Mach Learn (2006) 65:273–308
DOI 10.1007/s10994-006-9455-4

Training a reciprocal-sigmoid classifier by feature scaling-space

Kar-Ann Toh

Received: 21 December 2004 / Revised: 6 June 2006 / Accepted: 9 June 2006 / Published online: 14 July 2006

Abstract This paper presents a reciprocal-sigmoid model for pattern classification. This proposed classifier can be considered as a Φ-machine since it preserves the theoretical advantage of linear machines where the weight parameters can be estimated in a single step. The model can also be considered as an approximation to logistic regression under the framework of Generalized Linear Models. While inheriting the necessary classification capability from logistic regression, the problems of local minima and tedious recursive search no longer exist in the proposed formulation. To handle possible over-fitting when using high order models, the classifier is trained using multiple samples of uniformly scaled pattern features. Empirically, the classifier is evaluated using a benchmark synthetic data from random sampling runs for initial statistical evidence regarding its classification accuracy and computational efficiency. Additional experiments based on ten runs of 10-fold cross validations on 40 data sets further support the effectiveness of the reciprocal-sigmoid model, where its classification accuracy is seen to be comparable to several top classifiers in the literature. Main reasons for the good performance are attributed to effective use of reciprocal sigmoid for embedding nonlinearities and effective use of bundled feature sets for smoothing the training error hyper-surface.

Keywords Machine learning · Pattern classification · Φ-machine · Polynomials · Parameter estimation

1. Introduction

"The more relevant patterns at your disposal, the better your decisions will be."—Herbert Simon. The importance of pattern classification thus cannot be over-emphasized.

Editor: Risto Miikkulainen

K.-A. Toh (✉)
Biometrics Engineering Research Center, School of Electrical & Electronic Engineering,
Yonsei University, Seoul, Korea
e-mail: katoh@yonsei.ac.kr or katoh@ieee.org

While the statistical approach (see e.g. Duda et al., 2001) has received considerable attention, many estimators, predictors or approximators (see e.g. Poggio & Girosi, 1990; Bishop, 1995; Mitchell, 1997; Schürmann, 1996) can be used for pattern classification. From the model perspective, these estimators can be formulated either in linear or nonlinear form. The advantages of linear formulation include its tractability for optimization, sensitivity analysis, and prediction of confidence intervals. However, it is limited to decision hyperplanes that can be projected onto the prescribed set of linear parameters. The nonlinear formulation, on the other hand, offers a compact mapping capacity but it comes with an increased complexity in obtaining an appropriate solution. A good balance between the mapping capacity and solution complexity for predicting unseen test data becomes an issue for optimal classifier design.

The proposed reciprocal-sigmoid model attempts to embed nonlinearities in a linear formulation without sacrificing much of the capability to classify nonlinear hyper-surfaces. This model can be considered as a Φ-machine since it preserves the theoretical advantages of linear machines while allowing for nonlinear combinations of the inputs (Nilsson, 1965; Duda & Hart, 1973; Precup & Utgoff, 2004). The approximation capability of this model is shown prior to the introduction of *scaling-space* learning technique that aims to provide a stable learning. The concept of scale-space filtering has been well explored in computer vision where some derived image signals are used for extracting signal structures suitable for classification and other applications. Unlike these scale-space filters, our derived signal is not distorted except that it is uniformly scaled. Multiple samples of these scaled signals were used thereby forming a scaled feature space for classifier training, and hence the scheme is called *scaling-space* learning in brief. As we are not able to find a better term, we use "scaling-space" instead of "scale-space" to differentiate between our definition and that in the context of computer vision.

The scaling-space learning is also differentiated from the input scale parameter adaptation as seen in kernel or network parameter adjustment schemes (see e.g. Juszczak et al., 2000; Grandvalet & Canu, 2002) since a single scaling factor is attached to each kernel parameter in these schemes thereby giving rise to vast combinatorial possibilities. These combinatorial possibilities can be both advantageous and disadvantageous. The advantage is that they can cater for diverse applications and the disadvantage is that the search required to find a good solution could be difficult. Moreover, as the formulation for scaling factor search is nonlinear for many kernel basis functions, the issue of local solution complicates the matter and this may give rise to difficulties in arriving at well conclusive observations. A good balance between model complexity and predictivity becomes an issue for practical design.

The proposed feature scaling-space learning technique is consistent with the spirit of Bootstrapping (Efron & Tibshirani, 1993) for stable classifier design, but differs in its training replicates which are not obtained from re-sampling. Unlike that in Bootstrap Aggregating (Bagging; Breiman, 1994), *all* the scaled training replicates are bundled to form a large training set and only a single classifier is trained using this large data set. In essence, multiple virtual classifiers from different scalings are embedded within a single training step to form the final classifier. The computational effort for those aggregated classifiers like bagging, boosting and random subspace methods (see e.g. Breiman, 1996; Skurichina & Duin, 2002; Tax & Duin, 2000) is translated into a process of selecting an appropriate scaling factor in the scaling-space technique.

To summarize, the proposed reciprocal-sigmoid model can be considered as an approximation to logistic regression in Generalized Linear Models (GLM, see e.g. Nelder & Wedderburn, 1972; McCullagh & Nelder, 1989; Hardin & Hilbe, 2001; Gordon, 2002) demonstrating match of link and loss functions for linear estimation. Specifically, the

proposed model embeds the following types of nonlinearities for pattern classification: (i) use of nonlinear reciprocal of sigmoid as the basis function, and (ii) use of bundled feature sets (scaling-space) for training error hyper-surface smoothing. Main contributions of this paper are in the aspects of having: (i) proposed a reciprocal-sigmoid classifier where the number of constructed feature terms varies linearly with the order of the polynomial, instead of having a power law in the case of full multivariate polynomials, (ii) showed the approximation capability of the classifier for discrete data, and (iii) demonstrated that the scaling-space input provides means to improve numerical conditioning. Extensive numerical experiments, which provide benchmarks for such design, are performed to support the proposal. The significance of this work thus lies on the massive empirical evidences pointing to an efficient approximation of logistic regression model for pattern classification.

The paper is organized as follows: the next section provides a brief review on related background developments and our problem treatment. In Section 3, a linear reciprocal-sigmoid model is introduced. By removing some redundant terms from an original formulation, the approximation capability, and hence classification capability of the refined model is shown. With these fundamentals in place, Section 4 tackles the learning problem by a regularized scaling-space learning technique, aiming at a stable solution which does not over-fit the data. Section 5 uses a benchmark synthetic data to illustrate the usefulness of the proposed method on a plug-in polynomial model and the proposed classifier model. The applicability of the method is further supported using 40 benchmark data sets taken from UCI Machine Learning Repository.

2. Background and problem treatment

2.1. Supervised learning and related issues

In pattern classification, there are two major hurdles in good applications of nonlinear estimators, namely (1) model structure selection and, (2) solving for parameters that best fit the data (training), both of which aim to arrive at a classifier that performs well for unseen data. More frequently, these two problems are coupled such that overcoming both of them for good predictivity is a difficult task. In Huang and Babri (1998) and Huang (2003), an ingenious way to efficiently compute the network weights in a single step after certain initialization procedures can be found. However, for pattern classification, solving an exact solution of a nonlinear formulation to pass through every training data point may cause over-fitting which means that it may lose much predictivity for unseen test data. Moreover selection of an appropriate number of layers and number of nodes remains an issue for good predictivity. Much research activities are on-going in the field to resolve this issue.

The Support Vector Machine (SVM) developed by Boser et al. (1992) and Vapnik (1998) provides a mechanism to minimize the training error and compute a bound on the VC-dimension at the same time (see e.g. Osuna et al., 1997; Burges, 1998 for a very comprehensive account). This result is achieved via minimizing a parameter norm subject to a separation functional. For classification problems, the main idea behind the technique is to find a hyperplane that maximizes the separation margin rather than attempting to pass a hyperplane through all data points as in the case of curve fitting. This proposal has led to many useful applications in the literature.

In a recent attempt to achieve good predictivity, Tipping (2000, 2001) proposed the Relevance Vector Machine (RVM) from reduction of model complexity viewpoint. Basically, the RVM is a Bayesian treatment of a generalized linear model of similar functional form to

SVM. A prior over the model weights is introduced wherein each weight is associated with a hyper-parameter. The most probable values of these weights are then estimated iteratively from the data. The *relevance* of vectors comes from the weights associated with non-zero probability values of hyper-parameters. It has been reported that the RVM can maintain the generalization performance comparable to an equivalent SVM with typically much fewer kernel functions utilization (Tipping, 2000, 2001).

Another approach is by aggregation of variants of classifiers, which are trained diversely according to different feature and classifier settings. The bagging, boosting and random subspace methods are good examples for such approach (see e.g. Breiman, 1994; Skurichina & Duin, 2002; Tax & Duin, 2000; Vetter et al., 1997) where they have been shown to work well in practical problems with appropriate settings.

As above formulations are either iterative or aggregative in nature, they incur much computational effort in training and determining the parameters. We seek in this paper, a linear formulation similar to that of least-squares estimate, which provides possible means to stabilize the solution and hence reduces possible over-fitting. Instead of adopting the above hyperprior methods by statistical means, the stabilization problem is approached from the mapping point of view utilizing derived features obtained from scaling. The training effort to arrive at multiple classifiers for aggregation as in bagging technique is translated into the process of selecting a scaling parameter from a single-step least-squares training paradigm.

2.2. Pattern classification treatment

Given an *inference* measure g, for two-class problems, the threshold function $th : \Re \rightarrow \{0, 1\}$ given by

$$th(g) = \begin{cases} 0, & g < t_0 \\ 1, & g \geqslant t_1, \end{cases}, \quad t_0 \leqslant t_1, \tag{1}$$

is used for classification decision. For multi-class problems, given $\boldsymbol{g} = [g_1, g_2, \ldots, g_{N_C}]$ where N_C being the number of class labels, the following classification function $cls : \Re^l \rightarrow [1, 2, \ldots, N_C]$ is used to define each class label:

$$cls(\boldsymbol{g}) = arg\, max_i\ g_i,\ i = 1, 2, \ldots, N_C. \tag{2}$$

2.3. Feature scaling-space in our context

The *scale-space* concept has been much explored in image processing and computer vision (see e.g. Lindeberg, 1990). In the context of image processing and computer vision, scale-space refers to a family of derived signals that represent the original signal at various levels of scale (Lindeberg, 1990). These derived signals are usually obtained from convolution of the original data (image) with a kernel.

In our context here, feature *scaling-space* refers to the set of pattern features containing both the original and the derived signals. Unlike the above derived signals in scale-space or multiscale representation, our derived signals for *feature scaling-space* are obtained from simple uniform scaling of the original signal. The feature scaling-space is also differentiated from those 3L algorithms as seen in Helzer et al. (2004) and references within, in the way that the error objective of 3L and related algorithms are expressed using variations of the training target values, whereas in our feature scaling-space, similar target values are used.

Apart from the input scale parameters adaptation as seen in kernel or network parameter adjustment schemes (see e.g. Juszczak et al., 2000; Grandvalet & Canu, 2002; Tipping, 2001), the proposed feature scaling-space is also differentiated from the feature weighting and normalization in neural network learning schemes (see e.g. Bishop, 1995) and classification features selection (see e.g. Duch & Grudziński, 1999) since these schemes do not include multiple sets of scaled signals in processing. Scaling is an important preprocessing step in SVM and Neural Networks for pattern classification since it has notable outcomes for these methods. It is defined by a linear transformation of the input vector $\boldsymbol{x}$ space such that $\tilde{\boldsymbol{x}} = \boldsymbol{\Gamma x}$ where $\boldsymbol{\Gamma} = diag(\boldsymbol{\gamma})$, and $\boldsymbol{\gamma}$ is the scaling vector. However, the search for a good combination of these scaling values can be a daunting task since the number of possibilities increases tremendously with the input dimension.

Main reasons that feature scaling-space is not being explored in the literature include the following: (i) it is not applicable to linear decision models, (ii) its effect is notable for polynomials, but these polynomials are not well used due to the explosive number of expansion terms for high dimensional and high order problems, and (iii) its effect is also notable for nonlinear decision models, but they are often hampered by local solutions which render the effects non-conclusive.

3. A linear reciprocal-sigmoid model

3.1. The sigmoid function

The sigmoid activation function has been widely used in multilayer perceptron network applications. The success of such applications come from two aspects of its property: (i) its approximation capability has been well proven (see e.g. Hornik et al., 1989; Schürmann, 1996), and (ii) the convergence to local solutions is easily obtained in practice. For pattern classification, the sigmoid transformation can pull trained objects apart making them possibly linearly separable. We are thus motivated to explore into classification models adopting the sigmoid and related functions.

The wide usage of the sigmoid-based models however, is complicated by the effort required to fix the various learning hyper-parameters (e.g. number of layers, number of neurons in each layer, momentum, learning rate and etc.) for the learning to converge to some acceptable local solutions with good predictivity. In this section, observations on some properties of the sigmoid function are provided and then in next section, a linear model, which carries much of the approximation capability but at the same time reliefs much the effort to search for appropriate local solutions among the various combination of hyper-parameters, is proposed.

The sigmoid function is given by

$$\sigma(x) = \frac{1}{1 + e^{-x}}. \tag{3}$$

Rearranging (3) gives $e^{-x} = (1 - \sigma(x))/\sigma(x)$, which can be substituted into the following derivation:

$$\sigma(x_1 + x_2) = \frac{1}{1 + e^{-x_1} e^{-x_2}}$$

$$= \frac{1}{1 + \left(\frac{1-\sigma(x_1)}{\sigma(x_1)}\right)\left(\frac{1-\sigma(x_2)}{\sigma(x_2)}\right)}$$

$$= \frac{\sigma(x_1)\sigma(x_2)}{\sigma(x_1)\sigma(x_2) + (1-\sigma(x_1))(1-\sigma(x_2))}. \tag{4}$$

Dividing both numerator and denominator by $\sigma(x_1)\sigma(x_2)$, (4) can be re-written as

$$\sigma(x_1 + x_2) = \frac{1}{1+\omega}, \tag{5}$$

where

$$\omega = \frac{(1-\sigma(x_1))(1-\sigma(x_2))}{\sigma(x_1)\sigma(x_2)}$$

$$= \frac{1-\sigma(x_1)-\sigma(x_2)+\sigma(x_1)\sigma(x_2)}{\sigma(x_1)\sigma(x_2)}$$

$$= 1 - (\sigma(x_1))^{-1} - (\sigma(x_2))^{-1} + (\sigma(x_1)\sigma(x_2))^{-1}. \tag{6}$$

Using binomial series expansion, (5) can be expanded as

$$\sigma(x_1 + x_2) = (1+\omega)^{-1} = 1 - \omega + \omega^2 - \omega^3 + \omega^4 - \cdots. \tag{7}$$

Equations (7) and (6) show that $\sigma(x_1 + x_2)$ can be expressed in terms of products and powers of $(\sigma(x_1))^{-1}$ and $(\sigma(x_2))^{-1}$. Collectively, the following can be written:

$$\sigma(x_1 + x_2) = \alpha_0 + \alpha_1(\sigma(x_1))^{-1} + \alpha_2(\sigma(x_2))^{-1} + \alpha_3(\sigma(x_1)\sigma(x_2))^{-1}$$
$$+ \alpha_4(\sigma(x_1))^{-2} + \alpha_5(\sigma(x_2))^{-2} + \alpha_6(\sigma(x_1)\sigma(x_2))^{-2} + \cdots, \tag{8}$$

where α_i, $i = 0, 1, 2, \ldots$ are some scalar coefficients. This analysis shows that the reciprocal of sigmoid functions can be linearly combined by means of product and power terms, to achieve the equivalent 'shifting' and 'scaling' effects for a sigmoid function containing the nonlinear 'shifting' and 'scaling' parameters (i.e. a and b in $\sigma(a(x-b))$). An immediate implication is that any scaled or shifted sigmoid function may be expanded as a series consisting of linear combination of product and power terms of the reciprocal of a basis sigmoid function. A collection of these series, which represent differently scaled and shifted sigmoids, can thus be used for function approximation.

It is noted that the reciprocal of the sigmoid function is actually $1 + e^{-x}$. This means that no reciprocal inversion is required to be performed on the term $(1 + e^{-x})$ as in the case of using the usual sigmoid function.

