Finite Sample Error Bound for Parzen Windows*

Peng Zhang and **Jing Peng**
Department of Electrical Engineering & Computer Science
Tulane University
New Orleans, LA 70118
{zhangp,jp}@eecs.tulane.edu

Norbert Riedel
Department of Mathematics
Tulane University
New Orleans, LA 70118
riedel@math.tulane.edu

Abstract

Parzen Windows as a nonparametric method has been applied to a variety of density estimation as well as classification problems. Similar to nearest neighbor methods, Parzen Windows does not involve learning. While it converges to true but unknown probability densities in the asymptotic limit, there is a lack of theoretical analysis on its performance with finite samples. In this paper we establish a finite sample error bound for Parzen Windows. We first show that Parzen Windows is an approximation to regularized least squares (RLS) methods that have been well studied in statistical learning theory. We then derive the finite sample error bound for Parzen Windows, and discuss the properties of the error bound and its relationship to the error bound for RLS. This analysis provides interesting insight to Parzen Windows as well as the nearest neighbor method from the point of view of learning theory. Finally, we provide empirical results on the performance of Parzen Windows and other methods such as nearest neighbors, RLS and SVMs on a number of real data sets. These results corroborate well our theoretical analysis.

Introduction

In machine learning, nonparametric methods such as Parzen Windows and nearest neighbor methods for descriptive as well as predictive modeling are widely used and well studied. In order to build a predictive model, class conditional probabilities are estimated from the sample data using these methods, and then the decision is made by choosing the class having the maximum class probability. Parzen Windows and nearest neighbors are in fact closely related (Duda, Hart, & Stork 2000), and are different in choosing "window" functions, thus local regions. Both of these methods have several attractive properties. They are easy to program–no optimization or training is required. Their performance can be very good on some problems, comparing favorably with alternative, more sophisticated methods such as neural networks. They allow an easy application of a reject option, where a decision is deferred if one is not sufficiently confident about the predicted class.

*This work was supported in part by Louisiana BOR Grant LEQSF(2002-05)-RD-A-29 and ARO Grant DAAD19-03-C-0111.

It is well known that in the asymptotic limit, the probability estimates by these techniques converge to true (unknown) probabilities. Thus the classifier based on the estimated probabilities will converge to the optimal Bayes decision rule. Also, a well-known observation is that at least half classification information in an infinite data set resides in the nearest neighbor. However, when the sample size is finite, little systematic work has been carried out so far on the performance analysis for these methods. Known error bounds for these methods are typically obtained empirically (Mitchell 1997). While empirical analysis can be justified statistically, it only provides limited insights into algorithms' performance.

Statistical learning theory (Vapnik 1998; Cucker & Smale 2001) provides a framework for analyzing error bounds for predictive models given a finite sample data set. A good predictive model is the one that minimizes empirical errors on the sample data, while controlling the complexity of the model. The error bound for such a model thus usually contains two parts: the empirical error and the approximation error.

In this paper, we first derive the Parzen Windows method as an approximation to RLS under appropriate conditions. We then establish a performance bound for the Parzen Windows method based on the error bound for RLS given finite samples. Thus, our contributions are: (1) Demonstrating the Parzen Windows method as an approximation to RLS; (2) Estimating an error bound for the Parzen Windows classifier and discussing when the method will not perform well. We also provide some indirect theoretical insight into the nearest neighbor technique and demonstrate their performance using a number of data sets.

Parzen Windows

Parzen Windows (Duda, Hart, & Stork 2000; Fukunaga 1990; Parzen 1962) is a technique for density estimation that can be used for classification as well. Using a kernel function, it approximates a given training data distribution via a linear combination of the kernels centered on the training points. Here, each class density is approximated separately and a test point is assigned to the class having maximal (estimated) class probability.

