

$$f(x) = \arg\max_n[(\omega_n.\mathbf{x}) + \omega_0] \quad (2)$$

$$n = 1,\cdots,k.$$

The natural way to solve multi class problems is to construct several hyperplane for the multi classes in a single optimisation, for details see [15].

Then, we can obtain the multi class SVM decision function, which is defined as,

$$f(x) = \arg\max_n[\sum_{i=1}^{\ell}(c_i^n A_i - \alpha_i^n)(x_i.x) + \omega_0] \quad (3)$$

Where the parameters α_i^n are the solutions of the following quadratic optimization problem to be minimized:

$$W(\alpha) = 2\sum_{i,n}\alpha_i^n + \sum_{i,j,n}[-\frac{1}{2}c_j^{y_i}A_iA_j + \alpha_i^n\alpha_j^{y_i} - \frac{1}{2}\alpha_i^n\alpha_j^{y_i}](x_i.x_j) \quad (4)$$

Introducing the notation

$$A_i = \sum_{n=1}^{k}\alpha_i^n \qquad \text{and} \qquad c_i^n = \begin{cases} 1 & \text{if } y_i = n \\ 0 & \text{if } y_i \neq n \end{cases}$$

The subset of the examples which corresponding α_i^n values are different than zero are called Support Vectors (SVs).

Due to kernel implicit bias the decision functions for binary and multiclass data are formulated as follows:

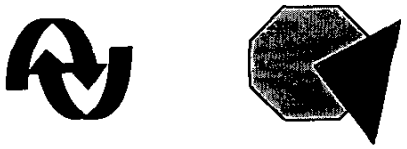
$$f(x) = \arg\max_n(\omega_n.\mathbf{x}) \quad (5)$$

$$f(x) = \arg\max_n[\sum_{i=1}^{\ell}(c_i^n A_i - \alpha_i^n)(x_i.x)] \quad (6)$$

We consider the linear kernel in eq. (6).

3 Kernel Methods

We now briefly describe a kernel function. Our aim is to introduce mathematical details of the polynomial kernel as followed by Mercer's theorem. We consider a situation where there is no alternative of the nonlinear discriminate function to classification. Figure 1. describes the two linearly non separable situations. In (a) it is clear that a classifier with a linear discriminant function will not perform well while in (b) the classes overlap each other and the optimal discriminant function is at least roughly linear.



(a) (b)

Figure 1. In (a) the optimal discriminant function is nonlinear while the optimal classifiers has no errors. In (b) the optimal discriminant function is linear while the classes overlap and thus the optimal classifier is not error free.

The real scenario of the classification problem is where the optimal discriminant function is often nonlinear. It should be mentioned that using a nonlinear discriminant function of course does not guarantee zero training error.

We will map the vectors $\mathbf{x}_i$, $i = 1,\ldots,n$, into a new space in the hope that the optimal separating hyperplane in new space performs better classifications than the optimal hyperplane in the original space. The mapping is $\Phi : \Re^d \to H$, where $\dim(H) \geq d$ and possibly $\dim(H) = \infty$ [16]. More specifically the mapping that will be considered is of the form

$$\Phi(\mathbf{x}) = \left(\sqrt{\lambda_1}\psi_1(\mathbf{x}), \sqrt{\lambda_2}\psi_2(\mathbf{x}), \ldots\right) \tag{7}$$

where λ_i and ψ_i are the eigen values and the normalized eigen functions of all integral operator $\ell_K : f \to \int \mathbf{K}(.,\mathbf{v}) f(\mathbf{v}) d\mathbf{v}$. In the SVM literature the space H is often called a feature space and the $\Phi(\mathbf{x}_i)'s$ are called feature vectors. Calculating the feature vectors can be computationally expensive, or even impossible, if the dimension of feature space is high or infinite. It should be noted that in the SVM algorithm all the calculations involving the $\Phi(\mathbf{x}_i)'s$ appear in the inner products. Instead of explicitly mapping the vectors into a high dimensional feature space and computing the inner product it is, under certain conditions, possible to use a function $\mathbf{K}(\mathbf{u},\mathbf{v})$ whose value directly gives the inner product between two vectors $\Phi(\mathbf{u})$ and $\Phi(\mathbf{v})$. A direct consequence is that by using $\boldsymbol{K}$ the inner products can be computed at roughly the same time in the feature space and in the original space. In the literature the function $\mathbf{K}(.,.)$ is usually called a kernel.

