Sparse Bayesian Learning and the Relevance Vector Machine(Mike E. Tipping)

Milind Nakul(180420), Nakul Singh(180455) and Astitva Chaudhary(180157)

Abstract

Index Terms

SVM, Sparse Bayesian Learning, Relevance Vector Machine

I. Introduction

The supervised learning focuses on learning a function over the input space which can make accurate and sensible predictions. Supervised learning can be broadly categorized into 2 categories based on the type of output variable. If the target variable is a continuous function of input attributes then the learning task is categorized as regression. On the other hand if the target variable is discrete then it becomes a classification problem.

For the purpose of regression and classification, models linear in the parameters are often exploited due to their flexibility and ease of computation. A special class of these linearly parameterized models is the support vector machine described by the following equation:

$$y(\mathbf{x};\mathbf{w}) = \sum_{i=1}^{N} w_i \mathbf{K}(\mathbf{x},\mathbf{x_i}) + w_0 \tag{1}$$

Here $\mathbf{K}$ is the kernel function which basically takes the dot product of the basis functions in a dimension where they are linearly separable. The task of supervised learning is then to learn the parameters $\mathbf{w}$.

The authors are with the Department of Electrical Engineering, IIT Kanpur, Kanpur 208016, India. email: {mnakul, nakulsh, astitvac}@iitk.ac.in.

A specially useful feature of the SVM is sparsity in the model, which is caused due to the fact that only a small sub set of the basis functions (often called the support vectors) can actually influence the optimization problem. This sparsity in the model parameters leads to substantial improvement in the computation speed.

However the SVM model also has several disadvantages :

- Not a probabilistic model. Similar to the wireless parlance of hard decoding, SVM outputs a point estimate, thus we have practically no information regarding the uncertainty in our predictions.
- In soft margin SVMs we have an additional parameter C which requires the cross-validation procedure for proper estimation, thus requiring additional computation.
- The kernel functions used are limited by the condition that they must be continuous symmetric kernel of positive integral operator.

Based on the limitations of the SVM model, this paper explores a probabilistic approach to the SVM model. A Bayesian treatment is adopted for estimation of the SVM model parameters of (1), termed as the Relevance Vector machine(RVM). Along with the advantages of the SVM model it also gets rid of the limitations of the SVM model thus providing a promising alternative.

The following sections explore the applications of RVM on regression and classification learning tasks.

II. Sparse Bayesian Learning for Regression

In this section the paper applies the idea of sparse Bayesian learning to regression tasks which can be solved by the SVM model (1). The first subsection describes the model used. The second and third subsections outline the inference and optimization procedures respectively. The last section details the procedure of making predictions

A. Model Specification

To adopt a probabilistic framework it is assumed that the target output is sampled from a model with additive noise. And the noise is sampled from a zero mean Gaussian distribution of variance σ^2.

$$t_n = y(\mathbf{x_n}; \mathbf{w}) + \epsilon_n \tag{2}$$

$$\epsilon_n \sim \mathcal{N}(0, \sigma^2) \tag{3}$$

The target outputs are obtained from (2), and the noise is sampled from (3). $\mathbf{y}$ is the SVM model as described in (1). Thus the target sample itself belongs to a Gaussian distribution with mean $y(\mathbf{x_n})$ and variance σ^2.

$$t_n \sim \mathcal{N}(y(\mathbf{x_n}),\, \sigma^2) \tag{4}$$

Now for n independent realizations drawn from (4) the likelihood of the complete data set can be written as :

$$p(\mathbf{t}|\mathbf{w}, \sigma^2) = (2\pi\sigma^2)^{-N/2} exp \frac{-1}{2\sigma^2} \|\mathbf{t} - \mathbf{\Phi w}\|^2 \tag{5}$$

where $\mathbf{t}$ and $\mathbf{w}$ are n dimensional column vectors and Φ is the design matrix given by :

$$\Phi = \begin{bmatrix} 1 & \mathbf{K(x_1, x_1)} & \mathbf{K(x_1, x_2)} & \dots \\ \vdots & \ddots & & \\ 1 & \mathbf{K(x_N, x_1)} & \dots & \mathbf{K(x_N, x_{N+1})} \end{bmatrix} \tag{6}$$

where $\mathbf{K(x_i, x_j)}$ is the kernel function as defined in (1).