Let

$$f_p(x) = \sum y_i k(x_i, x) \tag{1}$$

where $x \in \mathbb{R}^n$ is a predictor vector, $y \in \mathbb{R}$ is the class label, $k()$ is a kernel function. The Gaussian kernel

$$k(x, x') = e^{-\frac{\|x-x'\|^2}{\sigma^2}}. \quad (2)$$

is commonly used. For the binary case, the resulting Parzen Windows classifier is very simple

$$\tilde{f}_p(x) = sign(\sum y_i k(x_i, x)). \quad (3)$$

Notice that multiplying it by any positive constant will not change it. The Parzen Windows does not require any training and can be viewed as a generalization of k-nearest neighbor techniques. Rather than choosing the k nearest neighbors of a test point and labeling the test point with the majority of its neighbors' votes, one can consider all points in the voting scheme and assign their weight by the kernel function. With Gaussian kernels, the weight decreases exponentially with squared distance, so far away points are practically irrelevant. The width σ of the Gaussian kernel determines the relative weighting of near and far points. Tuning this parameter controls the predictive power of the system. Although the Parzen Windows classifier performs well asymptotically (Duda, Hart, & Stork 2000), it may fail to do so in applications with limited samples.

Regularized Least Squares Method

The regularized least squares method has been studied for a long time, under different names. In statistics, ridge regression (Hoerl & Kennard 1970) has been very popular for solving badly conditioned linear regression problems. After Tikhonov published his book (Tikhonov & Arsenin 1977), it was realized that ridge regression uses the regularization term in Tikhonov's sense. In the 1980's, weight decay was proposed to help prune unimportant neural network connections, and was soon recognized (J. Hertz & Palmer 1991) that weight decay is equivalent to ridge regression.

In the framework of statistical learning theory, the RLS algorithm has been revisited in (Poggio & Smale 2003), and as regularization networks in (Evgeniou, Pontil, & Poggio 2000). Most recently, the error bound for it given a finite sample data set was developed in (Cucker & Smale 2002). Here we briefly summarize the theory.

The RLS algorithm minimizes the following regularized functional:

$$\frac{1}{m}\sum_{i=1}^{m}(y_i - f(x_i))^2 + \lambda\|f\|_k^2, \quad (4)$$

where λ is the regularization parameter, and $\|f\|_k^2$ the norm in the hypothesis space $\mathcal{H}_k$ induced by the kernel k.

The minimizer exists and is unique and is given by (Poggio & Smale 2003)

$$f_{\lambda,z}(x) = \sum_{i=1}^{m} c_i k(x, x_i). \quad (5)$$

where the coefficients c_is are solved by the key algorithm

$$(m\lambda I + K)c = y. \quad (6)$$

Let $f_{\lambda,z}$ be the optimum for the regularized problem as exhibited in Equation (4), and f_ρ the true input-output function, then

$$\int (f_{\lambda,z} - f_\rho)^2 d\rho_X \leq S(\lambda) + A(\lambda), \quad (7)$$

where $A(\lambda)$ (the approximation error in this context) is

$$A(\lambda) = \lambda^{1/2}\|L_k^{-\frac{1}{4}} f_\rho\|^2. \quad (8)$$

L_k is an operator defined as $L_k f(x) = \int_X f(x')k(x, x')d\rho_X$. The sample error is

$$S(\lambda) = \frac{32M^2(\lambda + C_k)^2}{\lambda^2} v^*(m, \delta), \quad (9)$$

where M is a positive constant which is chosen to satisfy $|f(x) - y| \leq M$, and $v^*(m, \delta)$ is the unique solution of

$$\frac{m}{4}v^3 - \ln(\frac{4m}{\delta})v - c_v = 0. \quad (10)$$

$c_v > 0$ is a constant. C_k is defined by:

$$C_k = max\{1, \sup_{x,t\in X} |k(x, t)|\}, \quad (11)$$

where $k(x, t)$ is a kernel function, and for the Gaussian kernel, $C_k = 1$.

Finding the optimal solution of RLS is equivalent to finding the best tradeoff between $A(\lambda)$ and $S(\lambda)$ for given training data. That is, to minimize $S(\lambda) + A(\lambda)$ over $\lambda > 0$. There is a unique solution - a best λ choice in theory (Cucker & Smale 2002). In real applications, we usually find the λ through cross-validation.

Parzen Windows as Approximation to RLS

In this section we show that Parzen Windows can be viewed as an approximation to RLS. In the discussion, we use the ∞-norm for both matrices and vectors. That is, for a matrix A, $\|A\|_\infty = \max_i\{\sum_j |a_{ij}|\}$, and for a vector x, $\|x\|_\infty = \max_i\{x_i\}$.

Also for the sake of simplicity, we use the notation $\|\cdot\|$ to represent the ∞-norm in the following discussion. This norm, however, should not be confused with the norm used in the previous sections.