The classical kernels of SVMs are linear, polynomial and rbf. The most preferable kernel for nonlinear classification is polynomial or rbf.

3.1 Polynomial Kernel

Let us consider the pth degree polynomial kernel. In order to obtain explicit features $\Phi : \chi \to \Re$ we can expand the kernel function as follows

$$\begin{aligned}\left(\langle \mathbf{u},\mathbf{v}\rangle \chi\right)^p &= \left(\sum_{i=1}^{N} \mathbf{u}_i \mathbf{v}_i\right)^p = \left(\sum_{i_1=1}^{N} \mathbf{u}_{i1}\mathbf{v}_{i1}\right)\cdots\left(\sum_{i_p=1}^{N} \mathbf{u}_{ip}\mathbf{v}_{ip}\right) \\ &= \sum_{i_1=1}^{N}\sum_{i_p=1}^{N} \underbrace{\left(\mathbf{u}_{i1}\ldots\mathbf{u}_{ip}\right)}_{\Phi_i(\mathbf{u})} \underbrace{\left(\mathbf{v}_{i1}\ldots\mathbf{v}_{ip}\right)}_{\Phi_i(\mathbf{v})} = \langle \Phi(\mathbf{u})\Phi(\mathbf{v})\rangle\end{aligned} \tag{8}$$

Although it seems that there are N^p different features we see that two index vectors $\mathbf{i}_1$ and $\mathbf{i}_2$ lead to the same feature $\Phi_{i1} = \Phi_{i2}$ if they contain the same distinct indices the same number of times but at different positions [17], e.g., $\mathbf{i}_1$=(1,1,3) and $\mathbf{i}_2$=(1,1,3) both lead to $\Phi(\mathbf{u}) = \mathbf{u}_1\mathbf{u}_1\mathbf{u}_3 = \mathbf{u}_1^2\mathbf{u}_3$. One method of computing the number of different features Φ is to index them by an N-dimensional exponent vector $\mathbf{r} = (r_1,\ldots,r_N) \in \{0,\ldots,p\}^N$, i.e., $\Phi_{\mathbf{r}}(\mathbf{u}) = \mathbf{u}_1^{r_1}\ldots\mathbf{u}_N^{r_N}$. Since there are exactly p summands we know that each admissible exponent vector $\mathbf{r}$ must obey $r_1 + \ldots + r_{N=p}$. The number of different exponent vectors $\mathbf{r}$ is thus exactly given by

$$\binom{N+P-1}{p}$$

and for each admissible exponent vector $\mathbf{r}$ there are exactly

$$\frac{p!}{r_1!\ldots r_N!}$$

different index vectors $\mathbf{i} \in \{1,\ldots,N\}^p$ leading to $\mathbf{r}$. Hence the rth feature is given by

$$\Phi_{\mathbf{r}}(\mathbf{u}) = \sqrt{\frac{p!}{r_1!\ldots r_N!}}\cdot\mathbf{u}_1^{r_1},\ldots,\mathbf{u}_N^{r_N} \tag{9}$$

Finally note that the complete polynomial kernel is a pth degree polynomial kernel in an N+1-dimensional input space by the following identity

$$\left(\langle \mathbf{u},\mathbf{v}\rangle + b\right)^p = \left\langle\left(\mathbf{u},\sqrt{b}\right),\left(\mathbf{v},\sqrt{b}\right)\right\rangle^p \tag{10}$$

When b is positive the kernel is called inhomogeneous and correspondingly, homogeneous when b = 0. The inhomogeneous kernel avoids problems with the Hessians becoming zero in numerical calculation. To prove that this kernel is a Mercer kernel, it is sufficient to show that

$$\int\left(\sum_{i=1}^{d} \mathbf{u}_i\mathbf{v}_i\right)^p f(\mathbf{u}) f(\mathbf{v}) d\mathbf{u}\, d\mathbf{v} \geq 0 \tag{11}$$

where d is the dimension of the vectors and we have for simplicity set b = 0.