3.2. Linear combination of reciprocal-sigmoids

The property given in above subsection suggests certain approximation capability of the reciprocal-sigmoid model since, in theory, it has been shown that standard multilayer feed-forward networks with as few as one hidden layer using arbitrary squashing functions are

capable of approximating any Borel measurable function from one finite dimensional space to another to any desired degree of accuracy (see e.g. Hornik et al., 1989; Schürmann, 1996).

The full multivariate polynomial expansion provides a straight-forward means to span the linear combination of product and power terms. However, the number of terms grows exponentially with the number of inputs and the order of the system. To significantly reduce the huge number of terms in multivariate polynomials, a Reduced multivariate polynomial Model (RM) has been proposed by Toh (2003):

$$\begin{aligned} g_{RM}(\boldsymbol{\alpha}, \boldsymbol{x}) &= \boldsymbol{\alpha}^T p_{RM}(\boldsymbol{x}) \\ &= \alpha_0 + \sum_{k=1}^{r}\sum_{j=1}^{l} \alpha_{k,j} x_j^k + \sum_{k=1}^{r} \alpha_k \left(\sum_{j=1}^{l} x_j\right)^k \\ &\quad + \sum_{k=2}^{r} \left(\sum_{i=1}^{l} \alpha_{k,i} x_i\right) \left(\sum_{j=1}^{l} x_j\right)^{k-1}, \quad l, r \geqslant 2. \end{aligned} \tag{9}$$

where x_j, $j = 1, \ldots, l$ are the polynomial inputs, $\alpha_0, \alpha_{k,j}, \alpha_k, \alpha_{k,i}, \ldots$ are the weighting coefficients to be estimated, and l, r correspond to input-dimension, order of system respectively. The number of terms in this model can be expressed as $K = 1 + r + l(2r - 1)$.

For this case using the reciprocal-sigmoid as basis function, the polynomial inputs x_j are replaced by the reciprocal-sigmoid basis functions:

$$\phi(x_j) = (\sigma(x_j))^{-1} = 1 + e^{-x_j}, \quad j = 1, \ldots, l. \tag{10}$$

As can be seen from (9), the basic component of this polynomial model boils down to construction of new pattern features which are sums of the original features, and combination of these new and original features using power and product terms.

Table 1 lists the expansion terms of RM given by (9). It is noted that some parts of (9) can be obtained from linear combination of other terms. These linear dependencies are indicated in Table 1 by various corresponding markings, e.g. underlined term in column three corresponds to underlined terms in other columns (e.g. column-two), over-braced term in column three corresponds to over-braced terms in other columns (e.g. column-four), and so on. By removing these linearly dependent cases from column three of Table 1, the RM2 model (a refinement from RM) is defined as follows:

$$\begin{aligned} g_{RM2}(\boldsymbol{\alpha}, \boldsymbol{x}) &= \boldsymbol{\alpha}^T p_{RM2}(\phi(\boldsymbol{x})) \\ &= \alpha_0 + \sum_{k=1}^{r}\sum_{j=1}^{l} \alpha_{k,j} \phi(x_j)^k \\ &\quad + \sum_{k=2}^{r} \left(\sum_{i=1}^{l} \alpha_{k,i} \phi(x_i)\right) \left(\sum_{j=1}^{l} \phi(x_j)\right)^{k-1}, \quad l, r \geqslant 2 \end{aligned} \tag{11}$$

where $\phi(\cdot)$ is defined as in (10). Here, (11) will be called a reciprocal-sigmoid model as it is composed of a linear combination of sum and product terms of reciprocal sigmoids. It is noted that the total number of expansion terms of p_{RM2} is given by $K = 1 + l(2r - 1)$, which

Table 1 Full view of terms in RM model ($l = 3, r = 2, \ldots, 5$)

Order	2nd-term of (9)	3rd-term of (9)	4th-term of (9)
$r = 2$	$\underline{x_1, x_2, x_3}, x_1^2, x_2^2, x_3^2$	$\underline{(x_1 + x_2 + x_3)}, \overbrace{(x_1 + x_2 + x_3)^2}$	$\overbrace{x_1(x_1 + x_2 + x_3)}, \overbrace{x_2(x_1 + x_2 + x_3)}, \overbrace{x_3(x_1 + x_2 + x_3)}$
$r = 3$	$\underline{x_1, x_2, x_3}, x_1^2, x_2^2, x_3^2, x_1^3, x_2^3, x_3^3$	$\underline{(x_1 + x_2 + x_3)}, \overbrace{(x_1 + x_2 + x_3)^2}, \underbrace{(x_1 + x_2 + x_3)^3}$	$\overbrace{x_1(x_1 + x_2 + x_3)}, \overbrace{x_2(x_1 + x_2 + x_3)}, \overbrace{x_3(x_1 + x_2 + x_3)}, \underbrace{x_1(x_1 + x_2 + x_3)^2}, \underbrace{x_2(x_1 + x_2 + x_3)^2}, \underbrace{x_3(x_1 + x_2 + x_3)^2}$
$r = 4$	$\underline{x_1, x_2, x_3}, x_1^2, x_2^2, x_3^2, x_1^3, x_2^3, x_3^3, x_1^4, x_2^4, x_3^4$	$\underline{(x_1 + x_2 + x_3)}, \overbrace{(x_1 + x_2 + x_3)^2}, \underbrace{(x_1 + x_2 + x_3)^3}, \overline{(x_1 + x_2 + x_3)^4}$	$\overbrace{x_1(x_1 + x_2 + x_3)}, \overbrace{x_2(x_1 + x_2 + x_3)}, \overbrace{x_3(x_1 + x_2 + x_3)}, \underbrace{x_1(x_1 + x_2 + x_3)^2}, \underbrace{x_2(x_1 + x_2 + x_3)^2}, \underbrace{x_3(x_1 + x_2 + x_3)^2}, \overline{x_1(x_1 + x_2 + x_3)^3}, \overline{x_2(x_1 + x_2 + x_3)^3}, \overline{x_3(x_1 + x_2 + x_3)^3}$
$r = 5$	$\underline{x_1, x_2, x_3}, x_1^2, x_2^2, x_3^2, x_1^3, x_2^3, x_3^3, x_1^4, x_2^4, x_3^4, x_1^5, x_2^5, x_3^5$	$\underline{(x_1 + x_2 + x_3)}, \overbrace{(x_1 + x_2 + x_3)^2}, \underbrace{(x_1 + x_2 + x_3)^3}, \overline{(x_1 + x_2 + x_3)^4}, \overleftarrow{(x_1 + x_2 + x_3)^5}$	$\overbrace{x_1(x_1 + x_2 + x_3)}, \overbrace{x_2(x_1 + x_2 + x_3)}, \overbrace{x_3(x_1 + x_2 + x_3)}, \underbrace{x_1(x_1 + x_2 + x_3)^2}, \underbrace{x_2(x_1 + x_2 + x_3)^2}, \underbrace{x_3(x_1 + x_2 + x_3)^2}, \overline{x_1(x_1 + x_2 + x_3)^3}, \overline{x_2(x_1 + x_2 + x_3)^3}, \overline{x_3(x_1 + x_2 + x_3)^3}, \overleftarrow{x_1(x_1 + x_2 + x_3)^4}, \overleftarrow{x_2(x_1 + x_2 + x_3)^4}, \overleftarrow{x_3(x_1 + x_2 + x_3)^4}$

has r terms less than that in (9). The simplicity of the classifier model can be seen from the few lines of Matlab (The MathWorks, 2003) codes as shown in Appendix A.1. A XOR learning example is included in Appendix A.2 for immediate test.

With this refined model RM2 in place, the following formulation for multiple-class problems is presented:

Proposition 1. *Given m finite data samples $\{(\boldsymbol{x}_i, \boldsymbol{y}_i)|\boldsymbol{x}_i \in \Re^l, \boldsymbol{y}_i \in \Re^q, i = 1, \ldots, m\}$ with the multiple-class (q-classes) output vectors packed as $Y = [\boldsymbol{y}_1, \ldots, \boldsymbol{y}_m]^T \in \Re^{m \times q}$. Let $p_{RM2} : \Re^l \to \Re^K$ in (11) be the model expansion function on the reciprocal-sigmoid basis $\phi(\boldsymbol{x}_i)$, $i = 1, \ldots, m$ with $K \leqslant m$. If the matrix $P = [p_{RM2}(\phi(\boldsymbol{x}_1)), \ldots, p_{RM2}(\phi(\boldsymbol{x}_m))]^T$ has full rank, then there is a minimum $\epsilon \geqslant 0$ and an optimal solution $\Theta^* = [\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_q]$ such that $\|Y - P\Theta^*\|_2^2 \leq \epsilon$.*

Proof: The least-squares solution minimizing $\|Y - P\Theta\|_2^2$ is $\Theta^* = (P^T P)^{-1} P^T Y$ and the full rank condition guarantees that the matrix $P^T P$ is invertible. Hence the proof. □

Although RM2 contains independent expansion terms, duplicate data samples can still form dependent vectors. The full rank condition requires the data samples to be distinct in some sense, i.e. they do not form dependent column vectors of P for $K \leqslant m$ (K is the number of polynomial terms and m is the number of data samples).

This result shows the approximation capability for uncorrelated discrete data. When $K = m$, an exact fit to pass through all data points is possible when the matrix P is not ill-conditioned. It is also noted that the above results apply to the original polynomials as well as adopting the reciprocal-sigmoid as basis function. We shall illustrate empirically in later section that using the reciprocal-sigmoid basis function, which embeds much nonlinearity, can reduce the number of expansion terms as compared to that using polynomials.

3.3. Relationship with generalized linear models (GLM)

Using the sigmoid activation function (3) as a *perceptron* unit taking multiple inputs, a perceptron network can be constructed and employed to learn binary observations for classification (Bishop, 1995). Mathematically, a logistic regression perceptron model for learning binary data can be considered as a Generalized Linear Model with *binomial* response and *logit* link (Nelder & Wedderburn, 1972; McCullagh & Nelder, 1989; Hardin & Hilbe, 2001). Here, the m independent observations y_i $(i = 1, \ldots, m)$ are treated as a realization of a random variable $Y_i \sim B(n_i, \pi_i)$ with binomial denominator n_i and probability π_i (for 2-class problems, $n_i = 1, \forall i$). A linear predictor function (the systematic component) is then used to fit the *logit* (the link function) of the underlying probability π_i (the random component) giving rise to

$$logit(\pi_i) = \log \frac{\pi_i}{1 - \pi_i} = \boldsymbol{x}_i^T \boldsymbol{\beta}, \tag{12}$$

where $\boldsymbol{x}_i$ is a vector of covariates (data inputs) and $\boldsymbol{\beta}$ is a vector of regression coefficients (parameters). Solving for the probability π_i in (12) yields

$$\pi_i = \frac{1}{1 + e^{-\boldsymbol{x}_i^T \boldsymbol{\beta}}}, \tag{13}$$

where (13) can be associated to (3) by putting $\pi_i = \sigma(\boldsymbol{x}_i^T \boldsymbol{\beta})$. The main difference between a perceptron and a GLM is whether the underlying truth of data distribution is explicitly assumed or not. While the GLM exploits statistical inference for analysis, the perceptron does not explicitly assume data distribution and works on empirical evidences such as validation or training/test errors. This empirical framework will be adopted in our evaluations.

As mentioned, the main problem with the above perceptron and GLM frameworks is the parametric nonlinearity, and an iterative search is often required to solve for $\boldsymbol{\beta}$ (see e.g. Dunn, 1999, 2000). In this work, the nonlinearity problem is solved by a simple re-formulation. Instead of solving directly for $\boldsymbol{\beta}$ in (13), $\sigma(\boldsymbol{x}^T \boldsymbol{\beta})$ is re-written in the form $\sigma(\beta_1 x_1 + \cdots + \beta_l x_l)$ that is analogous to (8). Here, it is seen that $\sigma(\beta_1 x_1)$ takes the place of $\sigma(x_1)$ in (8) and so on. Since every variable x_j is decoupled from all other variables $x_k, k \neq j$ in (8), $\boldsymbol{\beta}$ here can be absorbed into the parameter vector $\boldsymbol{\alpha}$ during estimation. For example, $\bar{\alpha}_3(\sigma(\beta_1 x_1)\sigma(\beta_2 x_2))^{-1}$ can be written as $\alpha_3(\sigma(x_1)\sigma(x_2))^{-1}$ using a re-parameterized $\alpha_3 = \bar{\alpha}_3(f(\beta_1)f(\beta_2))^{-1}$ for a certain transformation function f obtained from $f(\beta_i)\sigma(x_i) = \sigma(\beta_i x_i)$. We need not worry about determining f and β's in practice because only the global effect of $\boldsymbol{\alpha}$ (α_3 for this particular example) is estimated.

The full multivariate polynomial (FM) can be utilized to realize the combinatorial terms in (8). However, as mentioned, the implementation cost would be high because the number of expansion terms grows exponentially with respect to the number of inputs and the order of system. The RM2 model alleviates this problem by removing many expansion terms and

replacing them with a few multinomial product terms. There is thus a certain difference between (8) realized by FM and (8) realized by RM2 (11) due to this approximation. Under the language of the GLM framework, the *logit* link has been approximated by the functional inverse of the RM2 series expansion. This approximated link function and the loss function (least squares objective in Proposition 1) are matched to facilitate convex and well-posed estimation like a linear predictor. Empirical evidence regarding the classification capability of this reciprocal-sigmoid classifier, which has been realized by RM2 series approximation, shall be explored in the experiment section.

4. Learning of scaling-space features

Learning a classifier without over-fitting the data is an important issue in machine learning. In this section, the well-used technique on weight decay regularization to stabilize the learning process is presented prior to proposal of a scaling-space approach to further stabilize the learning. The main reason for regularization is that it is a practical and useful means to stabilize the solution which, in some way, relates to generalization. The feature scaling-space learning will be detailed shortly.

4.1. Weight decay regularization

It is noted here that the linear least-squares estimation involves computation of inverse of a matrix, the problem of multi-collinearity may arise if some linear dependence among the elements of $\boldsymbol{x}$ are present. A simple approach to improve numerical stability is to perform a weight decay regularization using the following error objective:

$$s(\boldsymbol{\alpha}, \boldsymbol{x}) = \sum_{i=1}^{m} [y_i - g_{_{RM2}}(\boldsymbol{\alpha}, \boldsymbol{x}_i)]^2 + b\|\boldsymbol{\alpha}\|_2^2 = [\mathbf{y} - P\boldsymbol{\alpha}]^T [\mathbf{y} - P\boldsymbol{\alpha}] + b\,\boldsymbol{\alpha}^T\boldsymbol{\alpha}, \tag{14}$$

where $\|\cdot\|_2$ denotes the l_2-norm and b is a regularization constant. Let $\boldsymbol{g}_{_{RM2}}(\boldsymbol{\alpha}, \boldsymbol{x})$ be the output column vector of the model for m number of training samples and suppose there are K number of polynomial weights, then P is the Jacobian given by

$$P = \left(\frac{\partial \boldsymbol{g}_{_{RM2}}(\boldsymbol{\alpha}, \boldsymbol{x})}{\partial \boldsymbol{\alpha}^T} \right) \in \Re^{m \times K}. \tag{15}$$

For two-class problems, minimizing the objective function (14) results in

$$\boldsymbol{\alpha} = (P^T P + bI)^{-1} P^T \mathbf{y}, \tag{16}$$

where $\mathbf{y} \in \Re^{m \times 1}$ and I is a $(K \times K)$ identity matrix. For multiple class problems, the solution can be written as

$$\Theta = (P^T P + bI)^{-1} P^T Y, \tag{17}$$

since the matrix P is similar for all outputs and these multiple outputs can be stacked as Y (i.e. $Y = [\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_{N_C}]$, N_C being the number of pattern classes, see (Toh et al., 2004) for more details). The addition of a bias term into the least-squares regression model is also

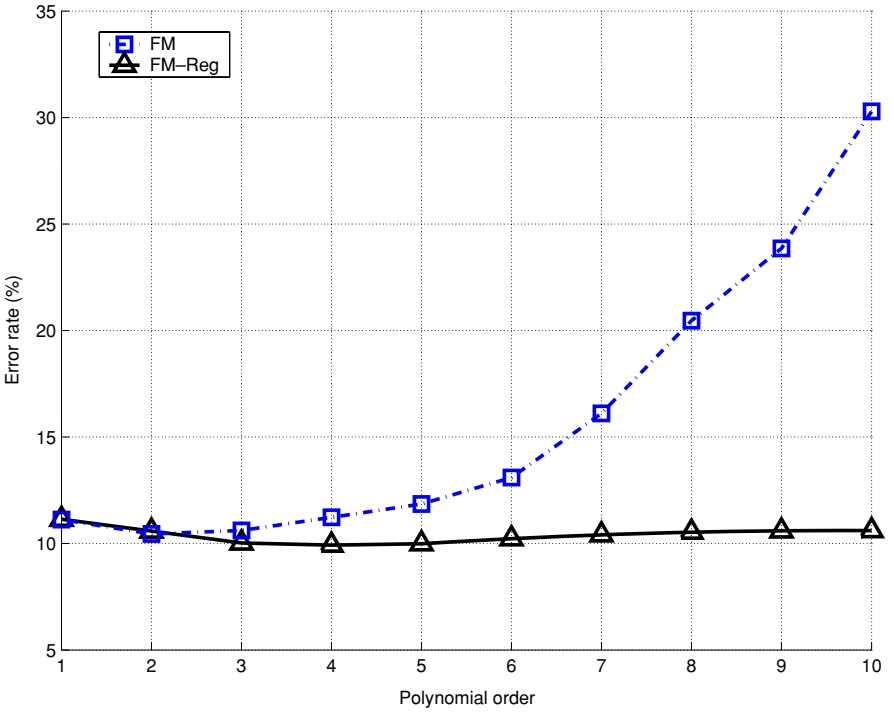


Fig. 1 Average test error rates versus polynomial order. This example provides a comparison between regularized (FM-Reg) and non-regularized (FM) full multivariate polynomials using Ripley's synthetic data (Ripley, 1996). The error rates were obtained from 1000 runs of training the 100-example subset randomly selected from the original 250-example training set and used the 1000-example test set for test error rates computation. It is seen that regularization demonstrates significant improvement of numerical stability in this example.

termed as *ridge regression* (Neter et al., 1996). The effect of regularization can be seen from the experiments as shown in Fig. 1.