Lemma 1 *Let $B = K - I$, where K is the kernel matrix $K_{ij} = k(x_i, x_j)$,k is the Gaussian kernel (2), and I is the identity matrix of proper size. If $\lambda > \frac{\|B\|-1}{m}$, we have*

$$(K + \lambda m I)^{-1} = \sum_{i=0}^{\infty}(-1)^i d^{-i-1} B^i,$$

where $d = 1 + \lambda m$.

Proof

$$\begin{aligned}\|d^{-1}B\| &= d^{-1}\|B\| \\ &= \frac{\|B\|}{(1 + \lambda m)} \\ &< \frac{\|B\|}{(1 + \|B\| - 1)} \\ &= 1\end{aligned}$$

Using $B = K - I$, we can write

$$(K + \lambda mI) = dI + B = d(I + d^{-1}B).$$

Then

$$(K + \lambda mI)^{-1} = d^{-1}(I + d^{-1}B)^{-1}. \quad (12)$$

When $\|d^{-1}B\| < 1$, it follows that (the Neumann series)

$$(I + d^{-1}B)^{-1} = \sum_{i=0}^{\infty}(-1)^i d^{-i}B^i.$$

And

$$(K + \lambda mI)^{-1} = \sum_{i=0}^{\infty}(-1)^i d^{-i-1}B^i. \quad \blacksquare$$

Let

$$\hat{c} = d^{-1}y. \quad (13)$$

We arrive at the Parzen Windows function (1)

$$f_p(x) = \sum_i \hat{c}_i k(x_i, x) \quad (14)$$

$$= \frac{1}{d}(\sum_i y_i k(x_i, x)).$$

Thus Lemma 1 states that under appropriate conditions, the Parzen Windows classifier is an approximation to RLS.

Let $c = (K + \lambda mI)^{-1}y$. We now ask: How well does our $\hat{c}$ approximate c? From Lemma 1 we have

$$(K + \lambda mI)^{-1} = d^{-1}(I - d^{-1}B + d^{-2}B^2 + \cdots).$$

Let

$$E = (I - d^{-1}B + d^{-2}B^2 + \cdots). \quad (15)$$

Then

$$c = d^{-1}Ey. \quad (16)$$

We can now establish a upper bound for the approximation error $|c - \hat{c}|$.

Lemma 2 *Let c and $\hat{c}$ as in (16) and (13), respectively. Then*

$$\|c - \hat{c}\| \leq \frac{\|B\|}{d(d - \|B\|)},$$

where $d = 1 + \lambda m$ and $B = K - I$.

Proof From Eqs (13), (15) and (16) we have

$$c - \hat{c} = -d^{-2}By + d^{-3}B^2y - d^{-4}B^3y + \cdots.$$

Then, observing that $\|y\| = 1$ (since $y \in \{-1, +1\}$)

$$\begin{aligned}\|c - \hat{c}\| &= \| - d^{-2}By + d^{-3}B^2y - d^{-4}B^3y + \cdots \| \\ &\leq \|d^{-2}By\| + \|d^{-3}B^2y\| + \|d^{-4}B^3y\| + \cdots \\ &= \|d^{-2}B\| + \|d^{-3}B^2\| + \|d^{-4}B^3\| + \cdots \\ &= \frac{\|B\|}{d^2}(1 + \|d^{-1}B\| + \|d^{-2}B^2\| + \cdots).\end{aligned}$$

Since $\frac{\|B\|}{d} < 1$, it follows that

$$\begin{aligned}&1 + \|d^{-1}B\| + \|d^{-2}B^2\| + \cdots \\ &= 1 + d^{-1}\|B\| + d^{-2}\|B^2\| + \cdots \\ &\leq 1 + d^{-1}\|B\| + d^{-2}\|B\|^2 + \cdots \\ &= \frac{d}{d - \|B\|}.\end{aligned}$$

Therefore,

$$\|c - \hat{c}\| \leq \frac{\|B\|}{d^2}\frac{d}{d - \|B\|} = \frac{\|B\|}{d(d - \|B\|)}. \quad \blacksquare$$

Error Bound for Parzen Windows

We now establish the error bound for the Parzen Windows function f_p in (1). We split the error for f_p into two parts. The first part is the error between f_p and RLS $f_{\lambda,z}$. The second part is the error between $f_{\lambda,z}$ and the true target f_ρ. By combining these two terms we obtain the error bound for Parzen Windows.