The final classifier with polynomial kernel is

$$f(x) = \underset{n}{\operatorname{argmax}}\left[\sum_{i=1}^{t}(c_i^n A_i - \alpha_i^n)\left(\sum_i (\mathbf{u}_i.\mathbf{v}_i) + b\right)^p\right] \tag{12}$$

The final classifier with the rbf kernel is

$$f(x) = \underset{n}{\operatorname{argmax}}\left[\sum_{i=1}^{t}(c_i^n A_i - \alpha_i^n)\left(\exp\left(-\frac{\|\mathbf{u}_i - \mathbf{v}_i\|^2}{2\sigma^2}\right)\right)\right] \tag{13}$$

where σ is the width of the rbf kernel.

4 Automatic Polynomial Parameter Selections

When a kernel is used it is often unclear what the properties of the mapping and the feature space are. It is always possible to make a mapping into a potentially very high dimensional space and to produce a classifier with no classification errors on the training set. However, then the performance of the classifier can be poor. On the other hand, it is possible that a classifier

with an infinite dimensional feature space performs well. Thus, the dimension of the feature space is not the essential quantity when choosing the right kernel [18]. Before choose the right kernel, it is important to measure the nature of the data, i.e., the distribution of the dataset. Our studies as like binary SVM, i.e., if the data distribution is normal then we suggest choosing the Gaussian or rbf kernel, otherwise polynomial kernel. However, we need to find out the optimal polynomial degree.

4.1 Interquartile Range (IQR)

It is easy way to gauge the normality of the data distribution by visually inspection using a histogram, normal probability plot, or dot plot. But for an automated method we need a machine-readable numeric value from some kind of distribution test. The chi-square test can solve this problem. It considers the dataset by different samples, then checks each sample property for normality. But the computational complexity is higher for this method. However, we suggest a method IQR based on classical statistics to overcome this limitation [19]. The IQR is used as a robust measure of scale. It considers the quartile values, one is lower and another is upper quartile. Quartiles can be defined at the 25th percentile, the 50th percentile, the 75th percentile and the 100th percentile. The IQR is the distance between the 25th percentile and the 75th percentile. The IQR is a measure of variability that can be appropriately applied with ordinal variables and therefore may be used especially in conjunction with non-parametric statistics.

The lower and upper quartile is

$$iq\_\ell = \beta(n+1) \tag{14}$$

$$iq\_u = \delta(n+1) \tag{15}$$

and round to the nearest integer. Where β indicates the lower and δ indicates the upper quartile.

Now, $$IQR = (n+1)(\delta - \beta) \tag{16}$$

Now the hypothesis is, if the data are approximately normal then $\frac{IQR}{s} \approx 1.3$, where s is the standard deviation of the population [19].

4.2 Polynomial Degree Selection

To choose an appropriate degree for polynomial kernel is an important issue. We compute the training error of the data for each possible degree and choose the best one for classification [14].

The data model is that an input vector x of length m multiplies a coefficient matrix **A** to produce an output vector y of length d, with Gaussian noise added;

$$y = \mathbf{A}x + e \tag{17}$$

$$e \sim N(0, \mathbf{V}) \tag{18}$$

$$p(y \mid x, \mathbf{A}, \mathbf{V}) \sim N(\mathbf{A}x, \mathbf{V}) \tag{19}$$

This is the conditional model for y only.

