IDIAP RESEARCH REPORT

The Analysis of Kernel Ridge Regression Learning Algorithm

Alexei Pozdnoukhov [a]

IDIAP–RR 02-54

November 2002

Dalle Molle Institute for Perceptual Artificial Intelligence • P.O.Box 592 • Martigny • Valais • Switzerland

phone +41 – 27 – 721 77 11
fax +41 – 27 – 721 77 12
e-mail secretariat@idiap.ch
internet http://www.idiap.ch

[a] IDIAP, P.O. Box 592, CH-1920 Martigny, Switzerland

IDIAP Research Report 02-54

The Analysis of Kernel Ridge Regression Learning Algorithm

Alexei Pozdnoukhov

November 2002

Abstract. The paper presents Kernel Ridge Regression (KRR), a nonlinear extension of the well known statistical model of ridge regression. New insights on the method are also presented. In particular, the connection between ridge regression and local translation-invariant squared loss minimization algorithm is shown. An iterative training algorithm is proposed, that allows training the KRR for large datasets. The training time is empirically found to scale quadratically with the number of samples. The application of the model is illustrated on the real datasets.

1 Introduction

Let us state the general problem of regression estimation as presented in the scope of Statistical Leaning Theory [?]. Suppose we are given a set of observations generated from an unknown probability distribution $P(x, y)$ $(x_1, y_1), ..., (x_L, y_L)$ with $x_i \in R^N, y_i \in R$ and a class of functions $F = \{f : R^N \to R\}$. Regression estimation is the problem of estimating the conditional expectation

$$r(x) = \int y dP(y \mid x) \tag{1}$$

in the defined set of functions given the data. In Machine Learning this problem is reduced to the problem of minimizing the risk functional based on empirical data:

$$\min_{\Lambda} R(\alpha) = \int (y - f(x, \alpha))^2 dP(y, x) \tag{2}$$

where we assume that the class $F = \{f(x, \alpha) : R^N \to R\}$ is given by the admissible set of its hyper-parameters $\alpha \in \Lambda$. Under certain conditions risk minimization in a set of functions gives r(x) if $r(x) \in F$ and provides the solution closest to $r(x)$ in a $L_2(P)$ metric if $r(x) \notin F$.

Note that in (**??**) we restrict ourself to squared loss functions. Under the assumption that the data is corrupted with normal noise the minimization of risk with squared loss lead to the best unbiased estimation. But if it is only known that noise distribution is symmetric, a linear loss is preferable and results in a model from the robust regression family [?].

In the following we present the Ridge Regression model, firstly derived in statistics [?], and give an extension to non-linear regression using the kernel trick in the dual variables [?]. We then present some links between Ridge Regression and other learning algorithms, which give different interpretations to Ridge Regression. We also present an iterative algorithm to train KRR, describe its empirical properties and show some experiments and comparisons on real data.

2 Ridge Regression as a ML Algorithm

2.1 Ridge Regression in Dual Variables

We consider the class F of linear functions $F = \{w \cdot x + b \mid w, b \in R\}$ and instead of minimizing the true risk R we minimize a "regularized" empirical risk. So the task is to minimize the following Ridge Regression functional:

$$\min \sum_{i=1}^{L} (y_i - w \cdot x_i - b)^2 + \gamma ||w||^2 \tag{3}$$

where γ is a regularization coefficient. Following the standard technique we reformulate this problem as

$$\min \sum_{i=1}^{L} \xi_i{}^2 + \gamma ||w||^2 \tag{4}$$

under the constraints

$$y_i - w \cdot x_i - b = \xi_i, \qquad i = 1, ..., L. \tag{5}$$

This constrained problem can be solved by finding the saddle point of the Lagrangian

$$L(w, b, \alpha) = \sum_{i=1}^{L} \xi_i^2 + \sum_{i=1}^{L} \alpha_i (y_i - w \cdot x_i - b - \xi_i)^2 + \gamma ||w||^2 \tag{6}$$

where we introduced the Lagrange multipliers $\alpha_i, i = 1, ..., L$. Differentiating (**??**) in w and b we obtain:

$$w = \frac{1}{2\gamma}\sum_{i=1}^{L}\alpha_i x_i, \tag{7}$$

and

$$\sum_{i=1}^{L}\alpha_i = 0 \tag{8}$$

Substituting (**??**) into (**??**) and differentiating in ξ_i we obtain $\xi_i = \frac{\alpha_i}{2}$. The final optimization problem in a dual form becomes

$$\max D(\alpha) = \sum_{i=1}^{L} y_i\alpha_i - \frac{1}{4}\sum_{i=1}^{L}\alpha_i^2 - \frac{1}{4\gamma}\sum_{i,j=1}^{L}\alpha_i\alpha_j(x_i, x_j) \tag{9}$$

with the equality constraint (**??**).