Lemma 3 *Let $\lambda > \frac{\|B\|-1}{m}$. Then*

$$\int_X (f_p - f_\rho)^2 d\rho_X \leq 2(D(\lambda) + S(\lambda) + A(\lambda)), \quad (17)$$

where $D(\lambda) = \frac{\|B\|^2}{\lambda^2(\lambda m + 1 - \|B\|)^2}$, $S(\lambda)$ and $A(\lambda))$ are given by (9) and (8).

Proof First we prove: $\int_X (f_p - f_{\lambda,z})^2 d\rho_X \leq D(\lambda)$. That is:

$$\int_X (f_p - f_{\lambda,z})^2 d\rho_X \leq \frac{\|B\|^2}{\lambda^2(\lambda m + 1 - \|B\|)^2}. \quad (18)$$

From (5) and (14), we have:

$$\begin{aligned}\int_X (f_p - f_{\lambda,z})^2 d\rho_X &= \int_X (\sum_{i=1}^m (c_i - \hat{c}_i)k(x_i, .))^2 d\rho_X \\ &\leq \int_X (\sum_{i=1}^m \frac{\|B\|}{d(d - \|B\|)}k(x_i, .))^2 d\rho_X \\ &\leq \max_x |\sum_{i=1}^m \frac{\|B\|}{d(d - \|B\|)}k(x_i, .)|^2 \\ &\leq (\frac{\|B\|}{d(d - \|B\|)})^2 m^2 \\ &\leq \frac{\|B\|^2}{\lambda^2(\lambda m + 1 - \|B\|)^2}.\end{aligned}$$

Thus

$$\begin{aligned}\int_X (f_p - f_\rho)^2 d\rho_X &= \int_X (f_p - f_{\lambda,z} + f_{\lambda,z} - f_\rho)^2 \\ &= \int_X (f_p - f_{\lambda,z})^2 + \int_X (f_{\lambda,z} - f_\rho)^2 \\ &\quad + 2\int_X (f_p - f_{\lambda,z})(f_{\lambda,z} - f_\rho) \\ &\leq 2\int_X (f_p - f_{\lambda,z})^2 + 2\int_X (f_{\lambda,z} - f_\rho)^2 \\ &\leq 2(D(\lambda) + S(\lambda) + A(\lambda)). \quad \blacksquare\end{aligned}$$

Discussion

Now let us take a look at the error bound. The error bound is a function of λ, training data size m, and indirectly the kernel function k (affecting $v^*(m, \delta)$). M is a positive constant that is chosen to satisfy $|f(x) - y| \leq M$. C_k is determined by the kernel function. For the Gaussian kernel it is 1, as we showed before (Eq. (11)). Here we consider that we are given a fixed training data size m and the kernel function k, then the error bound is a function of λ only

.We will minimize the RLS error bound $S(\lambda) + A(\lambda)$ for $\lambda > 0$. To be the minimum of $S(\lambda) + A(\lambda)$, it is necessary that $-S'(\lambda) = A'(\lambda)$. Taking derivatives, we obtain

$$\begin{aligned} A(\lambda) &= \lambda^{1/2}\|L_k^{-\frac{1}{4}} f_\rho\|^2, \\ A'(\lambda) &= \frac{1}{2}\lambda^{-1/2}\|L_k^{-\frac{1}{4}}, f_\rho\|^2, \\ A''(\lambda) &= -\frac{1}{4}\lambda^{-3/2}\|L_k^{-\frac{1}{4}} f_\rho\|^2. \end{aligned}$$

Similarly,

$$\begin{aligned} S(\lambda) &= \frac{32M^2(\lambda + C_k)^2}{\lambda^2} v^*(m,\delta), \\ -S'(\lambda) &= 64M^2(\frac{C_k}{\lambda^2} + \frac{C_k^2}{\lambda^3}) v^*(m,\delta), \\ -S''(\lambda) &= -64M^2(\frac{2C_k}{\lambda^3} + \frac{3C_k^2}{\lambda^4}) v^*(m,\delta). \end{aligned}$$