The scenario is that we are given a dataset of exchangeable pairs $D = \{(y_1, x_1), \cdots, (y_N, x_N)\}$. Collect $\mathbf{Y} = [y_1 \cdots y_N]$ and $\mathbf{X} = [x_1 \cdots x_N]$. The distribution of **Y** given **X** under the model is

$$\begin{aligned} p(\mathbf{Y} \mid \mathbf{X}, \mathbf{A}, \mathbf{V}) &= \prod_i p(y_i \mid x_i, \mathbf{A}, \mathbf{V}) \\ &= \frac{1}{|2\pi\mathbf{V}|^{N/2}} \exp\left(-\frac{1}{2}\sum_i (y_i - \mathbf{A}x_i)^T \mathbf{V}^{-1}(y_i - \mathbf{A}x_i)\right) \\ &= \frac{1}{|2\pi\mathbf{V}|^{N/2}} \exp\left(-\frac{1}{2} tr\left(\mathbf{V}^{-1}(\mathbf{Y} - \mathbf{A}\mathbf{X})(\mathbf{Y} - \mathbf{A}\mathbf{X})^T\right)\right) \\ &= \frac{1}{|2\pi\mathbf{V}|^{N/2}} \exp\left(-\frac{1}{2} tr\left(\mathbf{V}^{-1}\left[\mathbf{A}\mathbf{X}\mathbf{X}^T\mathbf{A}^T - 2\mathbf{Y}\mathbf{X}^T\mathbf{A}^T + \mathbf{Y}\mathbf{Y}^T\right]\right)\right) \end{aligned} \tag{20}$$

A conjugate prior for **A** is the matrix normal density, which implies the posterior for **A** as well as for **Y** will be matrix normal. A random d by m matrix normal distributed with parameters **M, V**, and **K** if the density of **A** is

$$\begin{aligned} p(\mathbf{A}) &\sim N(\mathbf{M}, \mathbf{V}, \mathbf{K}) \\ &= \frac{|\mathbf{K}|^{d/2}}{|2\pi\mathbf{V}|^{m/2}} \exp\left(-\frac{1}{2} tr\left((\mathbf{A} - \mathbf{M})^T \mathbf{V}^{-1}(\mathbf{A} - \mathbf{M})\mathbf{K}\right)\right) \end{aligned} \tag{21}$$

Where **M** is d by d, and **K** is m by m. This distribution has two covariance matrices: **V** for the rows and **K** for the columns. If **V** is diagonal, then the rows of **A** are independent normal vector.

Now, we consider

$$\mathbf{S}_{xx} = \mathbf{X}\mathbf{X}^T + \mathbf{K}$$

$$\mathbf{S}_{yx} = \mathbf{Y}\mathbf{X}^T + \mathbf{M}\mathbf{K}$$

$$\mathbf{S}_{yy} = \mathbf{Y}\mathbf{Y}^T + \mathbf{M}\mathbf{K}\mathbf{M}^T$$

$$\mathbf{S}_{y|x} = \mathbf{S}_{yy} - \mathbf{S}_{yx}\mathbf{S}_{xx}^{-1}\mathbf{S}_{yx}^T$$

Then the likelihood (20) times conjugate prior (21) is

$$\begin{aligned} p(\mathbf{Y}, \mathbf{A} \mid \mathbf{X}, \mathbf{V}) &\propto \exp\left(-\frac{1}{2} tr\left(\mathbf{V}^{-1}\left[\mathbf{A}\mathbf{S}_{xx}\mathbf{A}^T - 2\mathbf{S}_{yx}\mathbf{A}^T + \mathbf{S}_{yy}\right]\right)\right) \\ &\propto \exp\left(-\frac{1}{2} tr\left(\mathbf{V}^{-1}\left[(\mathbf{A} - \mathbf{S}_{yx}\mathbf{S}_{xx}^{-1})\mathbf{S}_{xx}(\mathbf{A} - \mathbf{S}_{yx}\mathbf{S}_{xx}^{-1})^T + \mathbf{S}_{y|x}\right]\right)\right) \end{aligned} \tag{22}$$