Using (**??**) the regression function can be expressed as a linear combination of dot products between the test sample and training samples:

$$f(x) = \frac{1}{2\gamma}\sum_{i=1}^{L}\alpha_i(x_i, x) + b. \tag{10}$$

Hence Lagrange multipliers can be roughly interpreted as a measure of importance of the correponding sample for the solution.

2.2 Kernel Ridge Regression

Let us now introduce the kernel trick. Note that both optimization problem (**??**) and regression function (**??**) depend not on the samples itself but on the dot products between samples.

Theorem(Mercer). Consider a continiuous symmetric function $K(x, x') : X^2 \to R$ where we denoted an input space as X. If for any $g \in L_2(C)$, C being the compact subset of X,

$$\int_C\int_C K(x, x')g(x)g(x')dxdx' \geq 0 \tag{11}$$

then it can be expanded in a absolutely and uniformly converging series

$$K(x, x') = \sum_{k=1}^{\inf} a_k\psi_k(x)\psi_k(x') \tag{12}$$

where $\psi_k(.)$ and $a_k \geq 0$ is the eigensystem of the corresponding integral operator.

So for every function $K(x, x')$ satisfying the conditions of the theorem there exist a feature space where it acts as a dot product. Mercer theorem gives one way of obtaining a dot product from kernel function. Note that the exact mapping from input space to the feature space is undefined, but we can be sure that this space exists. If one wants to use a definite feature space and can provide the mapping to it then the kernel function would be just a dot product in this feature space.

Substituting the dot products into (**??**) and (**??**)with a proper kernel function K we obtain the following optimization problem:

$$\max D(\alpha) = \sum_{i=1}^{L} y_i\alpha_i - \frac{1}{4}\sum_{i=1}^{L}\alpha_i^2 - \frac{1}{4\gamma}\sum_{i,j=1}^{L}\alpha_i\alpha_j K(x_i, x_j) \tag{13}$$

under the constraint

$$\sum_{i=1}^{L} \alpha_i = 0 \tag{14}$$

and the regression function becomes

$$f(x) = \sum_{i=1}^{L} \alpha_i K(x_i, x) + b. \tag{15}$$

Note that the optimization problem has the equality constraint (**??**). It appeared since we used a regression function $f(x) = w \cdot x + b$ with a constant threshold b. If we had used it without a threshold, $f(x) = w \cdot x$, we would have obtained the same optimization problem but without the constraint. There exist a closed form expression for the weights α minimizing (**??**) without constraint (**??**) which involves a LxLmatrix inversion hence can be inpractical for the large real-world datasets. Instead we propose to solve directly a quadratic constrained (QP) problem (**??**), (**??**) with a fast iterating algorithm of the SMO kind [**?**]. But before we move to the algorithm description, let's notice how the Ridge Regression can be derived alternatively directly from the risk minimization. This is the approach that follows from Vicinal Risk Minimization principle - a novel learning principle proposed by Vapnik [**?**].

2.3 Ridge Regression and Translation Invariance.

Consider the initial setting: minimization of the risk with squared loss (**??**). Let's notice that empirical risk minimization (ERM) principle can be derived from (**??**) assuming the following empirical distribution $P(x, y)$:

$$P(x, y) = \frac{1}{L} \sum_{i=1}^{L} \delta_{x_i}(x) \delta_{y_i}(y) \tag{16}$$