Since $\|L_k^{-\frac{1}{4}} f_\rho\|^2 > 0$, $v^*(m,\delta) > 0$, both $A(\lambda)$ and $S(\lambda)$ are positive functions. $A(\lambda)$ strictly increases in $(0, +\infty)$, while $S(\lambda)$ strictly decreases in $(0, +\infty)$ and converges to a positive constant $32M^2 v^*(m,\delta)$. $A'(\lambda)$ is a positive function strictly decreasing in $(0, +\infty)$. $-S'(\lambda)$ is a positive function monotonically decreasing in $(0, +\infty)$. The question is: is there a unique $\lambda^* > 0$ such that

$$-S'(\lambda^*) = A'(\lambda^*)?$$

Let right hand side be: $R(\lambda) = A'(\lambda)$, and the left hand side be: $L(\lambda) = -S'(\lambda)$. Both $L(\lambda)$ and $R(\lambda)$ are monotonically decreasing functions. We consider $\lambda R(\lambda)$ and $\lambda L(\lambda)$: $\lambda R(\lambda)$ is a monotonically increasing positive function in $(0, +\infty)$ and $\lambda R(\lambda) \to 0^+$ when $\lambda \to 0^+$. However $\lambda L(\lambda)$ is still monotonically decreasing in $(0, +\infty)$, and when $\lambda \to 0^+$, $\lambda L(\lambda) \to +\infty$. Then there must be a unique solution $\lambda^* > 0$ such that $\lambda^* L(\lambda^*) = \lambda^* R(\lambda^*)$. It is easy to see that if $L(\lambda) = R(\lambda)$ has more than one distinct solutions in $(0, +\infty)$, then so does $\lambda L(\lambda) = \lambda R(\lambda)$. That contradicts the fact for $\lambda L(\lambda) = \lambda R(\lambda)$. So $L(\lambda) = R(\lambda)$ must have a unique solution. That is, there is a unique λ^* in $(0, +\infty)$ such that $A'(\lambda^*) = -S'(\lambda^*)$. Figure 1 shows the error bound curves.

Now let us take a look at $D(\lambda)$, the first term of the upper bound for the Parzen Windows classifier. We take derivatives for $D(\lambda)$:

$$\begin{aligned} D(\lambda) &= \frac{\|B\|^2}{\lambda^2(\lambda m + 1 - \|B\|)^2} \\ -D'(\lambda) &= \frac{2\|B\|^2}{\lambda^3(\lambda m + 1 - \|B\|)^2} + \frac{2\|B\|^2 m}{\lambda^2(\lambda m + 1 - \|B\|)^3} \\ -D''(\lambda) &= -\frac{6\|B\|^2}{\lambda^4(\lambda m - 1 + \|B\|)^2} - \frac{8\|B\|^2 m}{\lambda^3(\lambda m + 1 - \|B\|)^3} \\ &\quad - \frac{6\|B\|^2 m^2}{\lambda^2(\lambda m + 1 - \|B\|)^4} \end{aligned}$$

Notice that λ can not be chosen arbitrarily in $(0, +\infty)$ in the upper bound (17). Instead, it is only in the range

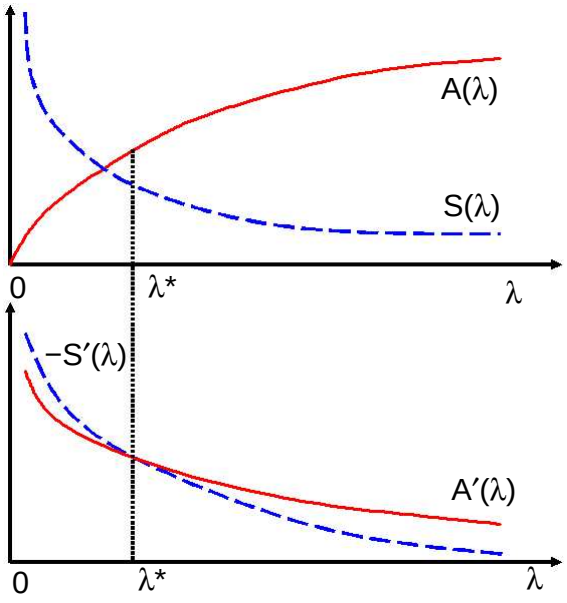

Figure 1: RLS: balance between sample error and approximation error.

$(\max\{\frac{\|B\|-1}{m}, 0\}, +\infty)$. $D(\lambda)$ is a positive function strictly decreasing in $(\max\{\frac{\|B\|-1}{m}, 0\}, +\infty)$. $-D'(\lambda)$ is positive function decreasing in $(\max\{\frac{\|B\|-1}{m}, 0\}, +\infty)$. We will consider: is there a unique $\lambda^\#$ in $(\max\{\frac{\|B\|-1}{m}, 0\}, +\infty)$ such that

$$-2D'(\lambda) - 2S'(\lambda) = 2A'(\lambda)$$

at $\lambda^\#$?