Having learned Θ and with matrix P_t generated from test data, the classification function given by (2) can be applied to the test output $\hat{Y} = P_t\Theta$ to determine the class label.

4.2. Feature scaling-space learning

Let $\boldsymbol{x} \in \Re^l$ be a l-dimensional pattern feature. The feature scaling-space is defined as $\{\boldsymbol{x}, \gamma_1\boldsymbol{x}, \gamma_2\boldsymbol{x}, \ldots\}$ where γ_i, $i = 1, 2, \ldots$ are the scaling factors for each derived feature $\gamma_i\boldsymbol{x}$. In other words, for each data sample $\boldsymbol{x}$, there corresponds additional scaled samples $\gamma_i\boldsymbol{x}$ that can be used for training. In the proposed learning framework, a symmetric scaling is adopted and defined as follows: $\gamma_1 = (1 - \rho)$ and $\gamma_2 = (1 + \rho)$ where $0 < \rho < 1$. The final feature scaling-space is thus the set of original and scaled samples containing $\{\gamma_1\boldsymbol{x}, \boldsymbol{x}, \gamma_2\boldsymbol{x}\}$. The training target vector contains replicates of the original target vector without scaling, i.e. $\{\mathbf{y}, \mathbf{y}, \mathbf{y}\}$. For testing, the original test inputs (i.e. without scaling) are used to predict the classification outcome. Obviously, differently scaled test inputs can be used individually for predicting the classification outcomes and these test outcomes can be aggregated to form the final decision. As there exists vast combinations of such aggregation which take up much computing effort, only the original test inputs will be used in this study.

Table 2 Full view of terms in a scaled RM2 model ($l = 3$, $r = 2, 3, 4$)

Order	2nd-term of (11)	3rd-term of (11)
$r = 2$	$\gamma x_1, \gamma x_2, \gamma x_3, \gamma^2 x_1^2, \gamma^2 x_2^2, \gamma^2 x_3^2$	$\gamma^2 x_1(x_1 + x_2 + x_3)$, $\gamma^2 x_2(x_1 + x_2 + x_3)$, $\gamma^2 x_3(x_1 + x_2 + x_3)$
$r = 3$	$\gamma x_1, \gamma x_2, \gamma x_3, \gamma^2 x_1^2, \gamma^2 x_2^2, \gamma^2 x_3^2$, $\gamma^3 x_1^3, \gamma^3 x_2^3, \gamma^3 x_3^3$	$\gamma^2 x_1(x_1 + x_2 + x_3)$, $\gamma^2 x_2(x_1 + x_2 + x_3)$, $\gamma^2 x_3(x_1 + x_2 + x_3)$, $\gamma^3 x_1(x_1 + x_2 + x_3)^2$, $\gamma^3 x_2(x_1 + x_2 + x_3)^2$, $\gamma^3 x_3(x_1 + x_2 + x_3)^2$
$r = 4$	$\gamma x_1, \gamma x_2, \gamma x_3, \gamma^2 x_1^2, \gamma^2 x_2^2, \gamma^2 x_3^2$, $\gamma^3 x_1^3, \gamma^3 x_2^3, \gamma^3 x_3^3, \gamma^4 x_1^4, \gamma^4 x_2^4, \gamma^4 x_3^4$	$\gamma^2 x_1(x_1 + x_2 + x_3)$, $\gamma^2 x_2(x_1 + x_2 + x_3)$, $\gamma^2 x_3(x_1 + x_2 + x_3)$, $\gamma^3 x_1(x_1 + x_2 + x_3)^2$, $\gamma^3 x_2(x_1 + x_2 + x_3)^2$, $\gamma^3 x_3(x_1 + x_2 + x_3)^2$ $\gamma^4 x_1(x_1 + x_2 + x_3)^3$, $\gamma^4 x_2(x_1 + x_2 + x_3)^3$, $\gamma^4 x_3(x_1 + x_2 + x_3)^3$

Notice that $\rho = 0$ corresponds to two redundant sets of data added to the original data $\boldsymbol{x}$. The effect is that a heavier weight would be given to least-squares error as compared to the regularization bias when a regularized error objective is adopted. When $\rho \neq 0$, this approach can turn an under-determined system to an over-determined system where the number of training samples is larger than the number of parameters. If however, $\boldsymbol{x}$ and $\gamma_i \boldsymbol{x}$ are concatenated to form a large dimension feature (e.g. $[\boldsymbol{x}^T, \gamma_1 \boldsymbol{x}^T, \gamma_2 \boldsymbol{x}^T, \ldots]$), under-determined system can arise for cases with small number of training samples.

For multivariate polynomials, each scaling of the training feature by γ gives rise to a nonlinearly scaled (hopefully independent) feature from the original one whenever γ is raised to different powers in the polynomial expansion. For example, x_1^2 will be nonlinearly scaled by γ^2 in the second-order expansion. The result of such scaling of training samples, even though it is uniform in the input space, could give rise to *independent* data samples in the polynomial feature space. This increases the number of training samples without 'distorting' the original signal like those using noise injection (Skurichina et al., 2000; Grandvalet, 2000) that may result in adding of infeasible samples. For RM2 formulation, the scaled expansion terms are shown in Table 2 for $r = 2, 3, 4$.

5. Ripley's synthetic data

In this section, Ripley's synthetic data is used to evaluate the accuracy and computing aspects of scaling-space learning on the reciprocal-sigmoid classifier and on the full multivariate polynomial model. The data consists of a binary output indicating two pattern classes with each sample containing two feature elements. According to (Tipping, 2001), a 100-example training set (randomly selected from Ripley's original 250) and a 1000-example test set was used to demonstrate the decision boundary and generalization capability of RVM. It was shown by Tipping (2000, 2001) that RVM produces comparable test error (slightly better) with that of a SVM while utilizing a considerably small number of Gaussian kernel functions.

In all the experiments to follow, 1000 runs of training were performed using each 100-example subset, which was randomly selected from the original 250-example training set. The same set of randomly selected data for all 1000 runs was used in all compared algorithms. This setup is to provide a good statistical picture regarding the performances, and especially the generalization property.[1] The feature scaling-space learning technique will be first demonstrated that it can be applied to a conventional full multivariate polynomial model. Figure 2

[1]For the Ripley's data, Tipping used a single run in Tipping (2001).

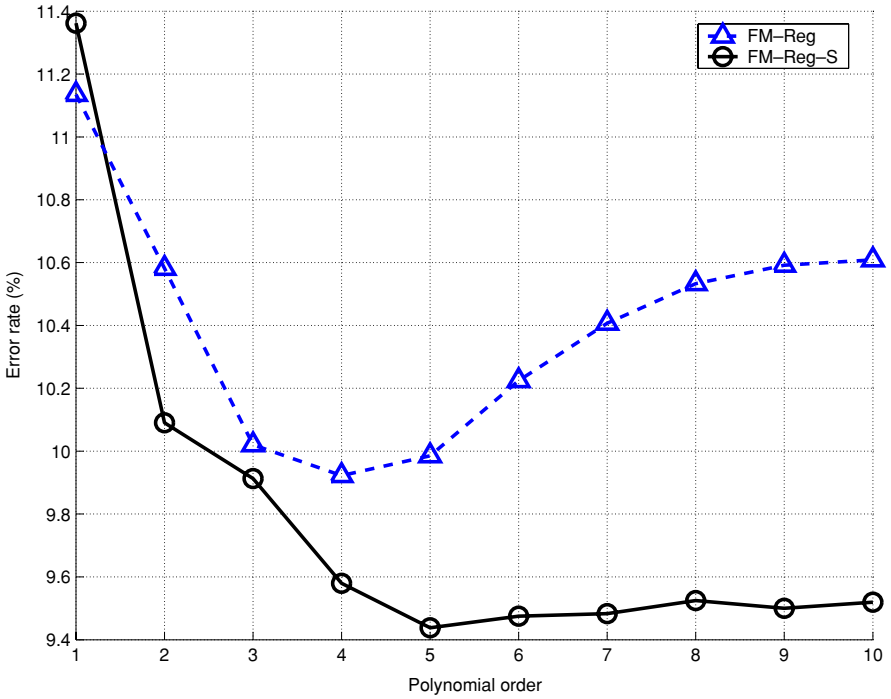


Fig. 2 Average (1000 runs) test error rates versus polynomial order. This example compares between FM-Reg and FM-Reg-S, which shows that feature scaling-space learning provides possible room for stabilization of the conventional full multivariate polynomial model at high model orders

shows the corresponding average error rates (from 1000 runs) plotted against each polynomial order for the regularized full multivariate polynomial regression with and without scaling: (i) FM-Reg: full polynomial with $b = 0.1$, and (ii) FM-Reg-S: full polynomial with $b = 0.1$ plus scaled input with $\rho = 0.2$ which was chosen empirically. These results show that the scaling-space learning provides possible room for stabilization of FM at high model orders.

In the next set of experiments, the reciprocal-sigmoid classifier will be compared with SVM, RVM and a Bagged FM-Reg. Both SVM and RVM adopt the polynomial kernel because our reduced model for comparison is also a polynomial. The reciprocal-sigmoid model expanded by RM2 with regularized scaling-space learning is abbreviated as RM-InvSig-S. The non-scaled learning for the reciprocal-sigmoid model is abbreviated similarly but without '-S' attached. Both RM-InvSig-S and FM-Reg-S used a regularization setting of $b = 0.1$. The Bagged FM-Reg, which aggregates (averages) 50 bootstrapped classifiers from re-sampling the training data with replacement (Breiman, 1994), is abbreviated as FM-Reg-Bagging. Figure 3 shows the average test error rates for 1000 runs of SVM, RVM, FM-Reg-Bagging, RM-InvSig-S and FM-Reg-S over 10 model orders.

For SVM, in order to demonstrate the effect of different parameter tunings for the 10 model orders, four settings of (C, *Gamma*, *Coeff*)[2] are used: (i) SVM-1 with (3,1,1); (ii)

[2]In Ma et al. (2002), C is the cost of constraint violation, *Gamma* and *Coeff* are the parameters of the polynomial kernel which has the form of $[Gamma* < X(:,i), X(:,j) > + Coeff]^{Degree}$. *Degree* is the order of the polynomial.

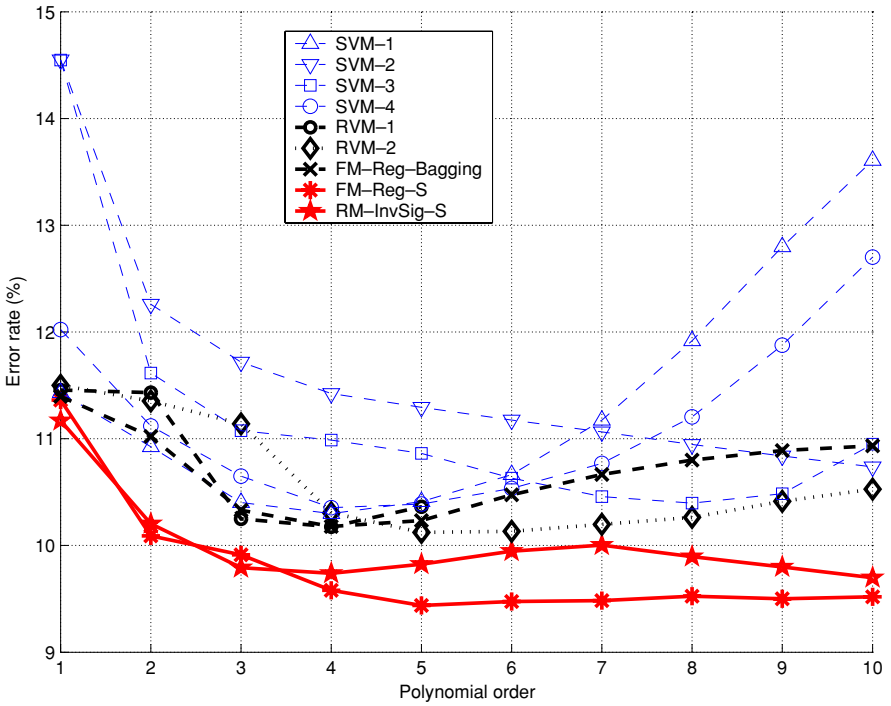


Fig. 3 Average (1000 runs) test error rates versus polynomial order. A comparison of error performance among SVM, RVM, FM-Reg-Bagging, FM-Reg-S and RM-InvSig-S over 10 model orders. This plot shows that RVM, FM-Reg-Bagging and the feature scaling-space methods (FM-Reg-S and RM-InvSig-S) provide good estimation stability over wide rage of model orders while SVM tends to have high test error rates at low and high order models due to under-fitting and over-fitting

SVM-2 with (3,0.1,1); (iii) SVM-3 with (3,0.1,3) and (iv) SVM-4 with (1,1,1). The RVM-1 (using default settings: width = 0.5, maximum number of iterations = 500) in this figure has only error rates for 1st-5th orders because matrix singularities are frequently encountered for the experimented data set for higher orders. RVM-2 used a width of 1 with maximum number of iterations equal 500 for all model orders.

The experiments show that SVM tends to have high test error rates at low and high order models due to under-fitting and over-fitting. The RVMs, FM-Reg-Bagging, FM-Reg-S and RM-InvSig-S maintained relatively consistent test error rates throughout all experimented model orders. The average error rates for RM-InvSig-S and FM-Reg-S are seen to be consistently lower (approx. 0.5–1.5%) than those of RVM and FM-Reg-Bagging for all model orders.

The average error rates for RM-InvSig-S are compared to those from the original RM-InvSig with similar parameter settings (i.e. $b = 0.1$) in Fig. 4. The thick line marked with '★' ('+') in the figure indicates the average error rates for RM-InvSig-S (RM-InvSig). The width of the light shaded tone on RM-InvSig indicates the standard deviation of error distribution. The dark tone indicates the standard deviation for error rates of RM-InvSig-S. This figure shows that up to 1.5% improvement of average test error over that of RM-InvSig can be achieved by RM-InvSig-S.