Not let us consider the minimization of risk in a class of linear functions $f(x) = w \cdot x + b$ and the Gaussian spherical Parzen window density estimate $N_\sigma(x - x_i) = N_\sigma(\delta x_i)$. Substituting into (**??**) gives

$$\begin{aligned} R_{vic}(f) &= \frac{1}{L} \sum_{i=1}^{L} \int (f(x) - y_i)^2 dN_\sigma(\delta x_i) \\ &= \frac{1}{L} \sum_{i=1}^{L} \int (f(x_i) - y_i + w \cdot \delta x_i)^2 dN_\sigma(\delta x_i) \\ &= \frac{1}{L} \sum_{i=1}^{L} (f(x) - y_i)^2 + \frac{1}{n} \sum_{i=1}^{L} \int (w \cdot \delta x_i)^2 dN_\sigma(\delta x_i) \\ &= \frac{1}{L} \sum_{i=1}^{L} (f(x) - y_i)^2 + \sigma^2 \|w\|^2 \end{aligned} \tag{17}$$

which coincides with (**??**).

The assumption on the probability distribution $N_\sigma(x - x_i)\delta_{y_i}(y)$ can be thought as adding artificial data distributed according to the given law. This is sometimes referred to as noise injection. Adding artificial training samples is also a way to include an a priory invariance information. It is proven to be equivalent to adding a regularization term ot the empirical risk under some conditions [**?**].

In particular, one can look for the translation-invariant algorithm minimizing empirical risk. As shown in [?], to get an unbiased estimator invariant to local transformation $g(x,\alpha), \alpha \to 0$; and $g(x,0) : x \to x$, one has to minimize

$$R_{inv}(f) = R_{emp}(f) + \sum_{i=1} \sigma_i^2 \int p(x)dx \Big(\frac{\partial g^\mu}{\partial \alpha_i}\frac{\partial f(x,w)}{\partial x^\mu}\Big)^2 \tag{18}$$

where derivative by α is taken at $\alpha = 0$. Now as we are interested in translation-invariance, hence $g(x,\alpha) = x + \alpha$, and (??) is simplified to

$$\begin{aligned} R_{inv}(f) &= R_{emp}(f) + \sigma^2 \int p(x)dx |\nabla_x f(x,w)|^2 \\ &= R_{emp}(f) + \sigma^2 \|w\|^2 \end{aligned} \tag{19}$$

which again coincides with (??).

So far, the conclusions one can make from the analysis of the Ridge Regression algorithm in a class of linear functions are as follows. Regularization with a squared norm of the coefficient vector is equivalent to minimization of risk with Gaussian Parsen windows density estimate and to the learning algorithm minimizing the squared loss and invariant to local translations of the input data. The observed properties gives a way to select the regularization parameter of the algorithm if one is given the accuracy of the inputs or given the scale of translation invariance if it exists. Therefore, we can expect that linear Ridge Regression algorithm should do particularly well for translation-invariant data. For the Kernel Ridge Regression we can only expect that it does well for the data that is translation-invariant in a feature space.

3 Training

To solve the optimization problem (??),(??) we propose a simple iterative algorithm of the SMO-kind. Two weights are optimized at each step of the algorithm, and the solution always satisfies the constraint (??). The algorithm updates the weights of the training samples that provide maximal residuals. The weights are updated according to

$$\alpha_1^{new} = \frac{1}{2\eta}(y_2 - y_1 - 2\eta(K_{22} - K_{12}) + f^{old}(x_1) - f^{old}(x_2))$$

$$\alpha_2^{new} = \chi - \alpha_2^{old},$$

where

$$\eta = 2K_{12} - K_{11} - K_{22}$$
$$\chi = \alpha_1^{old} + \alpha_2^{old},$$

where K_{11}, K_{22}, K_{12} are the values of kernel function for the optimized variables, and $f^{old}(x)$ is a regression function at point x calculated with old (not optimized) values of weights α_1^{old}, α_2^{old}. Initial values of α_i are zero.

Since the solution is not sparse - the coefficients $\alpha_i \neq 0$, and not bounded, shrinking or active set selection methods are not fully applicable for this case. A pair of active variables for updating is determined by maximal residuals, i.e. the pair is selected if it is worst fitted by the current regression function. The training residuals are saved in the residual cache and updated at each iteration. To speed up the algorithm, kernel functions are calculated and saved in a kernel cache. The algorithm is implemented as a part of the Torch3 machine learning library (www.torch.ch), [?].

Note that the model without a threshold, $f(x) = w \cdot x$, leads to an unconstrained optimization problem and can be solved by any gradient method.