In this case, the right hand side is: $R(\lambda) = 2A'(\lambda)$, and the left hand side be: $L(\lambda) = -2D'(\lambda) - 2S'(\lambda)$. Note that now $\lambda L(\lambda)$ is defined and monotonically decreasing in $(\max\{\frac{\|B\|-1}{m}, 0\}, +\infty)$, and when $\lambda \to \max\{\frac{\|B\|-1}{m}, 0\}$, $\lambda L(\lambda) \to +\infty$. The proof are simply the same as for RLS, and there is a unique $\lambda^\#$ in $(\max\{\frac{\|B\|-1}{m}, 0\}, +\infty)$ such that $-2D'(\lambda) - 2S'(\lambda) = 2A'(\lambda)$. Figure 2 shows the error bound curves.

Notice that λ actually is not a parameter of Parzen Windows. When $\lambda > \frac{\|B\|-1}{m}$, Parzen Windows is an approximation to RLS and its error bound can be derived from that for RLS. But the error for Parzen Windows does not depend on λ. We thus establish the following:

Theorem 4 *Parzen Window's error bound is:*

$$\int_X (f_p - f_\rho)^2 d\rho_X \le 2(D(\lambda^\#) + S(\lambda^\#) + A(\lambda^\#)), \quad (19)$$

where $D(\lambda) = \frac{\|B\|^2}{\lambda^2(\lambda m + 1 - \|B\|)^2}$, $S(\lambda) = \frac{32M^2(\lambda + C_k)^2}{\lambda^2} v^*(m,\delta)$ *and* $A(\lambda) = \lambda^{1/2}\|L_k^{-\frac{1}{4}} f_\rho\|^2$ *; and* $\lambda^\#$ *is the unique solution of* $A'(\lambda) = -S'(\lambda) - D'(\lambda)$.

Compared to RLS, the minimal point of the error bound for Parzen Windows is pushed toward right, i.e., the crossing between $2A'(\lambda)$ and $-2D'(\lambda) - 2S'(\lambda)$ is shifted to the

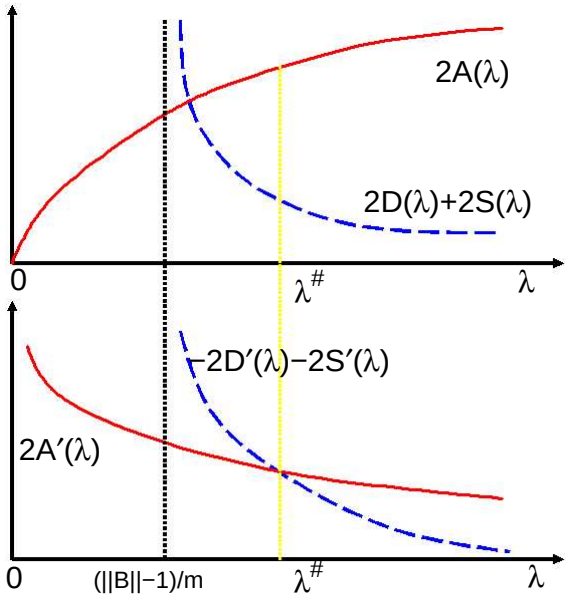

Figure 2: Parzen Windows error bound

right, as shown in Figure 2. The error bound for Parzen Windows is twice that of RLS's plus $2D(\lambda)$. Besides, optimal performance may require λ^* to be less than $\frac{(||B||-1)}{m}$, the barrier that the Parzen Windows can not cross. As a consequence, the performance of Parzen Windows will be much worse than RLS in such situations.

Experiments

We now examine the performance of Parzen Windows, RLS, and SVMs on a number of UCI data sets and an image data set. We also include the k-nearest neighbor method (k=3) in the experiments, since as we discussed before, the Parzen Windows classifier can be viewed a generalization of the nearest neighbor classifier. For simplicity, we choose only binary classification problems. For data sets that include multiple classes, we transfer them into a binary classification problem by combining classes (e.g. glass data) or by choosing only two classes from the data sets (for example, we choose only 'u' and 'w' from the Letter data). All data has been normalized to have zero mean and unit variance along each variable.