Returning the joint distribution (22) and integrating out A gives the evidence for linearity, with **V** known:

$$p(\mathbf{Y} \mid \mathbf{X}, \mathbf{V}) = \frac{|\mathbf{K}|^{d/2}}{|\mathbf{S}_{xx}|^{d/2}|2\pi\mathbf{V}|^{N/2}} \exp\left(-\frac{1}{2} tr\left(\mathbf{V}^{-1}\mathbf{S}_{y|x}\right)\right) \tag{23}$$

This can be also drive from (19), (21) and the properties of the matrix-normal distribution. The invariant prior reduces this to

$$p(\mathbf{Y} \mid \mathbf{X}, \mathbf{V}, \alpha) = \left(\frac{\alpha}{\alpha+1}\right)^{md/2} |2\pi\mathbf{V}|^{-N/2} \exp\left(-\frac{1}{2} tr\left(\mathbf{V}^{-1}\mathbf{S}_{y|x}\right)\right) \tag{24}$$

$$\mathbf{S}_{y|x} = \mathbf{Y}\mathbf{Y}^T - (\alpha+1)^{-1}\mathbf{Y}\mathbf{X}^T\left(\mathbf{X}\mathbf{X}^T\right)^{-1}\mathbf{X}\mathbf{Y}^T \tag{25}$$

By zeroing the gradient with respect to α, we find that the evidence is maximized when

$$\alpha = \frac{md}{tr\left(\mathbf{V}^{-1}\mathbf{Y}\mathbf{X}^T\left(\mathbf{X}\mathbf{X}^T\right)^{-1}\mathbf{X}\mathbf{Y}^T\right) - md} \tag{26}$$

This estimator behaves in a reasonable way: it shrinks more when N is small, m is large, or the noise

level V is large, in order to reduce over fitting. Equation (26) offered estimation for the optimal polynomial degree to solve classification problem.

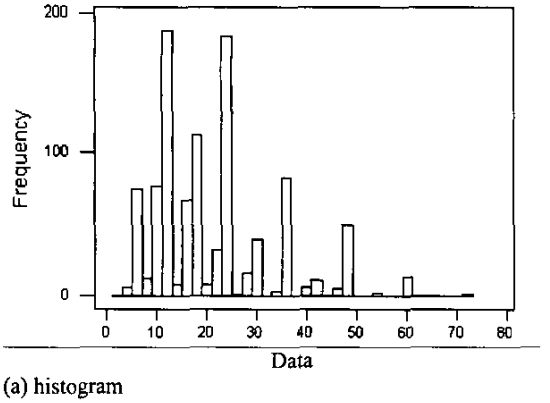
5 Experimental Results

Our aim was to test the efficiency of this new approach to automatic parameter selection for SVM classifications tasks. We used the data sets from two different sources. The dataset pid was collected from Knowledge Discovery Central [20] and rest from UCI repository [21]. We were considered binary class problems as well as multiclass problems. A split-sample strategy was used for application of the prediction methods. Randomly selected 70% of the data are used to learn the predictive model. The remaining 30% was used to test the model.