	Training time, sec.		
Training set size	$\sigma = 50, \gamma = 0.01$	$\sigma = 50, \gamma = 0.1$	$\sigma = 200, \gamma = 0.1$
500	0.32	0.11	0.11
1000	1.46	0.38	0.42
2000	6.36	1.54	1.64
4000	26.3	6.15	6.34
8000	106.2	24.4	24.96

Table 1: The dependence of the training time on the training set size. Kernel cache size is 250Mb.

Cache size, Mb	1	50	100	150	200	250	300
Cached samples	32	1638	3276	4915	6553	>8000	>8000
Training time, sec	506.7	439.0	368.3	265.3	130.4	33.9	33.9

Table 2: The dependence of the training time on the cache size. Training set size is 8000 samples.

4 Experiments

The experiments below were carried out on two datasets:

Sunspots. The original dataset is a time series representing the number of visible sunspots a day. It was converted into regression task: to predict the yearly average of the year starting next day using the previous 12 yearly averages. The training dataset consists of 40000 samples, while 2500 other samples are used for testing.

Boston. The task is to predict the median price of the houses in certain area of Boston based on 12 continiuous and 1 binary variables defining the characteristics of the area. The training dataset consists of 466 samples, while 40 samples were reserved for testing.

The isotropic Gaussian Radial Basis function $K(x, y) = e^{\frac{-\|x-y\|^2}{\sigma^2}}$ was used as kernel. This kernel has only one parameter - the kernel bandwidth σ, hence only two parameters have to be tuned to apply the algorithm: kernel bandwidth σ and the regularization parameter γ.

We now present some of the observed empirical characteristics of the algorithm such as the dependence of the training time on the cache size and the number of training samples. Table 1 gives the dependence of the training time on the number of training samples. The cache size is large enough to keep all the kernel values in memory. One can notice that the training time scales quadratically with the number of samples. When it is not possible to store the kernel matrix in memory, the scaling is observed to be upper quadratic. Table 2 presents the observed dependence of the training time on the cache size.

The closed form expression for the regression coefficients is as follows:

$$\alpha = (K^T K + \gamma I)^{-1} K^T Y \tag{20}$$

Training set size	500	1000	1500	2000
Iterative algorithm, sec	0.09	0.41	1.62	6.5
Standard training, sec	0.9	7.8	62	544

Table 3: The dependence of the training time on the training set size for iterative and standard algorithm.

	Parameters	Training RMSE	Testing RMSE
Boston housing	KRR $\gamma = 10^{-3}$, $\sigma = 150$	2.72	3.26
	SVR $\epsilon = 2, C = 10^3, \sigma = 80$	4.01	5.26
Sunspots	KRR $\gamma = 0.4$, $\sigma = 1200$	11	17.3
	SVR $\epsilon = 20, C = 10^3, \sigma = 900$	10.83	15.8

Table 4: Experimental Results on the Datasets.

where K is a kernel matrix, and Y is a training outputs vector. So ordinary way to train KRR is to use this formula with LxL matrix inversion. Table 3 presents the comparison of the training times for standard training with (**??**) and the proposed iterative algorithm. The results coincide with theoretical estimates on the training time; LU-decomposition is used for matrix inversion, and training requires an order of $O(N^3)$ operations.

The problem arises when the regularization parameter is close to zero. The matrix becomes badly conditioned and its inversion is not stable, leading to improper results. It was found that the iterative algorithm converges slower for small values of the regularization parameter, but still gives correct results while matrix inversion fails. Moreover, the iterative algorithm does not require storage of kernel and derived matrices. The convergence of the iterative algorithm can be proved analogously to the proof for SMO [**?**]. Nevetheless, it was found that in practice for some values of parameters it may not converge, mainly due to the roundoff errors.