An intuitive analysis tells us that the model might suffer from severe over-fitting due to the large number of parameters used. This is where our Bayesian perspective helps. We can avoid over-fitting by choosing an appropriate prior on the parameters $\mathbf{w}$. The prior can be thought as having a regularizing effect thus penalizing the model if it becomes too complex.

For less complex and sparse models, zero mean Gaussian distribution is often used as the prior distribution.

$$p(\mathbf{w}|\alpha) = \prod_{i=0}^{N} \mathcal{N}(0,\, \alpha_i^{-1}) \tag{7}$$

where α is a N+1 dimensional column vector containing a hyperparameter α_i associated independently with each of the parameter w_i.

Finally to complete the model specification, hyperpriors are defined over α and the noise variance σ^2. Since α and σ^2 both are modelling variance of normal distributions a popular choice of hyperprior is the Gamma distribution.

$$p(\alpha) = \prod_{i=0}^{N} Gamma(\alpha_i|a, b) \tag{8}$$

$$p(\beta) = Gamma(\beta|c, d) \tag{9}$$

where β is σ^{-2}. The variables a,b,c,d encode our prior beliefs for the values of α and β. To make a non-informative prior the variables a,b,c,d are set to very small values.

B. *Inference*

First of the paper defines the posterior over all the unknowns of the model using Bayes rule as follows,

$$p(\mathbf{w}, \alpha, \sigma^2|\mathbf{t}) = \frac{p(\mathbf{t}|\alpha, \sigma^2, \mathbf{w})p(\mathbf{w}, \alpha, \sigma^2)}{p(\mathbf{t})} \tag{10}$$

The predictive distribution for a new output $\mathbf{t}_*$ can be written using the following identity,

$$p(\mathbf{t}_*|\mathbf{t}) = \int p(\mathbf{t}_*|\mathbf{w}, \alpha, \sigma^2)p(\mathbf{w}, \alpha, \sigma^2|\mathbf{t})\ d\mathbf{w}\ d\alpha\ d\sigma^2 \tag{11}$$

We cannot compute the posterior over all the unknowns analytically because the normalising integral in the denominator $p(\mathbf{t}) = \int p(\mathbf{t}|\alpha, \sigma^2, \mathbf{w})p(\mathbf{w}, \alpha, \sigma^2)\ d\mathbf{w}\ d\alpha\ d\sigma^2$ cannot be performed and hence the authors resort to approximations to compute the predictive distribution.

So the author decomposes the posterior distribution over all the unknowns as product of posterior over the weights and posterior over α and σ^2.

$$p(\mathbf{w}, \alpha, \sigma^2|\mathbf{t}) = p(\mathbf{w}|\alpha, \sigma^2, \mathbf{t})p(\alpha, \sigma^2|\mathbf{t}) \tag{12}$$

The posterior over the weights can be written as follows,

$$p(\mathbf{w}|\alpha, \sigma^2, \mathbf{t}) = \frac{p(\mathbf{t}|\mathbf{w}, \sigma^2)p(\mathbf{w}|\alpha)}{p(\mathbf{t}|\alpha, \sigma^2)} \tag{13}$$

Since here the likelihood and the prior over the weights are both Gaussian which are conjugate pairs of one another, the posterior over the weights is also a Gaussian(same as prior) with updated set of hyperparameters.