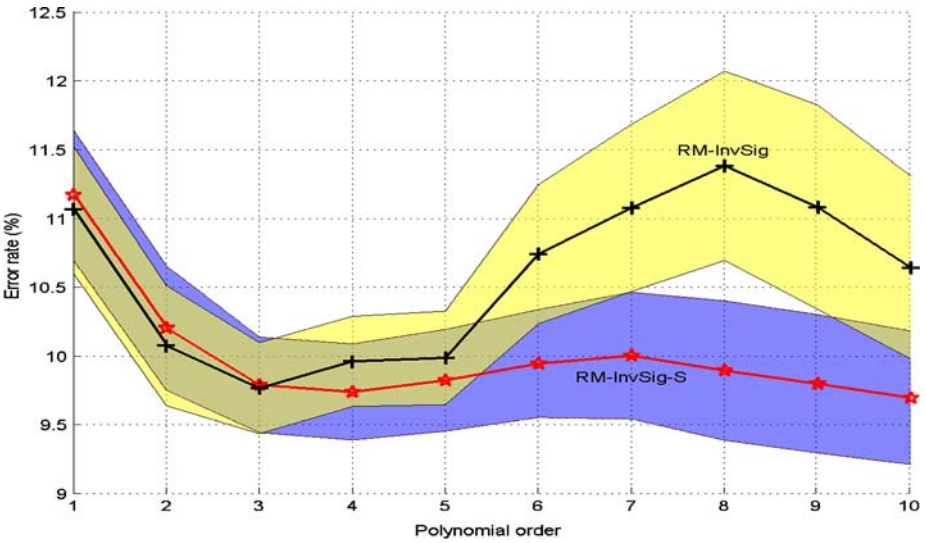


Fig. 4 Average (1000 runs) test error rates versus polynomial order. A comparison between RM-InvSig-S and the original RM-InvSig without the use of feature scaling space. The plot shows that up to 1.5% improvement of average test error over that of RM-InvSig can be achieved by RM-InvSig-S

Figure 5(a)–(b) shows the relative computing efforts for RVM, SVM, FM-Reg-Bagging, RM-InvSig, RM-InvSig-S and FM-Reg-S in terms of CPU time running under similar computing platform (1.4 GHz Pentium-Centrino, Matlab environment; The MathWorks, 2003). It is seen from this plot that for RM-InvSig-S, the increment of CPU overheads over the original RM-InvSig is relatively small as compared to RVM over SVM.

A training instance using a 5th-order polynomial for RM-InvSig, RM-InvSig-S and the RVM are plotted in Figure 6. The test data with decision boundary is also included in the figure. The decision boundaries at different decision thresholds ($p = 0.25/0.75$ and decision boundary at $p = 0.50$) for RVM, RM-InvSig and RM-InvSig-S are also shown in the figures where it is seen that RM-InvSig-S has lower localized decision boundary than those of RM-InvSig and RVM. In RVM, the relevance vectors control the localization of decision boundary whereas in RM-InvSig-S, the scaling of feature inputs affects the decision boundary.

6. Experiments on UCI data

To gather more information about the effect of the proposed linear machine on real-world problems, further experiments were performed on 40 benchmark UCI data sets (Newman et al., 1998). These data sets were selected based on (Toh et al., 2004) where 2 out of the total 42 data sets were left out due to matrix ill-conditioning. The data sets and algorithm settings are shown in Table 3. Both the RM-InvSig algorithm (reciprocal-sigmoid model given by (11)) with its respective *Scaled* counterpart (RM-InvSig-S) will be evaluated. The original polynomial expansion of RM2 is denoted as RM2.

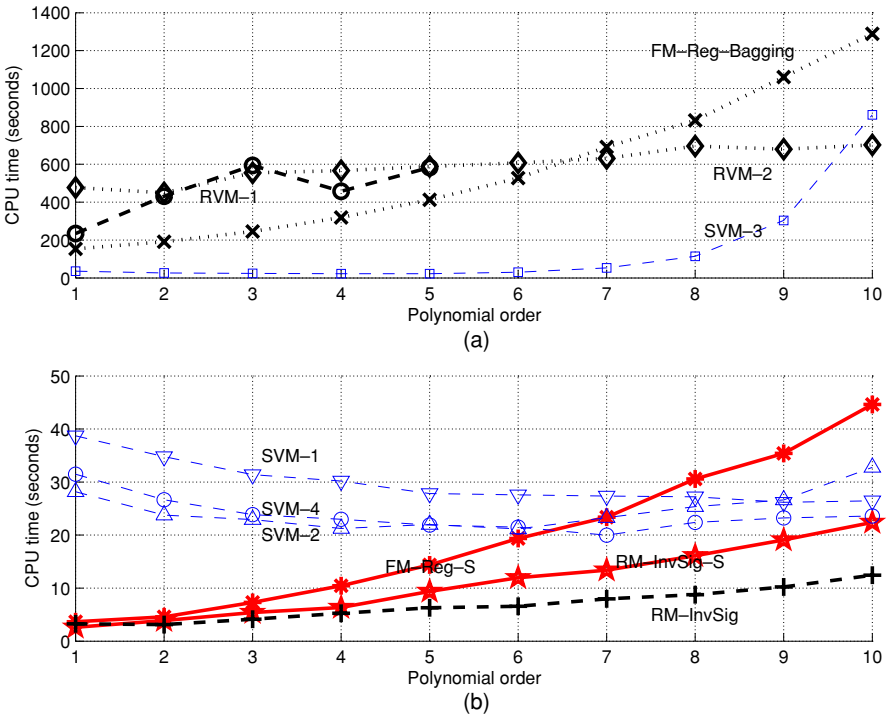


Fig. 5 Total CPU times for 1000 runs versus polynomial order. This plot shows the relative computing efforts among RVM, SVM, FM-Reg-Bagging, RM-InvSig, RM-InvSig-S and FM-InvSig-S in terms of CPU time running under similar computing platform. It can be seen from this plot that for feature scaling-space, the increment of CPU overheads for RM-InvSig-S over the original RM-InvSig is relatively small as compared to RVM over SVM

6.1. Experimental setup

In all the following experiments, the regularization parameter was fixed at $b = 0.0001$ according to Toh et al. (2004). Since the purpose here is to study the impact of introduction of the reciprocal-sigmoid basis function and scaling-space learning into the RM model, the polynomial order r was fixed following the RM-Tuned algorithm for each data set as seen in Toh et al. (2004).[3] The scaling parameter ρ was chosen from $\{0, 0.05, 0.10, \ldots 0.45\}$ based on 10-fold validation using only the training set (see Table 3). Similar to (Toh et al., 2004), ten runs of 10-fold stratified cross-validation were performed for all accuracy results concerned. Both the RM-InvSig and the scaled RM-InvSig-S algorithms were experimented under same settings stated above.

[3]In Toh et al. (2004), the polynomial order r was chosen from $\{1, 2, \ldots, 10\}$ based on cross-validation using only the training set. The selected model order r was then used to compute the accuracies for all the 10 runs of ten-fold tests. For the current RM-InvSig and RM-InvSig-S algorithms, r was selected similar to those chosen values for RM-Tuned in order to study the impact of new basis function and scaling-space learning on test accuracy.

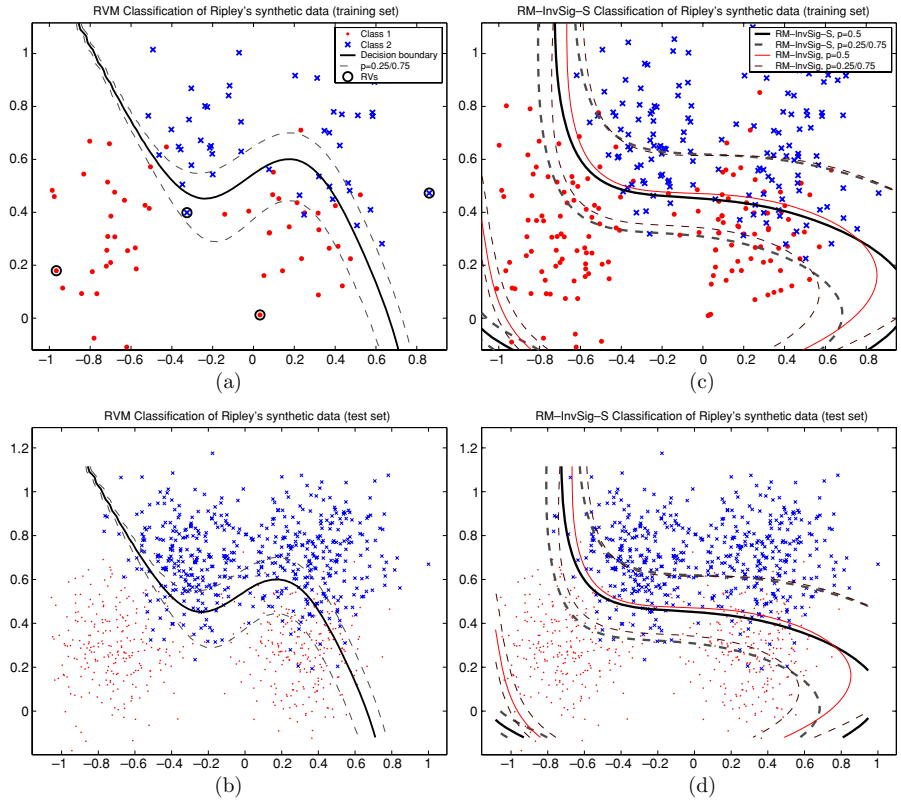


Fig. 6 Classification of Ripley's data: (a) decision boundary of RVM plotted on training set, (b) decision boundary of RVM plotted on test set, (c) decision boundaries of RM-InvSig and RM-InvSig-S plotted on scaling-space training set, (d) decision boundaries of RM-InvSig and RM-InvSig-S plotted on test set. $p = 0.5$ refers to decision boundary at threshold 0.5. These plots show that RM-InvSig-S has lower localized decision boundary than those of RM-InvSig and RVM

6.2. Classifiers comparison in literature

For comparisons against performances of existing classification algorithms in the literature, the classification results were directly taken from Lam et al. (2002), Lim et al. (2000), Precup and Utgoff (2004), Li et al. (2004) since much similarity among the experimental conditions (including ours) can be identified, besides the fact that the best known tuning of their proposed methods have been obtained by the originators themselves. The selected 40 sets of UCI data are mainly based on those in Toh et al. (2004), Lam et al. (2002) and Lim et al. (2000). For (Precup & Utgoff, 2004) and (Li et al., 2004), only data sets common to the chosen 40 sets were compared. As details of experiments can be found in the references, these comparative works are only briefly described below for immediate reference.

Ref-I: In the recent work by Lam et al. (2002), the proposed best tuned algorithm ICPL (Integrated Concept Prototype Learner, which integrates instance filtering and abstraction techniques) was compared with four other algorithms namely, RT3 (an instance pruning

Table 3 Summary of data sets and algorithm settings

		(i)	(ii)	(iii)	(iv)	RM-InvSig-S ($b = 0.0001$)	
	Database name	#cases	#feat	#class	#miss	r	ρ
1.	Shuttle-l-control	279(15)	6	2	no	2	0.15
2.	BUPA-liver-disorder	345	6	2	no	2	0
3.	Monks-1	124(432)	6	2	no	4	0.45
4.	Monks-2	169(432)	6	2	no	7	0
5.	Monks-3	122(432)	6	2	no	2	0
6.	Pima-diabetes	768	8	2	no	1	0.05
7.	Tic-tac-toe	958	9	2	no	2	0
8.	Breast-cancer-Wiscn	683(699)	9(10)	2	16	3	0.05
9.	StatLog-heart	270	13	2	no	2	0.10
10.	Credit-app	653(690)	15	2	37	1	0
11.	Votes	435	16	2	yes	2	0.05
12.	Mushroom	5644(8124)	22	2	attr#11	2	0
13.	Wdbc	569	30	2	no	3	0.20
14.	Wpbc	194(198)	33	2	4	1	0.35
15.	Ionosphere	351	34	2	no	3	0.40
16.	Sonar	208	60	2	no	1	0.05
17.	Iris	150	4	3	no	4	0.10
18.	Balance-scale	625	4	3	no	2	0.10
19.	Teaching-assistant	151	5	3	no	9	0.05
20.	New-thyroid	215	5	3	no	4	0.10
21.	Abalone	4177	8	3(29)	no	7	0.05
22.	Contraceptive-methd	1473	9	3	no	5	0.15
23.	Boston-housing	506	12(13)	3(cont)	no	4	0
24.	Wine	178	13	3	no	1	0
25.	Attitude-smoking$^+$	2855	13	3	no	1	0
26.	Waveform$^+$	3600	21	3	no	1	0.15
27.	Thyroid$^+$	7200	21	3	no	3	0.20
28.	StatLog-DNA$^+$	3186	60	3	no	3	0.05
29.	Car	2782	6	4	no	3	0.05
30.	StatLog-vehicle	846	18	4	no	6	0
31.	Soybean-small	47	35	4	no	1	0
32.	Nursery	12960	8	4(5)	no	4	0.05
33.	StatLog-satimage$^+$	6435	36	6	no	6	0.20
34.	Glass	214	9(10)	7	no	2	0.25
35.	Zoo	101	17(18)	7	no	1	0
36.	StatLog-image-seg	2310	19	7	no	6	0
37.	Ecoli	336	7	8	no	2	0.10
38.	LED-display$^+$	6000	7	10	no	1	0.45
39.	Yeast	1484	8(9)	10	no	6	0.05
40.	Pendigit	10992	16	10	no	6	0

(i) Total number of instances, i.e. examples, data points, observations (given number of instances). Note: the number of instances used is larger than the given number of instances when we expand those "don't care" kind of attributes in some data sets; (ii) Number of features used, i.e. dimensions, attributes (total number of features given); (iii) Number of classes (assuming a discrete class variable); (iv) Missing features;

$^+$ Accuracy measured from the given training and test set instead of 10-fold validation (for large data cases with test set containing at least 1,000 samples);

Note: Data from the Attitudes Towards Smoking Legislation Survey—Metropolitan Toronto 1988, which was funded by NHRDP (Health and Welfare Canada), were collected by the Institute for Social Research at York University for Dr. Linda Pederson and Dr. Shelley Bull.

technique), kNN (k-Nearest Neighbors), C4.5 (decision tree) and SVM-Poly (Support Vector Machine using polynomial kernel). Their experiments used a single run of the 10-fold stratified cross validation on 35 data sets from the UCI Machine Learning Repository. Only the average classification accuracies and the data retention rates (defined as a ratio of *the number of prototypes learned* over *the number of training instances*) were reported and no CPU times recorded. Application of the above five algorithms in the reported 35 data sets resulted in the following ranking in terms of the average classification accuracy: SVM-Poly (0.878), kNN (0.875), ICPL (0.863), RT3 (0.861) and C4.5(0.842). Both setups of the proposed classifier model (RM-InvSig and RM-InvSig-S) will be compared with these five algorithms using those 28 data sets with known and comparable settings. The reference (Lam et al., 2002) will be denoted as Ref-I for convenient reading.

Ref-II: In Lim et al. (2000), a total of thirty-three old and new classification algorithms (twenty-two belonged to the decision trees type including the C4.5, nine belonged to the statistical type including the Nearest Neighbor and two belonged to the neural networks type including the RBF) were compared using 16 data sets from the UCI Machine Learning Repository. Extensive experiments were performed on these data sets and a comprehensive analysis was presented regarding the *error rates*, *ranks*, *training time*, *size of trees*, and *scalability* aspects for the compared algorithms. For the reported results, most data sets used the average 10-fold validation error rates (a single run) except for those six data sets [4] marked with a '+' that used the given test set to compute the error rates. Their results placed a statistical spline-based algorithm (acronymed as POL) at the top in terms of average classification accuracy even though it was ranked third last in terms of training time. The reference (Lim et al., 2000) will be denoted as Ref-II for convenient reading and all 16 data sets will be used in the comparison.

Ref-III: Recently, Precup and Utgoff (2004) compared their proposed CLEF (a Constructive Learning method) algorithm with five existing algorithms including C4.5, SVM-RBF (using RBF kernel) and DNC(Dynamic Node Creation, a constructive neural network). Except for the Monk-2 data set that used the training and test sets provided, all accuracy results on the 20 data sets were reported using 10-fold stratified cross validation (a single run). In terms of average classification accuracy, the following ranking was established: CLEF (76.25%), SVM-RBF (75.81%), C4.5 (70.03%), DNC (69.39%), Φ-RT (67.51%) and Φ-DNC (65.45%). As some of the data sets are either different from that in UCI but with same name or different class grouping being used, only 9 data sets are found to be common to those in Lam et al. (2002) and Lim et al. (2000) and they are also listed for comparison. The reference (Precup & Utgoff, 2004) will be denoted as Ref-III for convenient reading.