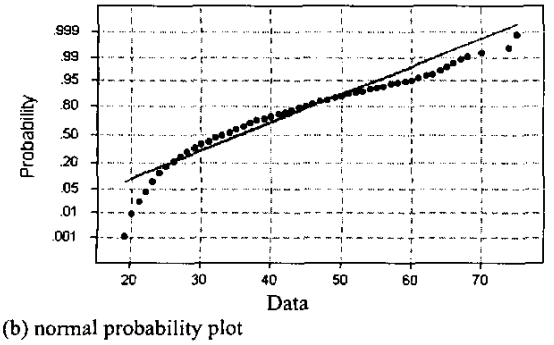
The datasets are described in Table 1 with number of attribute, instances and classes. The IQR values are also mentioned in the same table as well as the predicted polynomial degree. The experimental results have been introduced in Table 2, 3 and 4. We considered the polynomial degree from 2 to 5 and the rbf width from 0.2 to 1 with 0.2 intervals in classical approach. The datasets dna, germen, glass, h-d, pid and pima are not normally distributed and the rest of the datasets are normally distributed by the prior information from training data. The glass and pid datasets IQR values are less than normality value. So the classification error percentage of some non normal datasets with polynomial kernel is lower than rbf kernel. The polynomial with optimal degree (ploy_od) performed better classification accuracy for german, h-d and tictactoe dataset. It also performed equal error rate for dna, glass and pima within a single iteration compared to classical polynomial. Our method performed worse on those datasets which were perfectly normal distributed. The bold face in Table 2 indicates the lowest error percentage for a dataset. For some of datasets the training errors were reduced in our proposed method. The number of support vectors is similar between classical polynomial kernel and our approach.

Table 1. The datasets are described with number of attributes, instances, classes and IQR values.

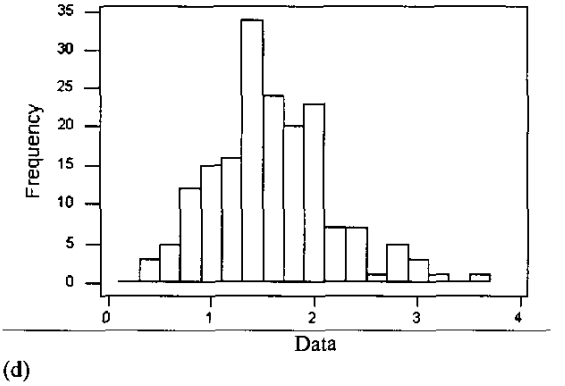
Dataset	# of Attribute	# of Instances	# of Class	IQR/s
bupa	6	345	2	1.34
dna	180	1186	3	1.77
german	25	1000	2	1.59
glass	9	214	6	0.86
h-d	13	303	2	1.43
pid	7	486	2	1.2
pima	8	768	2	1.44
tictactoe	9	958	2	1.35
wine	13	178	3	1.3



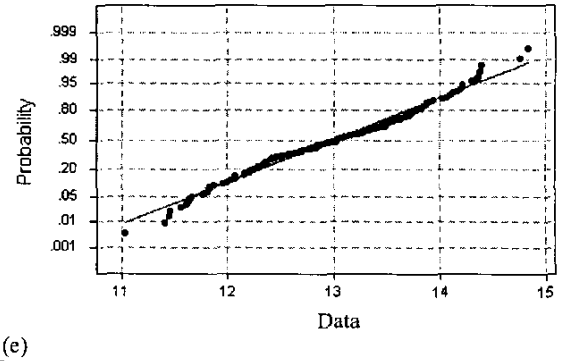
(a) histogram



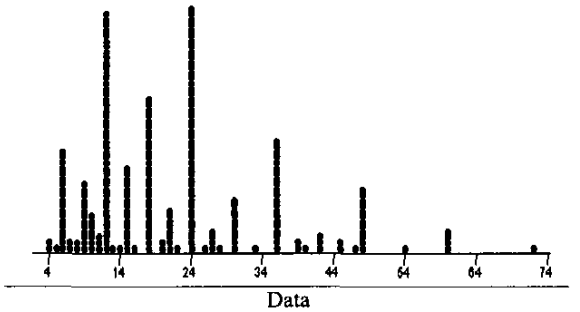
(b) normal probability plot



(d)



(e)



(c) dot plot

(f)

Figure 2. Graph for data distribution test. The german dataset is represented in (a), (b) and (c) and wine datasetis represented in (d), (e) and (f).

Figure 2 represents the graphical view of the german and wine datasets. From Table 1 we can see that wine is a normally distributed dataset but german is not. The german dataset has more outlier than wine. Figure 2 supports our argument and shows three well known normality test methods, histogram, normal probability test and dot plot. The histogram and the dot plots bell-shape of the german dataset is far different than normal distribution curve, on the other hand wine data set's graph has similarity with the bell-shape of normality curve.