$$p(\mathbf{w}|\alpha, \sigma^2, \mathbf{t}) = \mathcal{N}(\mathbf{w}|\mu, \mathbf{\Sigma}) \tag{14}$$

where,

$$\mathbf{\Sigma} = (\sigma^{-2}\ \mathbf{\Phi^T\Phi} + \mathbf{A})^{-1} \tag{15}$$

$$\mu = \sigma^{-2}\ \mathbf{\Sigma\Phi^T t} \tag{16}$$

$$\mathbf{A} = diag(\alpha_0, \alpha_1, , , , , \alpha_N) \tag{17}$$

The author says that the posterior over α and σ^2 has to be approximated as a delta function at its mode, α_{MP} and σ^2_{MP}.

$$p(\alpha, \sigma^2|\mathbf{t}) \approx \delta(\alpha_{MP}, \sigma^2_{MP}) \tag{18}$$

However this does not mean that the posterior over α and σ^2 has its entire mass at the mode of the distribution. For the objective of the paper of computing the predictive distribution only the following has to hold,

$$\int p(\mathbf{t}_*|\alpha, \sigma^2)p(\alpha, \sigma^2|t)\ d\alpha\ d\sigma^2 \approx \int p(\mathbf{t}_*|\alpha, \sigma^2)\delta(\alpha, \sigma^2)\ d\alpha\ d\sigma^2 \tag{19}$$

We need to compute the mode for the posterior distribution, $p(\alpha, \sigma^2|t) \propto p(\mathbf{t}|\alpha, \sigma^2)p(\alpha)p(\sigma^2)$. For uniform priors over α and σ^2, $p(\alpha, \sigma^2|t) \propto p(\mathbf{t}|\alpha, \sigma^2)$.

$$p(\mathbf{t}|\alpha, \sigma^2) = \int p(\mathbf{t}|\mathbf{w}, \sigma^2)p(\mathbf{w}|\alpha)d\mathbf{w} \tag{20}$$

This is the marginal likelihood. Since both the distributions inside the integral are Gaussians, the marginal likelihood can be seen as convolution of Gaussians and computed as,

$$p(\mathbf{t}|\alpha, \sigma^2) = (2\pi)^{-N/2}|\sigma^2\mathbf{I} + \mathbf{\Phi A^{-1}\Phi^T}|^{-1/2}exp(-\frac{1}{2}\mathbf{t^T}(\boldsymbol{\sigma^2}\mathbf{I} + \mathbf{\Phi A^{-1}\Phi^T})^{\mathbf{-1}}\mathbf{t}) \tag{21}$$

Then the author maximizes this marginal likelihood expression to obtain the mode of the posterior, α_{MP} and σ^2_{MP}

C. Optimising the Hyperparameters

The author differentiates the marginal likelihood with respect to α and σ^2 and sets it to zero. However, closed form solutions for α_{MP} and σ^2_{MP} cannot be obtained. The author relies on iterative measures for finding the optimal values of the hyperparameters.

$$\frac{\partial p(\mathbf{t}|\alpha,\boldsymbol{\sigma^2})}{\partial \alpha} = 0 \; and \; \frac{\partial p(\mathbf{t}|\alpha,\boldsymbol{\sigma^2})}{\partial \sigma^2} = 0 \tag{22}$$

From the above equation the author obtains the following update equations for the iterative approach.

$$\alpha_i^{new} = \frac{\gamma_i}{\mu_i^2} \tag{23}$$

where μ_i is the i^{th} posterior mean over the weights as has been calculated earlier and,

$$\gamma_i = 1 - \alpha_i \Sigma_{ii}, \; \gamma_i \in [0,1] \tag{24}$$

and Σ_{ii} is the i^{th} diagonal element of the covariance matrix of the posterior over $\mathbf{w}$.

Each γ_i can be interpreted as a measure of how precisely the corresponding w_i is computed. For α_i approaching infinity, we know that the variance of the corresponding w_i approaches 0 and $\Sigma_{ii} = \alpha_i^{-1}$, hence $\gamma_i \approx 0$.

For small values of α_i and w_i fits the data $\gamma_i \approx 1$.