Ref-IV: In Li et al. (2004), extensive experiments based on 10-fold cross-validation were performed on 40 data sets for a few instance-based algorithms, namely DeEPs (Decision-making by Emerging Patterns), kNN and C5.0. The study revealed that DeEPs outperformed kNN and C5.0 for majority of data sets but with slower computing speed than those of kNN and C5.0. A few other classifiers are also compared, but due to different partitioning, they are not listed here for comparison. Among the 40 data sets, only those

[4] According to Lim et al. (2000), the following six data sets are partitioned into two sets namely the training set and the test set for experimentation: Attitude-smoking, Waveform, Thyroid, StatLog-DNA, StatLog-sat-image and LED-display. These data sets are considered to be large (Lim et al., 2000) as their sizes are much larger than 1000 and the test set sizes are all at least 1000. In these six cases, the error rates are estimated from the test sets and these error rates are compared with those in Lim et al. (2000).

19 sets in common to our 40 data sets were used for comparison. The reference (Li et al., 2004) will be denoted as Ref-IV for convenient reading.

6.3. Results

(i) Comparing performance of RM-InvSig-S with that of RM-InvSig

Table 4 lists the accuracy statistics for RM-InvSig and RM-InvSig-S. The average test accuracies of RM-InvSig-S are plotted relative to those of RM-InvSig (accuracies of RM-InvSig-S minus accuracies of RM-InvSig) in Fig. 7 (average accuracies are taken from 10 runs of 10-fold experiments). Those points indicated by ★ are data sets that encounter Matlab's (The MathWorks, 2003) reciprocal condition warning (RCOND Warning) during parameter estimation. Here, it is seen that numerical ill-conditioning does not influence much the accuracy of RM-InvSig-S relative to RM-InvSig.

The data sets are marked with their corresponding ρ settings as shown in Fig. 7. The accuracy improvement ranges from near zero to about 4.3% for RM-InvSig-S over RM-InvSig. Figure 8(a)–(b) show the lower (minimum) and upper (maximum) performance ranges of the average relative accuracy from the 10 runs of 10-fold cross-validation. A general performance improvement trend is observed from these plots for instances where the training adopted the feature scaling-space method.

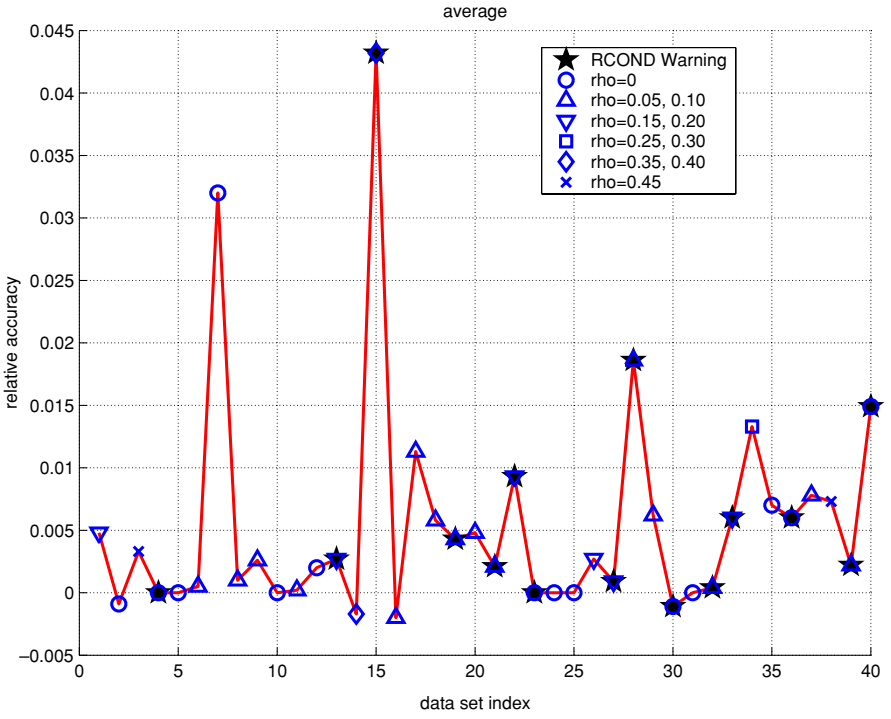
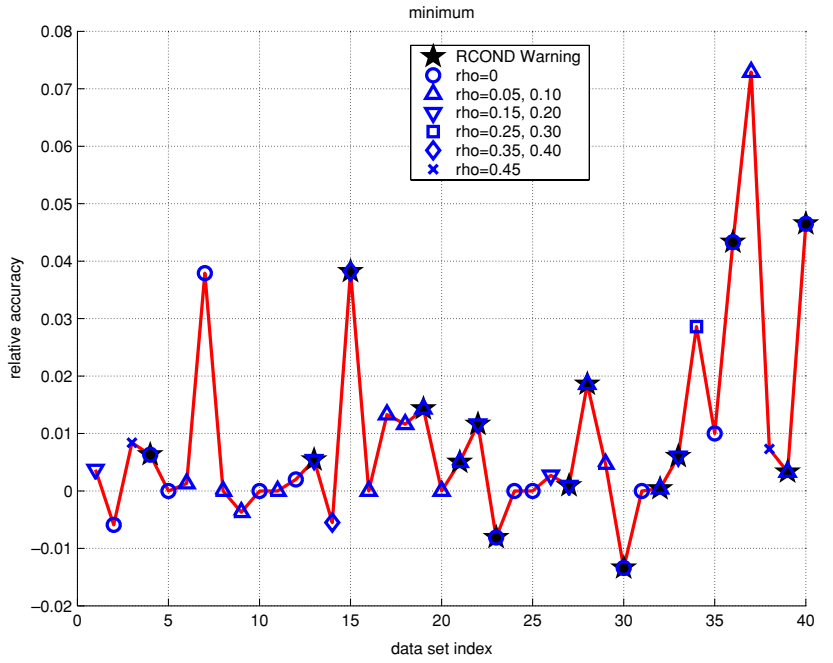


Fig. 7 Average relative accuracy of RM-InvSig-S with respect to RM-InvSig. The accuracy improvement ranges from near zero to about 4.3% for RM-InvSig-S over RM-InvSig. Apart from the relative accuracy results, the data sets are marked with their corresponding ρ settings in feature scaling-space learning

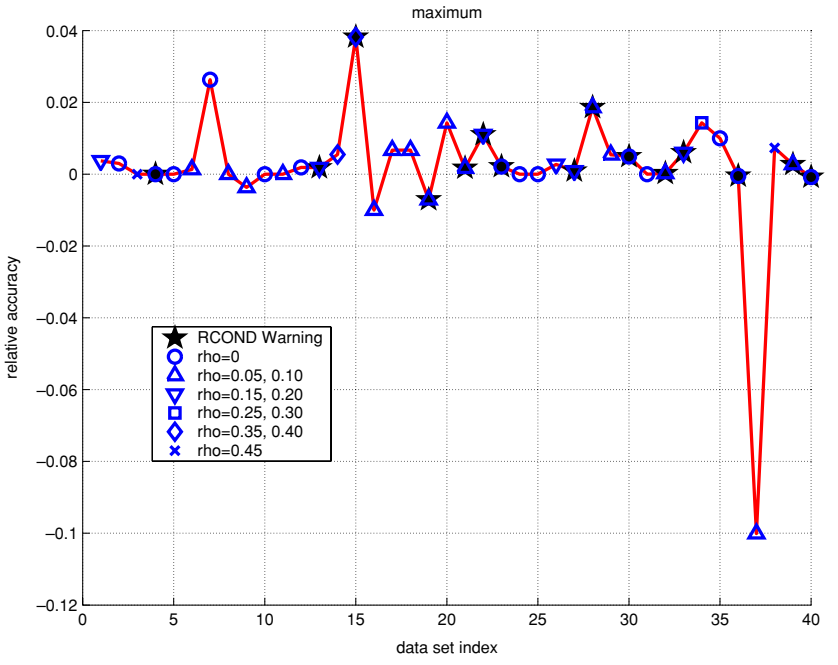
Table 4 Classification accuracy statistics for RM-InvSig and RM-InvSig-S using ten runs of 10-fold cross validation

		RM-InvSig				RM-InvSig-S			
No	Name	Min	Ave	Max	Std	Min	Ave	Max	Std
1	Shuttle-l-contr	0.9519	0.9552	0.9593	0.0026	0.9556	0.9600	0.9630	0.0028
2	BUPA-liver	0.7294	0.7391	0.7529	0.0073	0.7235	0.7382	0.7559	0.0090
3	Monk-1	0.7833	0.8142	0.8333	0.0149	0.7917	0.8175	0.8333	0.0142
4	Monk-2	0.7000	0.7550	0.7875	0.0289	0.7063	0.7550	0.7875	0.0266
5	Monk-3	0.8833	0.9033	0.9167	0.0093	0.8833	0.9033	0.9167	0.0093
6	Pima-diabetes	0.7645	0.7695	0.7750	0.0035	0.7658	0.7700	0.7763	0.0036
7	Tic-tac-toe	0.9453	0.9515	0.9579	0.0041	0.9832	0.9835	0.9842	0.0005
8	Breast-cancer-W	0.9687	0.9721	0.9776	0.0028	0.9687	0.9731	0.9776	0.0028
9	StatLog-heart	0.8222	0.8430	0.8630	0.0120	0.8185	0.8456	0.8593	0.0122
10	Credit-app	0.8625	0.8641	0.8672	0.0018	0.8625	0.8641	0.8672	0.0018
11	Votes	0.9500	0.9531	0.9548	0.0019	0.9500	0.9533	0.9548	0.0019
12	Mushroom	0.9957	0.9960	0.9963	0.0001	0.9977	0.9980	0.9982	0.0002
13	Wdbc	0.9482	0.9566	0.9607	0.0036	0.9536	0.9593	0.9625	0.0029
14	Wpbc	0.7944	0.8128	0.8389	0.0136	0.7889	0.8111	0.8444	0.0138
15	Ionosphere	0.8412	0.8568	0.8794	0.0122	0.8794	0.9000	0.9176	0.0116
16	Sonar	0.7250	0.7595	0.7900	0.0188	0.7250	0.7575	0.7800	0.0183
17	Iris	0.9667	0.9727	0.9800	0.0036	0.9800	0.9840	0.9867	0.0033
18	Balance-scale	0.8767	0.8900	0.8950	0.0048	0.8883	0.8958	0.9017	0.0037
19	Teaching-assist	0.5286	0.5557	0.5857	0.0177	0.5429	0.5600	0.5786	0.0112
20	New-thyroid	0.9238	0.9290	0.9333	0.0026	0.9238	0.9338	0.9476	0.0069
21	Abalone	0.6505	0.6611	0.6675	0.0061	0.6555	0.6632	0.6692	0.0035
22	Contraceptive-mthd	0.5418	0.5486	0.5527	0.0033	0.5534	0.5579	0.5637	0.0034
23	Boston-housing	0.7612	0.7765	0.7857	0.0084	0.7531	0.7765	0.7878	0.0097
24	Wine	0.9812	0.9869	0.9875	0.0019	0.9812	0.9869	0.9875	0.0019
25	Attitude-smoking$^+$	0.6950	0.6950	0.6950	0.0000	0.6950	0.6950	0.6950	0.0000
26	Waveform$^+$	0.8103	0.8103	0.8103	0.0000	0.8130	0.8130	0.8130	0.0000
27	Thyroid$^+$	0.9390	0.9390	0.9390	0.0000	0.9399	0.9399	0.9399	0.0000
28	StatLog-DNA$^+$	0.9064	0.9064	0.9064	0.0000	0.9250	0.9250	0.9250	0.0000
29	Car	0.8419	0.8454	0.8498	0.0023	0.8466	0.8516	0.8552	0.0025
30	StatLog-vehicle	0.7500	0.7962	0.8183	0.0201	0.7366	0.7951	0.8232	0.0256
31	Soyabean-small	0.9500	0.9500	0.9500	0.0000	0.9500	0.9500	0.9500	0.0000
32	Nusery	0.9062	0.9067	0.9073	0.0003	0.9066	0.9071	0.9075	0.0003
33	StatLog-satimage$^+$	0.8445	0.8445	0.8445	0.0000	0.8505	0.8505	0.8505	0.0000
34	Glass	0.6190	0.6457	0.6619	0.0146	0.6476	0.6590	0.6762	0.0083
35	Zoo	0.9400	0.9560	0.9700	0.0092	0.9500	0.9630	0.9800	0.0100
36	StatLog-image-seg	0.9013	0.9413	0.9494	0.0135	0.9446	0.9473	0.9489	0.0012
37	Ecoli	0.7847	0.8610	0.9789	0.0517	0.8576	0.8688	0.8788	0.0064
38	Led-display$^+$	0.7335	0.7335	0.7335	0.0000	0.7408	0.7408	0.7408	0.0000
39	Yeast	0.5899	0.5961	0.6014	0.0040	0.5932	0.5983	0.6041	0.0028
40	Pendigit	0.8285	0.9161	0.9504	0.0382	0.8750	0.9310	0.9496	0.0250
	Average	0.8234	**0.8391**	0.8516	0.0085	0.8326	**0.8446**	0.8535	0.0064

$^+$: Accuracy measured from the given training and test set instead of 10-fold validation.
RM-InvSig: Reciprocal-sigmoid model expanded by RM2 as shown in (11).
RM-InvSig-S: Scaling-space learning of reciprocal-sigmoid model expanded by RM2 as shown in (11).



(a)



(b)

Fig. 8 Average minimum and maximum relative accuracies of RM-InvSig-S with respect to original RM-InvSig's minimum and maximum accuracies. These two plots supplement the average relative accuracy in Fig.7 in terms of the lower and upper performance ranges

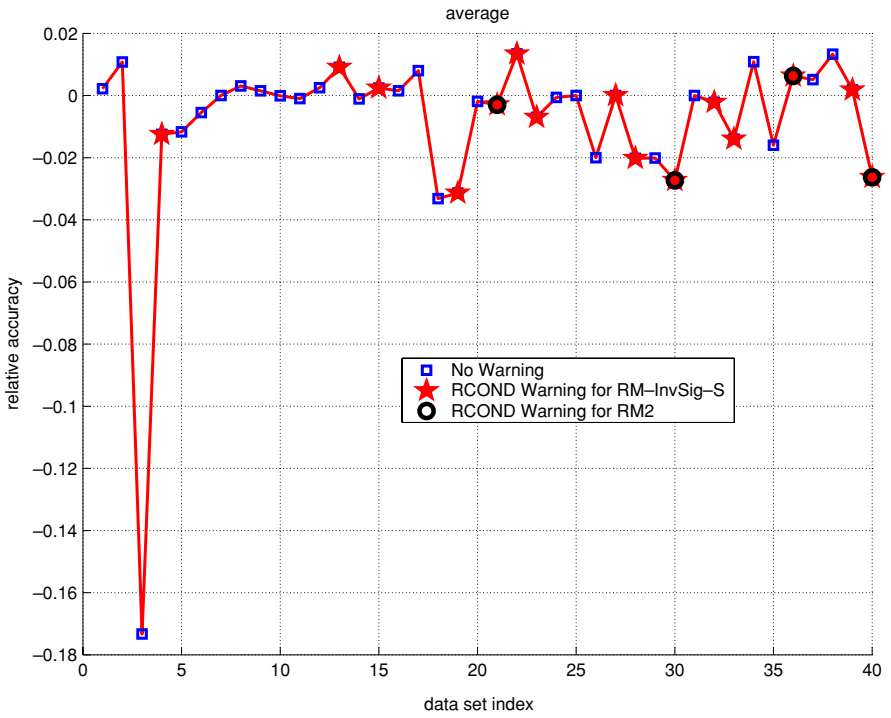


Fig. 9 Average relative accuracy of RM-InvSig-S with respect to RM2. These results show that the performance of RM-InvSig-S is comparable to RM2 and may degrade in some cases at similar order settings

(ii) Comparing performance of RM-InvSig-S with that of RM2

The average test accuracies of the RM-InvSig-S with respect to the original RM2 [5] at similar model-order settings are shown as relative values (% difference between accuracies of RM-InvSig-S and RM2) in Fig. 9. It was found that out of the total 40 data sets experimented, 15 data sets showed better accuracy for RM-InvSig-S than that of RM2. Four data sets had equal performances between RM-InvSig-S and RM2. Twenty-one data sets had poorer performance for RM-InvSig-S against that of RM2. The average accuracy (for all 40 data sets) of RM-InvSig-S was found to be 0.84% lower than that of RM2. These results show that the performance of RM-InvSig-S is comparable to RM2 and may degrade slightly in some cases at similar order settings.