Table 2. The percentage of error for the test data with polynomial and rbf kernel. The optimal polynomial degree is placed within bracket.

Dataset	poly				poly_od	rbf				
	2	3	4	5		0.2	0.4	0.6	0.8	1
bupa	55.45	52.48	55.45	**46.53**	**46.53** (5)	41.58	32.67	30.69	31.68	29.7
dna	22.63	18.7	**16.71**	20.68	**16.71** (4)	46	46.74	46.74	46.74	46.03
german	61.82	32.77	32.43	26.35	**23.99** (22)	33.78	32.77	31.42	30.41	29.05
glass	56.25	59.38	56.25	56.25	**35** (9)	40.63	51.56	51.56	51.56	51.56
h-d	35.23	25	26.14	25	**21.59** (9)	28.41	25	25	22.73	18.18
pid	**27.97**	30.77	34.27	34.97	34.97 (5)	25.87	21.68	24.48	24.48	25.17
pima	25.11	56.62	**21.92**	23.74	23.74 (6)	26.94	21.92	23.74	26.03	26.03
tictactoe	61.07	53.21	52.14	48.21	**43.93** (1)	3.93	4.29	3.93	6.43	10.36
wine	**6.9**	6.9	6.9	10.38	10.34 (9)	3.45	3.45	3.45	3.45	5.17

Table 3. The percentage of SVs for polynomial and rbf kernel.

Data set	poly				poly_od	rbf				
	2	3	4	5		0.2	0.4	0.6	0.8	1
bupa	84.4	46.3	44.3	44.3	44.3	100	94.3	89.8	88.9	88.5
dna	67.34	86.19	97.23	97.83	97.23	68.88	68.88	68.88	68.88	68.88
german	36.2	49.9	84.2	100	100	100	100	99.4	93.8	88.2
glass	42	46	46	48	48.7	100	96	89.3	89.3	88.7
h-d	31.6	41.4	46.5	100	100	100	100	95.8	82.3	75.8
pid	71.1	40.2	38.8	39.4	38.8	100	90.7	79.6	75.2	74.3
pima	78.7	48.8	38.8	40.1	40.1	100	87.4	81.2	77.8	76.9
tictactoe	8	23.2	45.1	75.2	100	19.53	47.39	57.72	67.95	70.66
wine	22.5	27.5	30.8	30.8	100	100	100	85	63.3	55.8

6 Discussions

We proposed an approach for automatic parameter selection for polynomial kernel. We tested our algorithm on SVM. The algorithm is based on classical statistical theory

Table 4. Train error for polynomial classical kernel and with new approach.

Dataset	poly				poly_od
	2	3	4	5	
bupa	55.33	53.28	54.92	52.46	54.92
dna	0	0	0	0	0
german	61.08	0	0	29.12	7.67
glass	56.67	54	50.67	43.33	22.727
h-d	30.23	0	0	0	7.9
pid	26.53	28.57	22.16	21.57	22.16
pima	23.68	59.2	21.49	20.77	20.77
tictactoe	67.11	57.52	57.67	52.95	49.26
wine	0	0		0	0

IQR and Bayesian inference. We suggest to consider the IQR value from 1.3 to 1.4 for normality test. The main benefit of this technique is to reduce the cost of classification for learning algorithm and help to make a quick decision. Another prominent feature of the presented approach is that it is easy to search the appropriate value to test the dataset normality. Our method is five times faster than classical polynomial approach. The rbf kernel performed better on those datasets with an IQR values range 1.3~1.4. We suggest choosing polynomial kernel for those datasets with IQR values lower bound is below 1.3 and upper bound is above 1.4.

In our further research, we will test our method with larger dataset as well as larger number of classes. We have an additional target to test the method with 10 fold cross validation and also explore how to reduce the training error. This idea could be also explored with some other new kernels. Moreover, we have plans to study how to choose the optimal parameter value for rbf kernel.