For the noise variance we have the following update equation,

$$(\sigma^2)^{(new)} = \frac{||\mathbf{t} - \boldsymbol{\Phi}\mu||^2}{N - \sum_i \gamma_i} \tag{25}$$

Here N is the total number of data points available and μ is the mean of the posterior over weights calculated using the current values of the hyperparameters.

Thus the optimization algorithm proceeds by the iteration over the above mentioned update equations. This is followed by the update of μ and $\boldsymbol{\Sigma}$ of the posterior over weights. The iterations are continued until a certain convergence criteria is satisfied or the a minimum number of required iterations are completed.

After the author finds the most optimal values of the hyperparameters, the observation is that many of the α_i tend to infinity. This results in near 0 variance of the correponding w_i and so $p(w_i|\mathbf{t},\alpha,\boldsymbol{\sigma^2})$ becomes concentrated at 0. Thus the corresponding basis functions can be omitted from the final solution as they don't have a role to play.

D. Making Predictions

After using the iterative approach,the author finds the optimal values of the hyperparameters as $\alpha_{\mathbf{MP}}$ and σ^2_{MP}. We can easily compute the predictive distribution of the output $\mathbf{t}_*$ for a new data point $\mathbf{x}_*$ using the following identity,

$$p(\mathbf{t}_*|\mathbf{t}, \alpha_{\mathbf{MP}}, \sigma^2_{MP}) = \int p(\mathbf{t}_*|\mathbf{w}, \sigma^2_{MP}) p(\mathbf{w}|\mathbf{t}, \alpha_{\mathbf{MP}}, \sigma^2_{MP}) \, d\mathbf{w} \tag{26}$$

Since both of the distributions inside the integral are Gaussians, the predictive distribution is a convolution of Gaussian distributions and hence its expression can be computed analytically. Thus,

$$p(\mathbf{t}_*|\mathbf{t}, \alpha_{\mathbf{MP}}, \sigma^2_{MP}) = \mathcal{N}(\mathbf{t}_*|\mu_*, \sigma^2_*) \tag{27}$$

where,

$$\mu_* = \mu^{\mathbf{T}} \phi(\mathbf{x}_*) \tag{28}$$

$$\sigma^2_* = \sigma^2_{MP} + \phi(\mathbf{x}_*)^T \mathbf{\Sigma} \phi(\mathbf{x}_*) \tag{29}$$

In these equations μ and $\mathbf{\Sigma}$ have been calculated according to earlier equations using the optimised values of hyperparameters.

The mean of the predictive distribution is the linear sum of the basis functions weighted by the posterior mean weights. The sparsity comes from the fact that many of these weights come out to be zero. The variance of the predictive distribution is the sum of the noise on the data($\phi(\mathbf{x}_*)^T \mathbf{\Sigma} \phi(\mathbf{x}_*)$) and the uncertainty in the prediction of the weights(σ^2_{MP}).

III. Sparse Bayesian Classification

The previous sections have explained Regression. Here, the author discusses Classification which follows a similar framework. The target quantities are now discrete (as opposed to continuous target variables for regression), so the author changes the target conditional distributions and introduce a link function. For classification, we need to predict the posterior class probabilities for a given input vector $\mathbf{x}$. We use a sigmoid link function, $\sigma(y) = \frac{1}{1+e^{-y}}$. We apply $\sigma(y)$ to $y(\mathbf{x})$ and take the Bernoulli distribution for $P(t|\mathbf{x})$. The likelihood $P(t|\mathbf{x})$ is given as,

$$P(t|\mathbf{x}) = \prod_{n=1}^{N} \sigma\{y(\mathbf{x}_n;\mathbf{w})\}^{t_n}[1-\sigma\{y(\mathbf{x}_n;\mathbf{w})\}]^{1-t_n}, \tag{30}$$

where $t_n \in \{0,1\}$. Here, noise variance is not taken into account.