Table 5 lists the number of parameters used in RM2 and re-tuned RM-InvSig and RM-InvSig-S for those that used high order models.[6] Here, it is seen that the re-tuning led to accuracy improvement with respect to RM2 in some cases. The underlined numbers in the third-last column in Table 5 are cases where RM-InvSig (and RM-InvSig-S which used

[5]RM2 is the model given by (11) without using the reciprocal-sigmoid as basis function, i.e. obtained from replacing $\phi(x)$ by x in (11).

[6]For those cases where RM2 had selected a high order model for classification, a re-tuning process was performed on RM-InvSig and RM-InvSig-S by cross-validation using only the training data to choose a new model order such that the best validation performance was obtained. The selected model order was then used to compute the accuracies for all the 10 runs of 10-fold tests. The purpose of this re-tuning is to check if different performance can be obtained by introducing the reciprocal-sigmoid as basis function into RM2.

Table 5 Number of parameters for re-tuned RM-InvSig, RM-InvSig-S and RM2

		RM2				RM-InvSig				RM-InvSig-S	
Index	Data set name	acc	Std	r	No. para	acc	Std	r	No. para	acc	Std
3	Monk-1	0.9908	0.0079	4	43	0.8142	0.0149	4	43	0.8175	0.0142
4	Monk-2	0.7675	0.0195	7	79	0.7737	0.0426	9	103	0.7644	0.0188
17	Iris	0.9760	0.0033	4	87	0.9787	0.0040	2	39	0.9767	0.0033
19	Teaching-assist	0.5914	0.0156	9	258	0.5543	0.0125	6	168	0.5521	0.0207
20	New-thyroid	0.9357	0.0093	4	108	0.9548	0.0105	6	168	0.9595	0.0065
21	Abalone	0.6662	0.0013	7	315	0.6667	0.0014	4	171	0.6628	0.0010
22	Contraceptive-mthd	0.5445	0.0046	5	246	0.5560	0.0037	3	138	0.5588	0.0044
23	Boston-housing	0.7835	0.0046	4	255	0.7765	0.0084	4	255	0.7765	0.0097
30	StatLog-vehicle	0.8224	0.0068	6	796	0.8143	0.0049	4	508	0.8182	0.0048
32	Nusery	0.9093	0.0004	4	228	0.9082	0.0003	2	100	0.9074	0.0003
33	StatLog-satimage	0.8645	0	6	2382	0.8555	0	4	1518	0.8425	0
36	StatLog-image-seg	0.9410	0.0008	6	1470	0.9406	0.0015	5	1204	0.9420	0.0018
39	Yeast	0.5965	0.0033	6	890	0.6000	0.0027	4	570	0.5980	0.0032
40	Pendigit	0.9573	0.0005	6	1770	0.9422	0.0006	3	810	0.9422	0.0005

RM2: RM2 model without adopting the reciprocal-sigmoid basis, i.e. obtained from replacing ϕ by x in (11).
RM-InvSig: Reciprocal-sigmoid model expanded by RM2 as shown in (11).

the same re-tuned model-orders as those in RM-InvSig) used fewer parameters than RM2 but with comparable accuracy performance. It is seen that for many cases, the number of parameters used can be significantly fewer for RM-InvSig than that of RM2 while maintaining comparable test accuracy.

(iii) Comparing performance of RM-InvSig-S with a kernel ridge regression algorithm

The performance of RM-InvSig-S is next compared to a kernel ridge regression algorithm (see e.g. Shawe-Taylor & Cristianini, 2004) adopting the following kernel:

$$Ker(\boldsymbol{x}, \boldsymbol{z}) = (1 + \phi(\boldsymbol{x}) \cdot \phi(\boldsymbol{z}))^l, \tag{18}$$

where $\phi(x_1, \ldots, x_l) = [(1 + e^{-x_1}), \ldots, (1 + e^{-x_l})]$ and l is the input dimension. It is noted that this kernel adopts a full multivariate polynomial expansion and its terms grow exponentially with respect to the order and number of inputs (Bishop, 1995; Schürmann, 1996). Although this kernel ridge regression algorithm has a much larger number of expansion terms comparing to our case, which is linear with respect to the order or number of inputs (see Section 3.2, $K = 1 + l(2r - 1)$ for RM2), the use of a similar reciprocal-sigmoid basis function provides an interesting ground for performance comparison.

Table 6 shows the results from 10 runs of 10-fold experiments, which are performed under similar conditions ($b = 0.0001$ and similar 10-fold partitioning used in previous experiments) for the full polynomial kernel ridge regression for two settings: 3rd-order (FP-InvSig-3) and lth-order (FP-InvSig-l). Here, it is seen that more than half of the experiments for FP-InvSig-l encountered either memory problem or numerical ill-conditioning. Comparing the individual data sets with those from RM-InvSig-S, Table 6 highlights those superior performance cases by bold characters. It can be seen that 13 data sets from FP-InvSig-3 and 4 data sets from FP-InvSig-l have equal or better average accuracy than RM-InvSig-S. The overall average

Table 6 Classification accuracy statistics for FP-InvSig-3 and FP-InvSig-*l* using ten runs of 10-fold cross validation

		FP-InvSig-3				FP-InvSig-*l*			
No	Name	Min	Ave	Max	Std	Min	Ave	Max	Std
1	Shuttle-l-contr	0.9815	**0.9859**	0.9926	0.0036	0.9889	**0.9930**	0.9963	0.0026
2	BUPA-liver	0.7118	0.7279	0.7441	0.0114	0.6588	0.6885	0.7176	0.0172
3	Monk-1	0.8500	**0.8683**	0.8917	0.0133	0.7833	0.7967	0.8167	0.0107
4	Monk-2	0.6188	0.6706	0.6937	0.0205	0.6438	0.6769	0.7063	0.0200
5	Monk-3	0.8750	0.9025	0.9250	0.0154	0.6500	0.6983	0.7417	0.0258
6	Pima-diabetes	0.7395	0.7488	0.7632	0.0059	0.6105	0.6258	0.6487	0.0122
7	Tic-tac-toe	0.9832	**0.9835**	0.9842	0.0005	*	*	*	*
8	Breast-cancer-W	0.9209	0.9301	0.9403	0.0054	*	*	*	*
9	StatLog-heart	0.7037	0.7296	0.7519	0.0162	*	*	*	*
10	Credit-app	0.8172	0.8281	0.8375	0.0062	*	*	*	*
11	Votes	0.8738	0.8848	0.8976	0.0075	*	*	*	*
12	Mushroom	*	*	*	*	*	*	*	*
13	Wdbc	0.8893	0.8998	0.9161	0.0084	*	*	*	*
14	Wpbc	0.7056	0.7361	0.7556	0.0156	*	*	*	*
15	Ionosphere	0.7441	0.7624	0.7824	0.0121	*	*	*	*
16	Sonar	0.7950	**0.8255**	0.8450	0.0154	*	*	*	*
17	Iris	0.9733	0.9760	0.9800	0.0033	0.9667	0.9733	0.9800	0.0030
18	Balance-scale	0.9050	**0.9092**	0.9117	0.0024	0.9000	**0.9042**	0.9083	0.0026
19	Teaching-assist	0.5214	0.5586	0.6071	0.0228	0.4714	0.5300	0.5714	0.0263
20	New-thyroid	0.9286	**0.9457**	0.9619	0.0088	0.9286	**0.9362**	0.9524	0.0065
21	Abalone	*	*	*	*	*	*	*	*
22	Contraceptive-mthd	0.5308	0.5453	0.5548	0.0086	*	*	*	*
23	Boston-housing	0.7735	**0.7820**	0.7939	0.0060	*	*	*	*
24	Wine	0.9438	0.9575	0.9688	0.0088	*	*	*	*
25	Attitude-smoking$^+$	0.0377	0.4233	0.8993	0.3642	*	*	*	*
26	Waveform$^+$	0.5230	0.5583	0.5860	0.0228	*	*	*	*
27	Thyroid$^+$	0.2599	0.7016	0.9877	0.2925	*	*	*	*
28	StatLog-DNA$^+$	*	*	*	*	*	*	*	*
29	Car	0.9347	**0.9365**	0.9390	0.0012	0.9718	**0.9731**	0.9747	0.0010
30	StatLog-vehicle	0.8171	**0.8311**	0.8415	0.0081	*	*	*	*
31	Soyabean-small	0.9500	**0.9500**	0.9500	0.0000	*	*	*	*
32	Nusery	*	*	*	*	*	*	*	*
33	StatLog-satimage$^+$	*	*	*	*	*	*	*	*
34	Glass	0.6571	**0.6724**	0.6857	0.0087	0.3952	0.4276	0.4619	0.0196
35	Zoo	0.9400	**0.9660**	0.9800	0.0120	*	*	*	*
36	StatLog-image-seg	0.9312	0.9369	0.9411	0.0029	*	*	*	*
37	Ecoli	0.8167	0.8563	0.9456	0.0345	0.7848	0.7952	0.8061	0.0079
38	Led-display$^+$	0.6449	**0.7408**	0.8303	0.0586	*	*	*	*
39	Yeast	0.5959	0.5981	0.6007	0.0019	0.5669	0.5729	0.5784	0.0037
40	Pendigit	*	*	*	*	*	*	*	*
	Average	0.7616	0.8038	0.8437	0.0302	0.7372	0.7566	0.7758	0.0114

$^+$: Accuracy measured from the given training and test set instead of 10-fold validation.
*: Either matrix size too large to be computed or singularity of matrix encountered.
Bold numbers: Have either equal or better average accuracy than RM-InvSig-S.
FP-InvSig-3: Reciprocal-sigmoid kernel expanded by 3rd-order full polynomials.
FP-InvSig-*l*: Reciprocal-sigmoid kernel expanded by *l*th-order full polynomials (*l* is the input feature dimension).

accuracies for FP-InvSig-3 and FP-InvSig-*l* are seen to be lower than that of RM-InvSig and RM-InvSig-S.

(iv) Comparing performance of RM-InvSig-S with a GLM

Since the proposed model (11) can be related to logistic regression (12)–(13) in GLM, a comparison between (11) and (12)–(13) would be interesting. Two versions of GLM are considered: 1. a GLM which adopted the logit link with binomial distributions (GLM),[7] and 2. the logit link GLM which was trained using scaling-space inputs (GLM-S). The purpose here is to check whether scaling-space is beneficial to GLM. Both GLM and GLM-S were experimented using GLMLAB from Dunn (1999, 2000) with maximum number of iterations fixed at 20 (default). Table 7 lists the results from 10 runs of 10-fold experiments, which were performed under similar partitioning conditions as in previous experiments. Figure 10(a) shows the average relative accuracies of RM-InvSig and RM-InvSig-S with respect to GLM, and Fig. 10(b) shows the average relative accuracy of GLM-S with respect to GLM. These results show that both RM-InvSig and RM-InvSig-S outperform the GLM in many data sets and that scaling-space inputs do not benefit GLM in many cases.

(v) Comparing performance of RM-InvSig-S with others in literature

Apart from above comparisons, the average accuracy of RM-InvSig-S is next compared with a few other results which reported extensive experiments. The comparison will be limited to only non-aggregate type of classifiers since both RM-InvSig and RM-InvSig-S belong to this type. Here it is noted that Ref-I through Ref-IV used only a single run of 10-fold cross-validation whereas our experiments on RM-InvSig and RM-InvSig-S used ten runs of 10-fold cross-validation. Table 8 shows the minimum, average and maximum accuracies of RM-InvSig and RM-InvSig-S from the 10 runs of 10-fold cross-validation tabulated in ranking order with respect to those reported average accuracies over those similar data sets. These results suggest that RM-InvSig-S have comparable accuracy with the top classifiers found in the literature.

(vi) Effect of data attributes on the proposed method

Having seen the accuracy aspect, the effect of sample size and feature dimension on RM-InvSig and RM-InvSig-S will be studied. Figure 11(a) shows the distribution of average accuracies over the total number of samples and the number of features of the data sets. From the plot on top, no significant trend of accuracy distribution according to sample size can be found. For the bottom plot, the maximum accuracy appears to have a mild decreasing trend over the feature dimension.

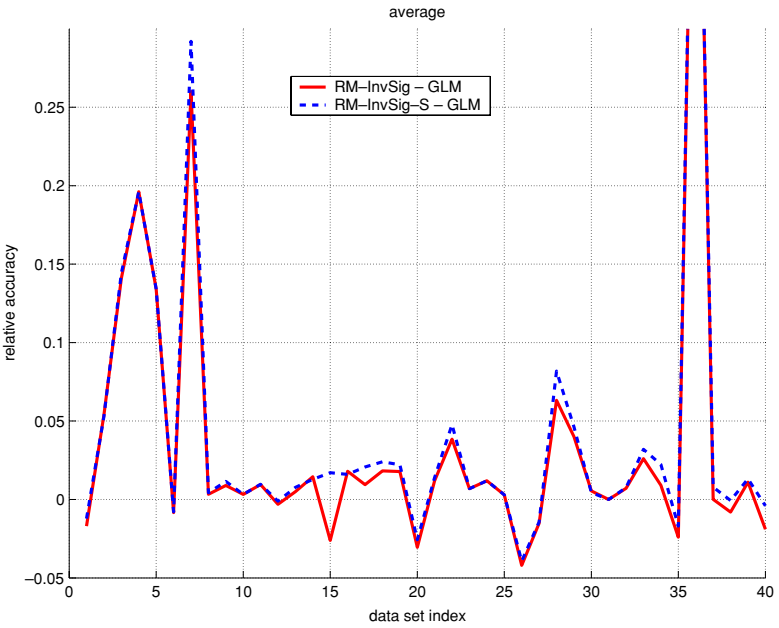
Figure 11(b) shows the relative accuracy (average accuracy of RM-InvSig-S minus average accuracy of RM-InvSig) plotted over the ratio between sample number and feature dimension. Here, it is seen that notable improvement of RM-InvSig-S over RM-InvSig oc-

[7] In order to observe the performance under similar ground with that of the reciprocal-sigmoid model, the logistic regression model was implemented to learn multiple two-class problems inlined with (2) and (17) for multi-category cases. While we limit our study within this multiple two-class platform for comparable empirical error evaluations, the interested reader is referred to Agresti (2002) for an extended literature regarding multinomial response models.