7 References

[1] D. E. Rumelhart, G. E., Hinton, and R. J. Williams. Learning internal representations by error propagation. *In parallel distributed processing: Explorations in the macrostructures of cognition*, volume 1, pages 318-362, Cambridge, MA, 1986.

[2] H. Lodhi, C. Saunders, N. Cristianini, C. Watkins, J. Shawe-Taylor, Text classification using string kernels, Appeared in *Journal of Machine Learning Research*, 2003.

[3] N. Cristianini and J. Shawe-Taylor. *An introduction to support vector machines*, Cambridge University Press, Cambdridge, UK, 2000.

[4] V. Vapnik. *The Nature of Statistical Learning Theory*. Springer Verlag, New York, 1995.

[5] E. Boser, I. M. Guyon, and V. N. Vapnik. A training algorithm for optimal margin classifiers. In D. Haussler, editor, *Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory*, pages 144-152, Pittsburgh, PA, 1992. ACM Press.

[6] T. Joachims, Text categorization with support vector machines: learning with many relevant features. *In European Conference on Machine Learning (ECML)*, 1998.

[7] A. Zien, G. Rätsch, S. Mika, B. Schölkopf, C. Lemmen, A. Smola, T. Lengauer, and K.-R. Müller. Engineering support vector machine kernels that recognize translation initiation sites. German Conference on Bioinformatics, 1999.

[8] B. Schölkopf, and A. Smola. Learning with Kernels: Support Vector Machines, Regularization, Optimisation, and Beyond. *The MIT Press*, England, 2002.

[9] S. Ali, and A. Abraham, An Empirical Comparison of Kernel Selection for Support Vector Machines, *2nd International Conference on Hybrid Intelligent Systems, Soft Computing Systems: Design, Management and Applications*, IOS Press, The Netherlands, pp. 321-330, 2002.

[10] C. Burges and D. Crisp. Uniqueness of the SVM solution. In *Proceedings of the Twelfth Conference on Neural Information Processing Systems*. S. A. Solla, T. K. Leen, and K.-R. Müller (Eds.), MIT Press, 1999

[11] S. Ali, M. U. Chowdhury and S. R. Subramanya, Nonlinear Discrimination Using Support Vector Machine, *Proceedings of the18th International Conference on Computers and Their Applications*, pages 287-290, Hawaii, USA, March 26-28, 2003.

[12] E. Parrado-Hernandez, I. Mora-Jimenez, J. Arenas_Garcia, A. R. Figueiras-Vidal, A Navia-Vazquez, Growing support vector classifiers with controlled complexity, *Pattern Recognition*, page 1479-1488, v.36, 2003.

[13] W. Wang, Z. Xu, W. Lu and X. Zhang, Determination of the spread parameter in the Gaussian kernel for classification and regression, *Neurocomputing*, Elsevier Science, 2002.

[14] T. P. Minka, Baysian linear regression, MIT, USA, 2001.

[15] J. Weston., and Watkins, Multi-class support vector machines, presented at the Proc. ESANN99, M. Verleysen, Ed., Brussels, Belgium, 1999.

[16] V. Vapnik., *Statistical Learning Theory*. John Wiley & Sons, 1998.

[17] R. Herbrich., *Learning Kernel Classifiers Theory and Algorithms*. The MIT Press, England, 2002.

[18] P. Erasto., Support Vector Machines – Backgrounds and Practice, PhD Thesis, Rolf Nevanlinna Institute, 2001.

[19] W. Mandenhall and T. Sincich., Statistics For Engineering and the Sciences, 4th ed. *Prentice Hall*, 1995.

[20] Loh. W., Knowledge Discovery Central, Data Sets, <http://www.KDCentral.com/>, 2002.

[21] C. Blake., and C. J. Merz., *UCI Repository of Machine Learning Databases*. Irvine, CA: University of California, 1999.