Due to non conjugate prior and likelihood distributions, we are unable to get the expressions for the weights analytically and so cannot get the closed form expressions for the marginal likelihood $P(\mathbf{t}|\alpha)$ and the posterior distribution $p(\mathbf{w}|\mathbf{t},\alpha)$. We use Laplace's method to approximate the weights.

The $\mathbf{w}_{MP}$ (most probable weights) can be found as the mode of the posterior distribution for a fixed set of α values. Closed form expression of the posterior distribution cannot be obtained as we discussed before, so we make use of the fact that $p(\mathbf{w}|\mathbf{t},\alpha) \propto P(\mathbf{t}|\mathbf{w})p(\mathbf{w}|\alpha)$ and also that the logarithm is a strictly concave function. We find $\mathbf{w}_{MP}$ as,

$$\mathbf{w}_{MP} =_{\mathbf{w}} log\{p(\mathbf{w}|\mathbf{t},\alpha)\} =_{\mathbf{w}} log\{P(\mathbf{t}|\mathbf{w})p(\mathbf{w}|\alpha)\} \tag{31}$$

where $log\{P(\mathbf{t}|\mathbf{w})p(\mathbf{w}|\alpha)$ is given as,

$$logP(\mathbf{t}|\mathbf{w})p(\mathbf{w}|\alpha) = \sum_{n=1}^{N}[t_n log(y_n) + (1-t_n)log(1-y_n)] - \frac{1}{2}\mathbf{w}^T\mathbf{A}\mathbf{w} \tag{32}$$

where $y_n = \sigma\{y(\mathbf{x}_n;\mathbf{w})\}$. We can use the Second Order Newton Rhapson method to calculate the weights iteratively, using the Hessian. The Hessian is calculated as,

$$logp(\mathbf{w}|\mathbf{t},\alpha)\mathbf{w}\mathbf{w}^T|\mathbf{w}_{MP} = -(\mathbf{\Phi}^T\mathbf{B}\mathbf{\Phi}+\mathbf{A}) \tag{33}$$

where $\mathbf{B} = diag(\beta_1, \beta_2, ..., \beta_N)$ with $\beta_n = \sigma\{y(\mathbf{x}_n)\}[1-\sigma\{y(\mathbf{x}_n)\}]$.

Using the Gaussian second order approximation for the posterior and the fact that $logp(\mathbf{w}|\mathbf{t},\alpha)\mathbf{w}|\mathbf{w}_{MP} = 0$, we can write the parameters of the Gaussian distribution as,

$$\mathbf{\Sigma}^{-1} = -logp(\mathbf{w}|\mathbf{t},\alpha)\mathbf{w}\mathbf{w}^T|\mathbf{w}_{MP} \tag{34}$$

$$\mathbf{\Sigma} = (\mathbf{\Phi}^T\mathbf{B}\mathbf{\Phi}+\mathbf{A})^{-1} \tag{35}$$

$$\mu = \mathbf{w}_{MP} = \mathbf{\Sigma}\mathbf{\Phi}^T\mathbf{B}\mathbf{t} = (\mathbf{\Phi}^T\mathbf{B}\mathbf{\Phi}+\mathbf{A})^{-1}\mathbf{\Phi}^T\mathbf{B}\mathbf{t} \tag{36}$$

The hyperparameters α are updated as in (23).

For K-class classification, the likelihood can be written as,

$$P(\mathbf{t}|\mathbf{w}) = \prod_{n=1}^{N}\prod_{k=1}^{K}\sigma\{y_k(\mathbf{x}_n;\mathbf{w}_k)\}^{t_{nk}} \tag{37}$$

where ”one-of-K” or ”one-hot” encoding is used for t, i.e, $t_k = 1$ if the input vector $\mathbf{x}$ belongs to the kth class and $t_k = 0$ if the input vector $\mathbf{x}$ does not belong to the kth class.