We randomly select 60% of the data as training data to build classifiers, and the rest (40% of data) as test data. We repeat this process 10 times to obtain average error rates. For the Parzen Windows classifier, there is only one parameter to tune: σ of the Gaussian kernel. For RLS, there are two parameters to tune: σ and λ. For SVMs, there are also two parameters to tune: σ and C (which controls the softness of the margin (Cristianini & Shawe-Taylor 2000)). These parameters are chosen by ten-fold cross-validation. The classification results are shown in Table

On some data sets (e.g. heart-h and iris), the performance of Parzen Windows is close to RLS. For those data sets, it turns out that RLS tends to favor large λ values, or when using larger λ values, its performance does not sacrifice much. However, on the ionosphere data set, the performance of Parzen Windows is significantly worse than RLS. For this data set, RLS prefers small λ value (0.000 in the table means that average λ values picked by RLS is less than 0.001) and its performance becomes poor quickly when larger λ values are used. Remember that, Parzen Windows, as an approximation to RLS, only approximates RLS well when the value of λ is large.

So why is ionosphere so different? This is because, the ionosphere data has more irrelevant dimensions. This is also noticed and demonstrated by many others. For example in (Fukumizu, Bach, & Jordan 2004), it is shown that when the data (originally 33 dimensions) are carefully projected onto a subspace with roughly 10 dimensions, SVMs can perform significantly better.

We have also evaluated these classifiers on an image data set composed of two hundred images of cat and dog faces. Each image is a black-and-white 64×64 pixel image, and the images have been registered by aligning the eyes.

For the image data, it is more obvious that most dimensions are noise or irrelevant. It is well known that for the image data, only 5% dimensions contain critical discriminant information. So for the cat and dog data, RLS prefers extremely small λ values. Parzen Windows cannot approximate RLS well in such cases, and its performance is significantly worse. The results are listed in the last row of Table 1.

Then why does RLS choose small λ values in those data sets? Remember that λ is the regularization parameter. When choosing a larger λ, RLS selects smoother functions. A smoother function (function with a smaller norm) can learn quite well for simple problems. However, when there are many noisy and irrelevant dimensions, RLS is forced to use more complicated functions in order to approximate the target function. This implies that the Parzen Windows classifier lacks the capability of choosing complex functions to fit the data.

A closer look at the results reveals that the performance of the 3-NN classifier is strongly related to Parzen Windows. If Parzen Windows performs badly, so does 3-NN. For the ionosphere data, the Parzen Windows classifier registered 15% error rate, while 3-NN registered 17%, comparing to 6% error rates of both RLS and SVMs. For cats and dogs, the Parzen Windows classifier's error rate is 36%, and 3-NN 41%, while RLS achieved 12% and SVM obtained 8%. These also imply that the nearest neighbor classifier using simple Euclidean distance may not learn well in some situations where more complex functions are required, given finite samples.

It would be very interesting to compute the values of the error bounds for both Parzen Windows and RLS, and to compare how different they are for each data set. Unfortunately, the calculation of the error bounds involves the 'ground-truth' function f_ρ which is unknown, for the data sets we used in our experiments. One possible way is to use simulated data sets with pre-selected target functions and then compute the error bounds. This will be our future work.

	3nn		Parzen		RLS		SVM		ave (λ)	$ave(\|\|B\|\|-1)/m$
	$\bar{\epsilon}$	$\sigma(\epsilon)$	$\bar{\epsilon}$	$\sigma(\epsilon)$	$\bar{\epsilon}$	$\sigma(\epsilon)$	$\bar{\epsilon}$	$\sigma(\epsilon)$		
sonar	0.226	0.033	0.210	0.021	0.152	0.031	0.135	0.043	-	-
glass	0.088	0.028	0.067	0.018	0.058	0.019	0.062	0.022	0.005	0.457
creditcard	0.150	0.017	0.159	0.021	0.123	0.017	0.130	0.016	0.105	0.526
heart-c	0.188	0.024	0.187	0.023	0.180	0.022	0.180	0.020	0.023	0.542
heart-h	0.235	0.053	0.194	0.028	0.212	0.039	0.214	0.029	2.045	0.456
iris	0.068	0.027	0.080	0.028	0.083	0.041	0.048	0.018	8.010	0.101
ionosphere	0.170	0.019	0.150	0.022	0.063	0.018	0.061	0.016	0.000	0.518
thyoid	0.052	0.023	0.048	0.024	0.029	0.013	0.028	0.020	2.00	0.546
letter(u,w)	0.006	0.002	0.005	0.001	0.002	0.002	0.002	0.001	6.00	0.084
pima	0.274	0.018	0.265	0.019	0.240	0.016	0.243	0.023	0.006	0.656
cancer-w	0.027	0.007	0.037	0.009	0.027	0.007	0.033	0.007	0.005	0.263
cancer	0.321	0.030	0.271	0.010	0.276	0.020	0.286	0.019	-	-
catdog	0.411	-	0.362	-	0.118	-	0.084	-	0.000	0.464