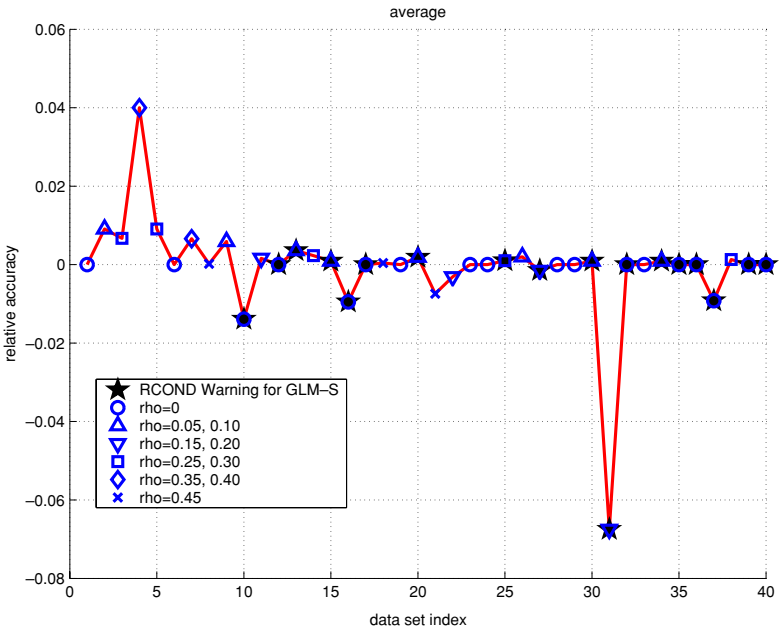
Table 7 Classification accuracy statistics for GLM and GLM-S using ten runs of 10-fold cross validation

		GLM				GLM-S			
No	Name	Min	Ave	Max	Std	Min	Ave	Max	Std
1	Shuttle-l-contr	0.9667	**0.9722**	0.9778	0.0030	0.9667	**0.9722**	0.9778	0.0030
2	BUPA-liver	0.6794	0.6853	0.6912	0.0046	0.6765	0.6944	0.7088	0.0082
3	Monk-1	0.6167	0.6733	0.7000	0.0229	0.6250	0.6800	0.7000	0.0201
4	Monk-2	0.5437	0.5588	0.5813	0.0126	0.5687	0.5988	0.6375	0.0172
5	Monk-3	0.7083	0.7692	0.7917	0.0233	0.7583	0.7783	0.8000	0.0130
6	Pima-diabetes	0.7737	**0.7779**	0.7816	0.0028	0.7737	**0.7779**	0.7816	0.0028
7	Tic-tac-toe	0.6853	0.6915	0.6989	0.0044	0.6905	0.6981	0.7074	0.0052
8	Breast-cancer-W	0.9642	0.9688	0.9731	0.0027	0.9642	0.9690	0.9746	0.0035
9	StatLog-heart	0.8185	0.8341	0.8481	0.0076	0.8333	0.8400	0.8519	0.0066
10	Credit-app	0.8562	0.8609	0.8672	0.0036	0.8219	0.8470	0.8625	0.0169
11	Votes	0.9333	0.9436	0.9595	0.0069	0.9381	0.9452	0.9619	0.0069
12	Mushroom	0.9988	**0.9991**	0.9993	0.0002	0.9988	**0.9991**	0.9993	0.0002
13	Wdbc	0.9446	0.9516	0.9589	0.0053	0.9464	0.9552	0.9643	0.0050
14	Wpbc	0.7667	0.7983	0.8278	0.0167	0.7778	0.8006	0.8333	0.0160
15	Ionosphere	0.8735	0.8829	0.8971	0.0060	0.8647	0.8838	0.9000	0.0094
16	Sonar	0.7150	0.7415	0.7750	0.0172	0.6650	0.7320	0.7650	0.0252
17	Iris	0.9467	0.9633	0.9733	0.0068	0.9467	0.9633	0.9733	0.0068
18	Balance-scale	0.8633	0.8718	0.8767	0.0042	0.8617	0.8722	0.8783	0.0048
19	Teaching-assist	0.5143	0.5379	0.5500	0.0101	0.5143	0.5379	0.5500	0.0101
20	New-thyroid	0.9571	**0.9595**	0.9714	0.0044	0.9571	**0.9614**	0.9762	0.0058
21	Abalone	0.6474	0.6492	0.6512	0.0012	0.6401	0.6418	0.6442	0.0013
22	Contraceptive-mthd	0.5048	0.5102	0.5240	0.0052	0.5027	0.5071	0.5110	0.0026
23	Boston-housing	0.7592	0.7696	0.7796	0.0063	0.7592	0.7696	0.7796	0.0063
24	Wine	0.9688	0.9750	0.9812	0.0028	0.9688	0.9750	0.9812	0.0028
25	Attitude-smoking$^+$	0.6920	0.6920	0.6920	0.0000	0.6930	0.6930	0.6930	0.0000
26	Waveform$^+$	0.8523	**0.8523**	0.8523	0.0000	0.8543	**0.8543**	0.8543	0.0000
27	Thyroid$^+$	0.9542	**0.9542**	0.9542	0.0000	0.9527	**0.9527**	0.9527	0.0000
28	StatLog-DNA$^+$	0.8432	0.8432	0.8432	0.0000	0.8432	0.8432	0.8432	0.0000
29	Car	0.8025	0.8053	0.8079	0.0015	0.8025	0.8053	0.8079	0.0015
30	StatLog-vehicle	0.7829	0.7907	0.8000	0.0045	0.7829	0.7916	0.8012	0.0057
31	Soyabean-small	0.9500	**0.9500**	0.9500	0.0000	0.8500*	0.8825*	0.9250*	0.0225*
32	Nusery	0.8988	0.8996	0.9007	0.0005	0.8988	0.8996	0.9007	0.0005
33	StatLog-satimage$^+$	0.8185	0.8185	0.8185	0.0000	0.8185	0.8185	0.8185	0.0000
34	Glass	0.6190	0.6367	0.6524	0.0093	0.6143	0.6376	0.6524	0.0125
35	Zoo	0.9500	**0.9800**	0.9900	0.0134	0.9500	**0.9800**	0.9900	0.0134
36	StatLog-image-seg	*	*	*	*	*	*	*	*
37	Ecoli	0.8515	0.8609	0.8667	0.0046	0.7917	0.8517	0.9094	0.0368
38	Led-display$^+$	0.7415	**0.7415**	0.7415	0.0000	0.7428	**0.7428**	0.7428	0.0000
39	Yeast	0.5818	0.5851	0.5892	0.0020	0.5818	0.5851	0.5892	0.0020
40	Pendigit	0.9343	**0.9351**	0.9357	0.0004	0.9343	**0.9351**	0.9357	0.0004
	Average	0.7885	0.8015	0.8149	0.0074	0.7848	0.8010	0.8176	0.0093

$^+$: Accuracy measured from the given training and test set instead of 10-fold validation.
*: Matrix too ill-conditioned and results may not be reliable.
Bold numbers: Have either equal or better average accuracy than RM-InvSig-S.
GLM: GLM adopting a logit link as in (12).
GLM-S: Scaling-space learning of GLM.

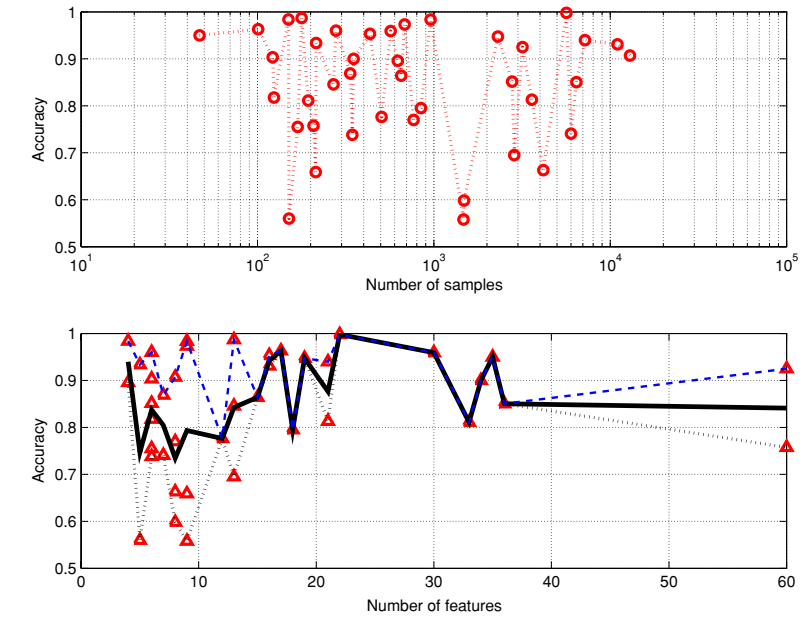


(a)

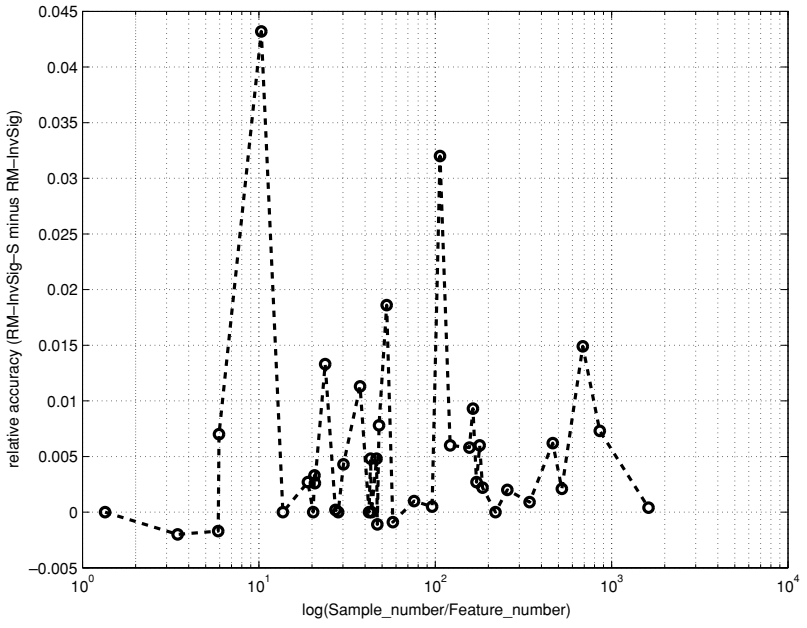


(b)

Fig. 10 Average relative accuracy: (a) RM-InvSig (solid) and RM-InvSig-S (dashed) with respect to GLM. The average relative accuracy at data 36 should be ignored as it was obtained based on comparison with a poorly learned GLM due to matrix ill-conditioning, (b) GLM-S with respect to GLM. These results show that both RM-InvSig and RM-InvSig-S outperform the GLM in many data sets and that scaling-space inputs do not benefit GLM in many cases



(a)



(b)

Fig. 11 Distribution of average accuracies: (a) accuracy of RM-InvSig-S versus sample size and feature dimension: The dark line in the lower plot corresponds to the mean accuracy obtained from the multiple data sets having the same number of features. The dashed and the dotted lines are the maximum and minimum accuracies respectively; (b) average relative accuracy of RM-InvSig-S with respect to RM-InvSig versus the ratio of sample size over feature dimension. These plots show the relationship between the accuracy and the sample/feature sizes

Table 8 Summary of average accuracy in rank order with respect to Ref-I, Ref-II, Ref-III and Ref-IV

Ref-I	Ref-II	Ref-III	Ref-IV
RM-InvSig-S-max(0.889)	RM-InvSig-S-max(0.812)	RM-InvSig-S-max(0.902)	RM-InvSig-S-max(0.896)
RM-InvSig-max(0.888)	RM-InvSig-max(0.809)	RM-InvSig-max(0.899)	RM-InvSig-max(0.891)
SVM-Poly(0.886)	**RM-InvSig-S-ave(0.805)**	**RM-InvSig-S-ave(0.893)**	**RM-InvSig-S-ave(0.888)**
kNN(0.883)	POL(0.805)	**RM-InvSig-ave(0.890)**	DeEPs(0.886)
ICPL(0.881)	**RM-InvSig-ave(0.802)**	CLEF(0.884)	**RM-InvSig-ave(0.880)**
RM-InvSig-S-ave(0.879)	LOG(0.796)	RM-InvSig-S-min(0.883)	RM-InvSig-S-min(0.874)
RT3(0.878)	RM-InvSig-S-min(0.795)	RM-InvSig-min(0.878)	C4.5(0.867)
RM-InvSig-ave(0.873)	MDA(0.793)	SVM-RBF(0.874)	RM-InvSig-min(0.863)
RM-InvSig-S-min(0.867)	QL0(0.792)	C4.5(0.834)	kNN(0.859)
RM-InvSig-min(0.856)	LDA(0.792)	DNC(0.808)	–
C4.5(0.831)	RM-InvSig-min(0.790)	Φ-DNC(0.791)	–
	QL1(0.789)	Φ-RT(0.780)	–
–	PDA(0.787)		–
–	⋮	–	–
–	QDA(0.699)	–	–
–	T1(0.646)	–	–

RM-InvSig: Reciprocal-sigmoid model expanded by RM2 as shown in (11).
RM-InvSig-S: Scaling-space learning of reciprocal-sigmoid model expanded by RM2 as shown in (11).
-min: Minimum accuracy from ten runs of 10-fold cross-validation.
-ave: Average accuracy from ten runs of 10-fold cross-validation.
-max: Maximum accuracy from ten runs of 10-fold cross-validation.

curs within the range of [6,900] according to the sample number over feature dimension ratio.

(vii) Decision landscapes

The classification capability of RM-InvSig-S can perhaps be inferred from its decision landscape of biometrics data fusion (a 2-class problem; Toh, 2003). A highly localized training may end-up in poor test results since the training data may not be globally representative (over-fitting). Conversely, under-fitting of training data may occur. Figures 12 and 13 show the genuine and imposter classes distribution with decision boundaries and decision landscapes corresponding to a two-layer Multilayer Perceptron (MLP) with 2 hidden nodes, a SVM using RBF kernel, a 6th-order RM2 and the proposed RM-InvSig-S (6th-order). Except for a small shift in contour position and orientation due to local training solution, the decision landscape for logistic regression in this particular case appears analogous to that of MLP and thus is not shown here. This phenomenon is understandable because the logistic regression is indeed a single perceptron unit. It is seen from these figures that RM-InvSig-S shares certain localization properties of the selected SVM-RBF and MLP network.[8]

[8]The localization capability depends on the size of the network and kernel units. Good application of SVM and MLP depends much on choice of model structures with balanced localization property. The chosen size and structure of MLP and SVM are according to their good performance in combining the two biometrics (Toh, 2003).

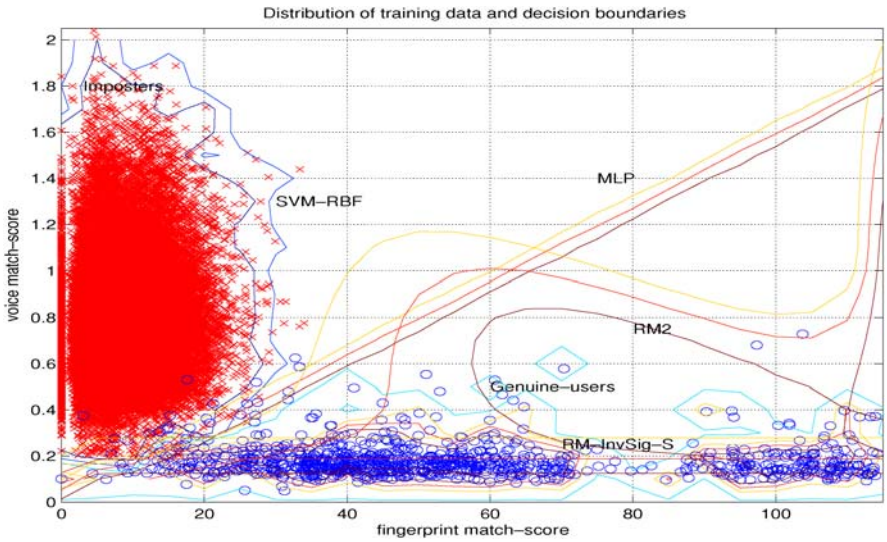


Fig. 12 Decision boundaries for MLP, SVM-RBF, RM2 and RM-InvSig-S. The decision contours were obtained at three sample thresholds (0.25, 0.5, 0.75) of normalized outputs. These plots show that RM-InvSig-S shares certain localization properties of the selected SVM-RBF and MLP network

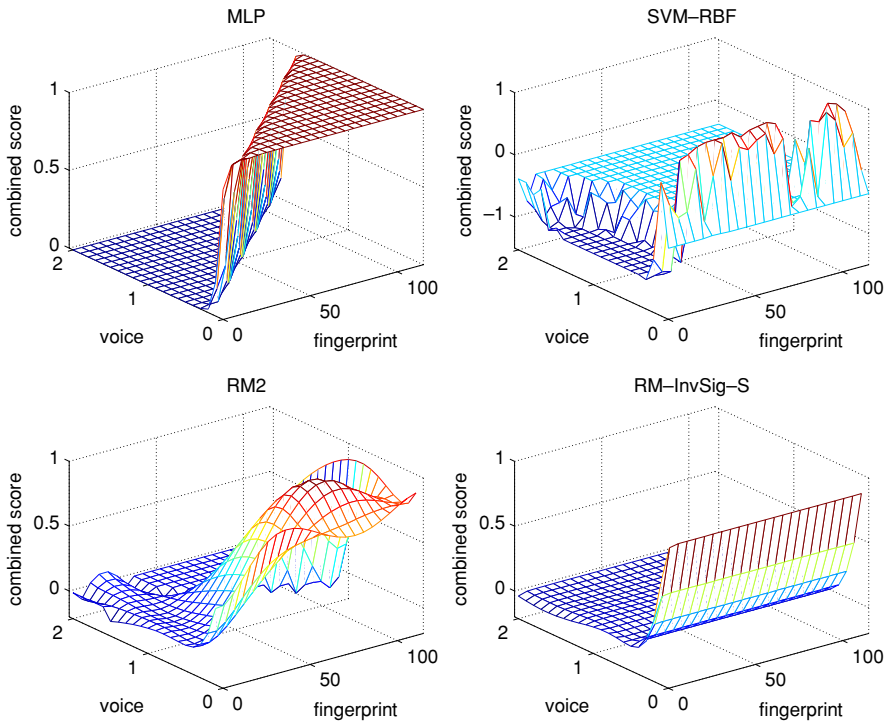


Fig. 13 Decision landscapes for MLP, SVM-RBF, RM2 and RM-InvSig-S. These 3-dimensional plots show that RM-InvSig-S shares certain localization properties of the selected SVM-RBF and MLP network