IV. Relevance Vector Examples

The previous sections provided an outline of the procedure to be followed while using the Relevance vector machine for classification and regression tasks. This section of the paper deals with the applications of RVM to synthetic data sets and a benchmark comparison to its SVM counterpart.

A. Relevance Vector Regression: the ‘sinc’ function

For regression tasks instead of a decision boundary an ϵ sensitive region is introduced around the function, within which errors are not penalized. Thus the support vectors lie on or outside this ϵ region. Now to compare sparse Bayesian learning with SVM on a synthetic data set, the sinc function is used. For using RVM model a noise of fixed variance (0.01^2) was added to the data set. This can be considered analogous to the ϵ tube defined for SVM. Setting ϵ as 0.01 and using a linear spline kernel :

$$\mathbf{K}(x_m, x_n) = 1 + x_m x_n + x_m x_n min(x_m, x_n) - \frac{x_m + x_n}{2} min(x_m, x_n)^2 + \frac{min(x_m, x_n)^3}{3} \tag{38}$$

In case of noise free sampling from the sinc function the number of support vectors for the SVM model was greater than the number of relevance vectors for the RVM model.Similarly for the case in which noise was added to the sampled values RVM outperformed SVM both in terms of accuracy and sparsity.

A detailed comparison between the two models is tabulated in section 4.4.1 based on several benchmark tests.

B. Relevance Vector Classification: Ripley’s synthetic data

Synthetic data is used to visually show the selection of RVM for classification. In the data generated by Ripley from a mixture of 2 overlapping Gaussians, class 1 is marked with $'\times'$ and class 2 is marked with $'\bullet'$. Performance of RVM is measured against SVM. A ’Gaussian’ Kernel is used which is given as,

$$K(\mathbf{x}_m, \mathbf{x}_n) = exp(-r^{-2}\|\mathbf{x}_m - \mathbf{x}_n\|^2) \tag{39}$$

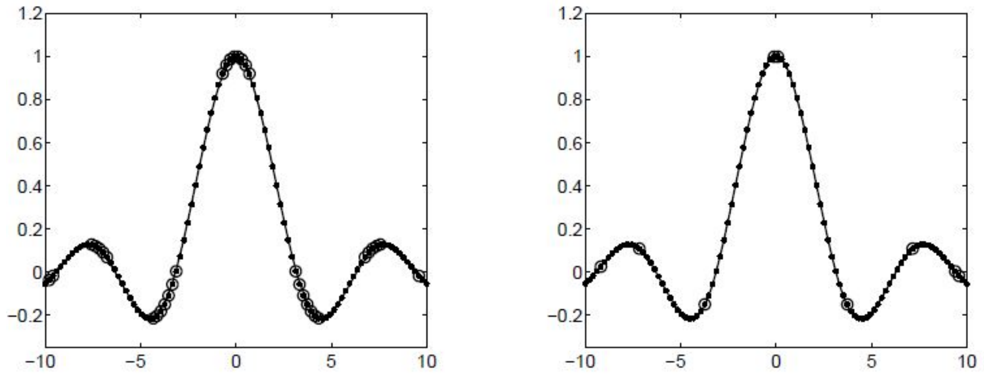

Fig. 1: SVM(left) and RVM(right) approximation to sinc function with 100 noise free samples. The circled points are the support and relevance vectors respectively.