Table 1: Error rate comparison of 3-NN, Parzen, RLS and SVM on UCI 12 datasets, and Cat & Dog image data. Average λ picked by RLS and average $(||B|| - 1)/m$ are also listed here. In the table $\bar{\epsilon}$ denotes the average error rate, and $\sigma(\epsilon)$ the standard deviation of the error rate.

Summary

In this paper, we have shown that Parzen Windows can be viewed as an approximation to RLS under appropriate conditions. We have also established the error bound for the Parzen Windows method based on the error bound for RLS, given finite samples. Our analysis shows that the Parzen Windows classifier has higher error bound than RLS in finite samples. More precisely, RLS has an error bound of $A(\lambda) + S(\lambda)$, while Parzen Windows has an error bound of $2D(\lambda) + 2A(\lambda) + 2S(\lambda)$.

It may be argued that the error bound for Parzen Windows is not tight. However, we have shown in this paper that Parzen Windows, as a special case of RLS, lacks the flexibility to produce complex functions, while RLS does with different choices of λ.

We have discussed the conditions under which when Parzen Windows is a good approximation to RLS and the conditions under which it is not. Our experiments demonstrate that on most UCI benchmark data sets, the Parzen Windows classifier has similar performance to RLS, which means that on these data sets, the Parzen Windows classifier is relatively a good approximation to RLS. However, on some data sets, especially the ionosphere and cat and dog data, the Parzen Windows classifier does not approximate RLS well, and thus performs significantly worse than RLS. Those data sets usually contain many noisy and irrelevant dimensions.

Our analysis also brings insight into the performance of the NN classifier. Our experiments also demonstrate that the NN classifier has similar performance to that of Parzen Windows. And the results imply that both the Parzen Windows and NN classifiers can approximate smooth target functions, but fail to approximate more complex target functions.

References

Cristianini, N., and Shawe-Taylor, J. 2000. *An Introduction to Support Vector Machines and other kernel-based learning methods*. Cambridge, UK: Cambridge University Press.

Cucker, F., and Smale, S. 2001. On the mathematical foundations of learning. *Bulletin of the American Mathematical Society* 39(1):1–49.

Cucker, F., and Smale, S. 2002. Best choices for regularization parameters in learning theory: On the bias-variance problem. *Foundations Comput. Math.* (4):413–428.

Duda, R.; Hart, P.; and Stork, D. 2000. *Patten Classification, 2nd edition*. John-Wiley.

Evgeniou, T.; Pontil, M.; and Poggio, T. 2000. Regularization networks and support vector machines. *Advances in Computational Mathematics* 13(1):1–50.

Fukumizu, K.; Bach, F. R.; and Jordan, M. I. 2004. Dimensionality reduction for supervised learning with reproducing kernel hilbert spaces. *Journal of Machine Learning Research* 5:73–99.

Fukunaga, K. 1990. *Introduction to statistical pattern recognition*. Academic Press.

Hoerl, A., and Kennard, R. 1970. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics* 12(3):55–67.

J. Hertz, A. K., and Palmer, R. 1991. *Introduction to the Theory of Neural Computation*. Addison Wesley.

Mitchell, T. 1997. *Machine learning*. McGraw Hill.

Parzen, E. 1962. On the estimation of a probability density function and the mode. *Ann. Math. Stats.* 33:1049–1051.

Poggio, T., and Smale, S. 2003. The mathematics of learning: Dealing with data. *Notices of the American Mathematical Society* 50(5):537–544.

Tikhonov, A. N., and Arsenin, V. Y. 1977. *Solutions of Ill-posed problems*. Washington D.C.: John Wiley and Sons.

Vapnik, V. 1998. *Statistical Learning Theory*. New York: Wiley.