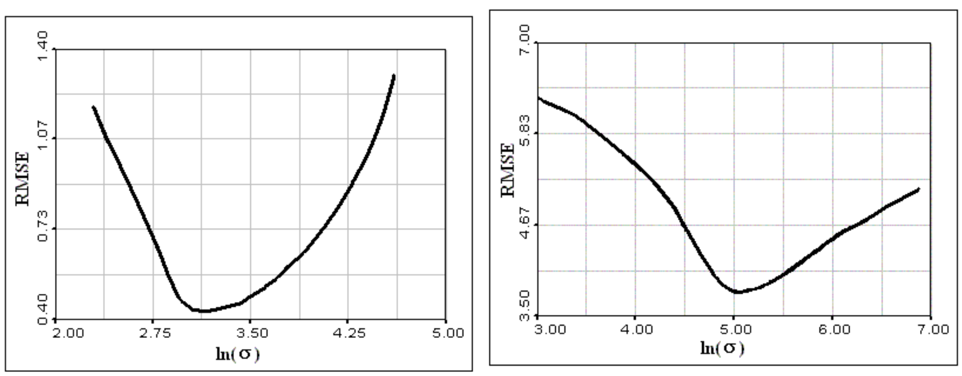

Figure 1: 5-fold cross-validation error curves: Sunspots dataset(left) and Boston housing(right).

A related machine learning regression algorithm is the Support Vector Regression [**?**], that minimizes the regularized ϵ-insensitive linear loss in a feature space. The solution is sparse for non-zero values of ϵ, providing good generalization abilities. Moreover, the sparsity and the finite upper bound for the weights are used to speed up the training process, since enable selection of active subset for optimization while other weights are fixed. It reduces the effective size of the optimization problem and gives the possibility to train sparse algorithms like SVR for larger datasets than non-sparse ones. Nevertheless, KRR has only one parameter to tune besides kernel parameters: the regularization parameter γ, while SVR has two parameters besides kernel function: the width of insensitive region ϵ, and the trade-off constant C. Let's also notice that SVR with quadratic loss, $\epsilon = 0$ and $C = \infty$ is equivalent to the KRR.

Table 4 presents the results obtained on the datasets. The general conclusion one can make is that the KRR is quite a competitive algorithm. It does worse than SVR on sunspots data, while outperforming it on the boston housing data. The parameters for the boston housing were selected according to the minima of 5-fold cross-validation error. The error curves are shown in figure 1.

The parameters for the sunspots dataset were selected according to the minima of validation error calculated on the subset of the training data. The cross-validation error for this dataset (figure 1, left) does not give the correct values of the parameters since data are not i.i.d. The reason for that was the specificy of the data, since it is in fact a time series prediction task.

5 Conclusions

The paper presented a nonlinear extension of the well known statistical model of ridge regression. Two new insights on the method were also presented. In particular, it was found that linear ridge regression is equivalent to minimization of squared loss on training data assuming them invariant to local translations. An iterative training algorithm was proposed, that allows using the KRR for large datasets hence makes it applicable for real-life regression problems. The training time was empirically found to scale quadratically with the number of samples. The application of the model was illustrated on real datasets.

6 Acknowledgements

The author want to thank the Swiss National Science Foundation for supporting this work through the National Centre of Competence in Research (NCCR) on "Interactive Multimodal Information Management (IM2)". This work was also partly supported by the European project "LAVA: Learning for Adaptable Visual Assistants".

References

[1] P. Huber. Robust Estimation of location parameter. Annals of Mathematical Statistics, 35 (1), 1964.

[2] A.E. Hoerl and R.W. Kennard. Ridge Regression: Biased estimation for non-orthogonal problems. Technometrics 12, pp. 55-67, 1970.

[3] C. Saunders, A. Gammerman, and V. Vovk Ridge Regression in dual variables. Technical Report, Royal Holloway University of London, 1998.

[4] J. Platt. Fast training of the Support Vector Machines using sequential minimal optimization. In B.Scholkopf, C.J.C.Burges, and A.J.Smola, editors, Advances in Kernel Methods - Support Vector Learning, pages 185-208, Cambridge, MA, MIT press, 1999.

[5] T.K. Leen. Invariance and regularization in learning. In Advances in Neural Information Processing Systems 7, MIT press, 1995.

[6] R. Collobert and S. Bengio. SVMTorch: Support vector machines for large-scale regression problems. *Journal of Machine Learning Research*, 1:143–160, 2001.

[7] S.S. Keerthi and E.G. Gilbert. Convergence of a generalized SMO algorithm for SVM classifier design. Technical Report CD-00-01, National University of Singapore, 2000.

[8] V. N. Vapnik. The Nature of Statistical Learning Theory. Second Edition, Springer-Verlag, New-York, NY, USA, 2000.