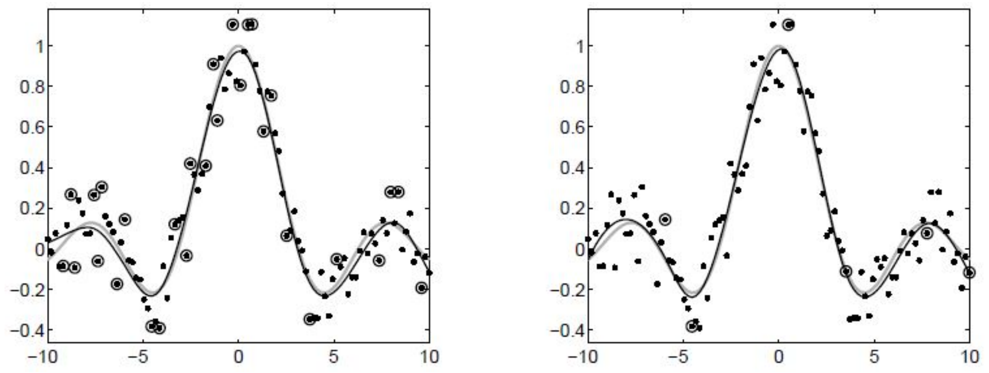

Fig. 2: SVM(left) and RVM(right) approximation to sinc function with 100 noisy samples. The circled points are the support and relevance vectors respectively.

where the width parameter r is chosen to be 0.5 and the value of C is chosen by applying 5-fold cross validation on the training set. In Figure 3 of [1], the results of SVM and RVM for a randomly selected training dataset of 100 points (selected from a set of 250 points) is shown. We can see that the boundary of classification for SVM is more complex than the boundary of classification for RVM. Moreover, we know that the SVM utilises 38 kernel functions whereas the RVM utilises only 4 kernel functions for this example. This clear difference in sparsity can be appreciated. From Figure 3 of [1], it can also be seen that the relevance vectors(circled) are far away from the boundary as compared to the support vectors(circled). When tested over a 1000 point dataset, the RVM(error 9.3%) performs better than SVM(error 10.6%).

C. Extensions

We use synthetic data to highlight two more features of Sparse Bayesian approach, the ability to use arbitrary basis functions and the ability to directly optimise parameters within the kernel.

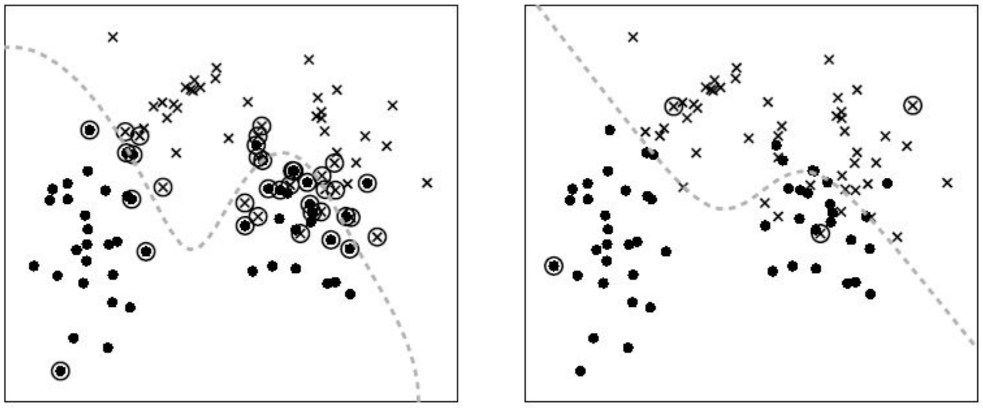

Fig. 3: SVM(left) and RVM(right) classifier on 100 examples from Ripley's Gaussian mixture data set. The decision boundary is shown dashed.

In both SVM and RVM, we have to choose the Kernel functions and choose values for the parameters for that kernel functions, for e.g., the width parameter $\eta = r^{-2}$ for the Gaussian Kernel used in the previous section. The value of η is determined through cross validation. This approach of determining values is fine for a single parameter η but not when there is a parameter η_i associated with each input variable. Multiple parameters are used to deal with irrelevant input variables.

We take up the example of estimating the function $y(x_1, x_2) = sinc(x_1) + 0.1x_2$. The training set consists of 100 data points with added Gaussian noise whose variance equals 0.1. However, there are two things that make it difficult to approximate. First, that the function $y(x_1, x_2)$ is linear in x_2, which will not be approximated optimally with non linear functions. Second, that the $sinc(.)$ function depends entirely on x_1. Upon approximating the $sinc(x_1)$ function, we will get a term depending on x_2 which will add to the noise.