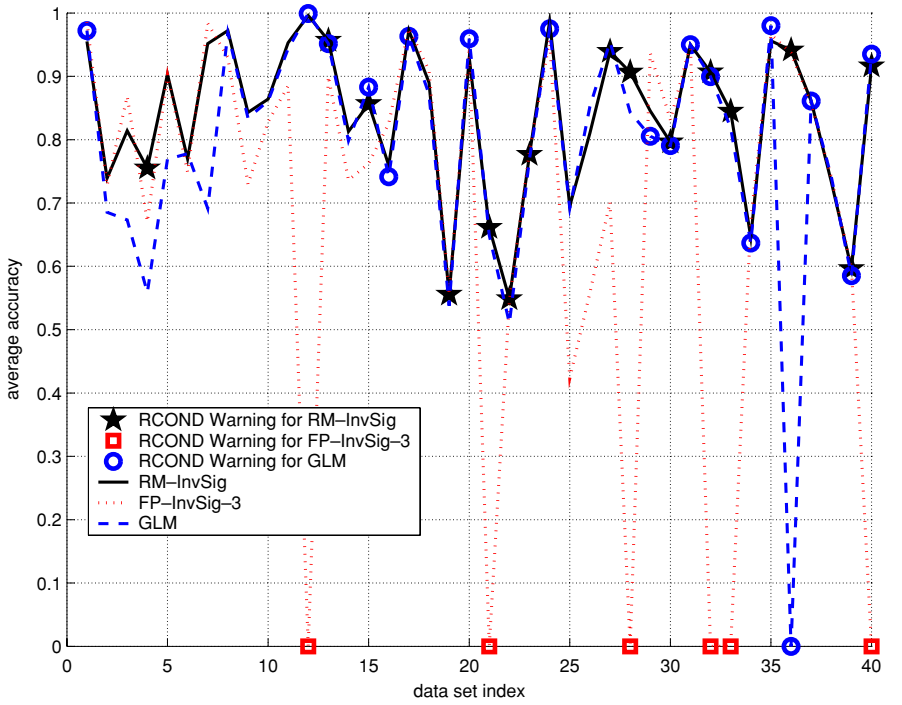


Fig. 14 Average accuracies of RM-InvSig, FP-InvSig-3 and GLM. This plot provides an overview of the performances of the baseline algorithms

6.4. Summary of results and comments

For an overview regarding the performances of the baseline algorithms, Fig. 14 shows the average accuracies of RM-InvSig, FP-InvSig-3 and GLM. The data sets which have better, equal and poorer performances for the scaling-space training over the original feature space are summarized in Table 9. Here, it is seen that x (better accuracy count) is significantly larger than z (poorer accuracy count) especially for scaling-space method (RM-InvSig-S) versus original feature space (RM-InvSig), i.e. 72.5% versus 10%. The accuracy improvement ranges from about near zero to about 4.3% for RM-InvSig-S over RM-InvSig. Apart from having comparable or better average accuracy than FP-InvSig-3, FP-InvSig-l, GLM and GLM-S, the performance of RM-InvSig-S is also shown to be comparable to the top classifiers found in the literature. The good generalization performance (based on the test accuracy results) of RM-InvSig and RM-InvSig-S as compared to the related GLM and MLP may be attributed to the modified link or activation function resulted from RM2 series approximation.

Here it is noted that due to different sensitivities for large and small feature values, the scaling-space is only useful for small neighborhood values around the original feature space ($\rho \in [0, 0.5]$ according to the empirical observation). From the data attributes point of view, a mild indication that the maximum accuracy may decrease with the feature dimension can be observed. Also, the feature scaling-space appears to benefit RM-InvSig for a particular range of data.

Table 9 Summary of results

Method \ Number of data sets	x	y	z	$\frac{x}{x+y+z}(\%)$	$\frac{y}{x+y+z}(\%)$	$\frac{z}{x+y+z}(\%)$
RM-InvSig-S vs RM-InvSig	29	7	4	72.5	17.5	10.0
RM-InvSig-S vs RM2	15	4	21	37.5	10.0	52.5
RM-InvSig-S vs FP-InvSig-3	27	3	10	67.5	7.5	25.0
RM-InvSig-S vs FP-InvSig-l	36	0	4	90.0	0	10.0
RM-InvSig-S vs GLM-S	31	0	9	77.5	0	22.5
RM-InvSig vs GLM	30	1	9	75.0	2.5	22.5

x: better performance, y: equal performance, z: poorer performance.
RM2: RM2 model without adopting the reciprocal-sigmoid basis, i.e. obtained from replacing ϕ by x in (11).
RM-InvSig: Reciprocal-sigmoid model expanded by RM2 as shown in (11).
RM-InvSig-S: Scaling-space learning of reciprocal-sigmoid model expanded by RM2 as shown in (11).
FP-InvSig-3: Kernel Ridge Regression using a 3rd-order model for the polynomial kernel as shown in (18).
FP-InvSig-l: Kernel Ridge Regression using a lth-order model for the polynomial kernel as shown in (18).
GLM: GLM adopting a logit link as in (12).
GLM-S: Scaling-space learning of GLM.

7. Conclusion

With a refinement of a polynomial formulation, a reciprocal-sigmoid Φ-machine-like model is proposed for pattern classification. The model can be considered as an approximation to logistic regression under the framework of Generalized Linear Models. Unlike logistic regression, the classifier is linear in parameters and the solution for the model weights can be computed in a single step. This deterministic solution relieves the problems of local minima and recursive computation as in gradient-based search on a nonlinear model. While this formulation is likely to use a smaller set of parameters as compared with one without using the reciprocal-sigmoid as the basis function, the reciprocal-sigmoid classifier is shown to have good approximation capability when a full rank polynomial matrix can be constructed from the training data. To stabilize the solution for high order models, the classifier is trained using the feature scaling-space. Unlike bagging, boosting and random subspace methods, the scaling-space learning uses only a single classifier for decision.

Extensive experiments show that for both synthetic and physical data, feature scaling-space training provides a clear indication of test accuracy improvement over that based on the original feature space when an appropriate scaling parameter is selected. Main reasons for the improved performance are attributed to (i) improved numerical conditioning from an increased pool of training data, and (ii) appropriate biasing of decision hyper-surface from adjustment of the scaling parameter. The empirical evidence also shows that the proposed classifier can be suitable for problems with low feature dimension, particularly, the scaling-space takes effect for those problems with ratio of sample number over feature dimension within [6, 900]. Besides translating the bagging effort into a process of selecting a scaling parameter from the single-step least-squares training paradigm, the average accuracy of the reciprocal-sigmoid classifier learned by the feature scaling-space is shown to be comparable to several top classifiers found in the literature.

Appendix

A.1. Matlab Codes for Generation of Regressor Matrix *P*

```
%------------------- Beginning of Function ---------------------%
function P = RM2invsig(order,X)
% Build regressor matrix P (mxK):
%    order = desired order of approximation,
%    X = input matrix (mxl), K = number of parameters to be est.
%    m = number of data samples, L = input dimension.
[m,L] = size(X);
X = 1 + exp(-X); %inverse of sigmoid
MM1=[]; MM3=[]; Msum=sum(X,2);
for i=1:order
   for k=1:L
      M1(:,k)=X(:,k).^i;
      if (i>1)
         M3(:,k)=X(:,k).*Msum.^(i-1);
      end
   end
   MM1=[MM1,M1];
   if (i>1)
      MM3=[MM3,M3];
   end
end
P = [ones(m,1),MM1,MM3];
return;
%---------------------- End of Function ------------------------%
```

A.2. XOR Learning Example

```
%------------------- Beginning of Example ----------------------%
X = [ 0  0; 0  1; 1  1; 1  0]; % XOR inputs
y = [0; 1; 0; 1];              % XOR outputs
XXX = [0.95*X; X; 1.05*X];  % Scaling-space inputs
yyy = [y; y; y];            % Scaling-space outputs

order = 2;                % order setting
P = RM2invsig(order,XXX); % generate matrix P
alpha=inv(P'*P)*P'*yyy;   % solve for alpha

Pt = RM2invsig(order,X);  % generate matrix P for test
t_train = Pt*alpha        % trained output
%---------------------- End of Example -----------------------%
```

Acknowledgments The author would like to thank Professor Risto Miikkulainen and the anonymous reviewers for their very constructive comments to improve the paper. Special thanks go to Dr Louis Shue for English proof reading.

References

Agresti, A. (2002). *Categorical data analysis*, 2nd ed. New Jersey: John Wiley & Sons.

Bishop, C. M. (1995). *Neural networks for pattern recognition*. New York: Oxford University Press Inc.

Boser, B. E., Guyon, I. M., & Vapnik, V. N. (ACM, 1992). A training algorithm for optimal margin classifiers. In: *Fifth Annual Workshop on Computational Learning Theory* (pp. 144–152). Pittsburgh.

Breiman, L. (1994). *Bagging predictors*. Department of Statistics, University of California, Berkeley. Technical Report No. 421.

Breiman, L. (1996). Bagging predictors. *Machine Learning, 24*(2), 123–140.

Burges, C. J. C. (1998). A tutorial on support vector machines for pattern recognition. *Data Mining & Knowledge Discovery, 2*(2), 121–167.
Duch, W., & Grudziński, K. (1999). Weighting and selection of features. In: *Proceedings of the Workshop on Intelligent Information Systems VIII* (pp 32–36). Ustron, Poland.
Duda, R. O., & Hart, P. E. (1973). *Pattern classification and scene analysis*. New York: Wiley & Sons.
Duda, R. O., Hart, P. E., & Stork, D. G. (2001) *Pattern classification*, 2nd ed. New York: John Wiley & Sons, Inc.
Dunn, P. (2000). GLMLAB: Generalized linear models in MATLAB. In [http://www.sci.usq.edu.au/staff/dunn/glmlab/glmlab.html]. Dept. of Mathematics & Computing, University of Southern Queensland. (Version 2.5).
Dunn, P. K. (1999). A graphical user interface to generalized linear models in MATLAB. *The Journal of Statistical Software*, *4*(4).
Efron, B., & Tibshirani, R. (1993). *An introduction to the bootstrap*. Chapman and Hall.
Gordon, G. J. (2002). Generalized^2 Linear^2 Models. In: *Advances in Neural Information Processing Systems (NIPS 2002)* (pp. 577–584). Vancouver, British Columbia, Canada.
Grandvalet, Y. (2000). Anisotropic noise injection for input variables relevance determination. *IEEE Trans. on Neural Networks, 11*(6), 1201–1212.
Grandvalet, Y., & Canu, S. (2002). Adaptive scaling for feature selection in SVMs. *Neural Information Processing Systems*.
Hardin, J., & Hilbe, J. (2001). *Generalized linear models and extensions*. LakeWay Drive: Stata Press.
Helzer, A., Barzohar, M., & Malah, D. (2004). Stable fitting of 2D curves and 3D surfaces by implicit polynomials. *IEEE Trans. Pattern Analysis and Machine Intelligence, 26*(10), 1283–1294.
Hornik, K., Stinchcombe, M., & White, H. (1989). Multi-layer feedforward networks are universal approximators. *Neural Networks, 2*(5), 359–366.
Huang, G.-B. (2003). Learning capability and storage capacity of two-hidden-layer feedforward networks. *IEEE Trans. Neural Networks, 14*(2), 274–281.
Huang, G.-B., & Babri, H. A. (1998). Upper bounds on the number of hidden neurons in feedforward networks with arbitrary bounded nonlinear activation function. *IEEE Trans. Neural Networks, 9*(1), 224–801.
Juszczak, P., Tax, D. M. J., & Duin, R. P. W. (2000). Feature scaling in support vector data description. In: N., Japkowicz (Ed.), *Learning from Imbalanced Data Sets* (pp. 25–30). Menlo Park, CA: AAAI Press.
Lam, W., Keung, C.-K., & Liu, D. (2002). Discovering useful concept prototypes for classification based on filtering and abstraction. *IEEE Trans. Pattern Analysis and Machine Intelligence, 24*(8), 1075–1090.
Li, J., Dong, G., Ramamohanarao, K., & Wong, L. (2004). DeEPs: A new instance-based lazy discovery and classification system. *Machine Learning, 54*(2), 99–124.
Lim, T.-S., Loh, W.-Y., & Shil, Y.-S. (2000). A comparison of prediction accuracy, complexity, and training time of thirty-three old and new classification algorithms. *Machine Learning, 40*(3), 203–228.
Lindeberg, T. (1990). Scale-space for discrete signals. *IEEE Trans. Pattern Analysis and Machine Intelligence, 12*(3), 234–254.
Ma, J., Zhao, Y., & Ahalt, S. (2002). OSU SVM classifier matlab toolbox (ver 3.00). In [http://eewww.eng.ohio-state.edu/∼maj/osu_svm/]. The Ohio State University.
McCullagh, P., & Nelder, J. A. (1989). *Generalized linear models* 2nd ed. London: Chapman and Hall.
Mitchell, T. M. (1997). *Machine learning*. Singapore, International Edition: The McGraw-Hill Companies, Inc.
Nelder, J. A., & Wedderburn, R. W. M. (1972). Generalized linear models. *Journal of the Royal Statistical Society, Series A*, *135*, 370–384.
Neter, J., Kutner, M. H., Nachtsheim, C. J., & Wasserman, W. (1996). *Applied linear regression models*, 3rd ed. Irwin, Chicago.
Newman, D. J., Hettich, S., Blake, C. L., & Merz, C. J. (1998). UCI repository of machine learning databases. In [http://www.ics.uci.edu/∼mlearn/MLRepository.html]. University of California, Irvine, Dept. of Information and Computer Sciences.
Nilsson, N. J. (1965). *Learning machines*. New York: McGraw-Hill.
Osuna, E. E., Freund, R., & Girosi, F. (1997). *Support Vector Machines: Training and Applications*. MIT Artificial Intelligence Laboratory and CBCL Dept. of Brain and Cognitive Sciences. (Technical Report: A.I. Memo No. 1602, C.B.C.L. Paper No. 144).
Poggio, T., & Girosi, F. (1990). Networks for approximation and learning. *Proceedings of the IEEE, 78*(9), 1481–1497.
Precup, D., & Utgoff, P. E. (2004). Classification using Φ-machines and constructive function approximation. *Machine Learning, 55*(1), 31–52.
Ripley, B. D. (1996). *Pattern recognition and neural networks*. Cambridge University Press.

Schürmann, J. (1996). *Pattern classification: A unified view of statistical and neural approaches*. New York: John Wiley & Sons, Inc.

Shawe-Taylor, J., & Cristianini, N. (2004). *Kernel methods for pattern analysis*. Cambridge University Press.

Skurichina, M., & Duin, R. P. W. (2002). Bagging, boosting and the random subspace method for linear classifiers. *Pattern Analysis and Applications*, *5*, 121–135.

Skurichina, M., Raudys, S., & Duin, R. P. W. (2000). K-nearest neighbours directed noise injection in multilayer perceptron training. *IEEE Trans. on Neural Networks, 11*(2), 504–511.

Tax, D. M. J., & Duin, R. P. W. (2000). Data description in subspaces. In: *Proc. 15th International Conference on Pattern Recognition (ICPR)*, (Vol. 2, pp. 672–675). Barcelona, Spain.

The MathWorks (2003). Matlab and simulink. In [http://www.mathworks.com/].

Tipping, M. E. (2000). The relevance vector machine. In: S. A. Solla, T. K. Leen, & K.-R. Müller (Eds.), *Advances in Neural Information Processing Systems*, (Vol. 12, pp 652–658).

Tipping, M. E. (2001). Sparse bayesian learning and the relevance vector machine. *Journal of Machine Learning Research*, *1*, 211–244.

Toh, K.-A. (2003). Fingerprint and speaker verification decisions fusion. In: *International Conference on Image Analysis and Processing (ICIAP)* (pp 626–631). Mantova, Italy.

Toh, K.-A., Tran, Q.-L., & Srinivasan, D. (2004). Benchmarking a reduced multivariate polynomial pattern classifier. *IEEE Trans. Pattern Analysis and Machine Intelligence, 26*(6), 740–755.

Vapnik, V. N. (1998). *Statistical Learning Theory*. Wiley-Interscience Pub.

Vetter, T., Jones, M. J., & Poggio, T. (1997). A bootstrapping algorithm for learning linear models of object classes. In *Proc. International Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 40–46).