Figure 4 of [1] shows the approximation of $y(x_1, x_2)$ using SVM with a Gaussian Kernel, taking width parameter $r = 3$.

To improve upon these results, we add more basis functions. We add $x_1, x_2, x_1^2, x_2^2, x_1x_2$ values as extra columns to the design matrix $\mathbf{\Phi}$ and also introduce their respective weights and hyperparameters, which are updated as before. The only relevant function that is added is x_2, and the rest of added function are to be pruned, i.e., set to 0. Also, to improve the results, we add parameters η_1 and η_2 to the kernel function which becomes,

$$K(\mathbf{x}_m, \mathbf{x}_n) = exp\{-\eta_1(x_{m1} - x_{n1})^2 - \eta_2(x_{m2} - x_{n2})^2\} \tag{40}$$

We maximise the likelihood w.r.t. to the parameters η_1 and η_2, at each step of hyperparameter update(23).

Figure 5 of [1], shows the RVM approximation which is more accurate than the previous approximation shown in figure 4 of [1]. Table 1 compares the error of RVM(0.0053) and SVM(0.0194). SVM uses 75 functions whereas RVM is sparser and uses only 8 functions. Table 2 shows the estimated weights for the RVM model. We see that of all the weights of the 'added functions' only w_{x_2} is non zero(0.102) and is quite close to the actual weight(0.1). Also, the width parameters are estimated, with $\eta_1(0.0997)$ being $\gg \eta_2(0.0002)$ such that $K(\mathbf{x_m}, \mathbf{x_n}) \approx exp\{-\eta_1(x_{m1} - x_{n1})^2\}$.

This approach is only limited by its computational complexity and the step of optimisation over α and η.

D. Benchmark Comparisons

In this section of the paper the performance of SVM and RVM on several benchmark data sets are compared and tabulated. The two models are compared on the basis of number of support/relevance vectors and their error in regression(and classification) tasks. N is the size of training data set and d is the dimension of input space.

The kernel used is Gaussian in all cases except the Friedman #1 data set.

Data Set	N	d	SVM(err)	RVM(err)	SVM(Vec)	RVM(Vec)
Sinc (Gaussian noise)	100	1	0.378	0.326	45.2	6.7
Sinc (Uniform noise)	100	1	0.215	0.187	44.3	7.0
Friedman #2	240	4	4140	3505	110.3	6.9
Friedman #3	240	4	0.0202	0.0164	106.5	11.5
Boston Housing	481	13	8.04	7.46	142.8	39.0
Normalized mean			1.00	0.86	1.00	0.15

1) Regression: As can be seen from the above results RVM provides more sparsity due to less number of relevance vectors required by the model. Also the RVM model generalizes better over the data-set as compared to SVM leading to relatively lower prediction error.

For the Friedman #1 data set η-RVM provided even better results by optimizing the input scale parameters of the Gaussian kernel(η). The improved performance can be attributed to the fact that the input variables which do not effect the output variable are suppressed by the low estimates of corresponding η_k

E. Classification

Data Set	N	d	SVM(err)	RVM(err)	SVM(Vec)	RVM(Vec)
Pima Diabetes	200	8	20.1%	19.6%	109	4
U.S.P.S.	7291	256	4.4%	5.1%	2540	316
Banana	400	2	10.9%	10.8%	135.2	11.4
Breast Cancer	200	9	26.9%	29.9%	116.7	6.3
Titanic	150	3	22.1%	23.0%	93.7	65.3
Waveform	400	21	10.3%	10.9%	146.4	14.6
German	700	20	22.6%	22.2%	411.2	12.5
Image	1300	18	3.0%	3.9%	166.6	34.6
Normalized mean			1.00	1.08	1.00	0.17

REFERENCES

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