Advanced Machine Learning DD2434 Final Project
The Relevance Vector Machine

Lucas Chizzali, Thomas Gaddy, Magdalini Papaioannou

January 21st, 2017

Abstract

The Support Vector Machine (SVM) is a popular supervised machine learning method used for both classification and regression problems. It is an example of a sparse kernel machine in which a subset of the training data is retained and used for prediction. Though the SVM is successfully used within the context of a wide range of applications, it does suffer from limitations. In particular, the SVM does not provide posterior probabilities and certain parameters must be determined through a computationally expensive cross-validation procedure or the like. The Relevance Vector Machine (RVM) introduced by Michael Tipping is a Bayesian approach to sparse kernel techniques that addresses some of the primary limitations of the SVM while also generating highly sparse models. In this paper, we critically examined the arguments provided by Tipping both for and against the RVM. We implemented the RVM based on Tipping's description and deployed it on a range of classification and regression problems. Additionally, we compared the RVM and SVM based on a number of criteria. Finally, we examined the state of the art in RVM research and how it has been deployed on real-life Machine Learning problems. Our implementation of the RVM gave similar results to an off-the-shelf SVM in terms of accuracy while maintaining a much sparser model, supporting Tipping's arguments in favor of the RVM.

1 Introduction

The Relevance Vector Machine (RVM) is a supervised learning technique making use of the formulation provided by a similar technique called Support Vector Machines (SVM), namely that the target values are a result of a linear combination of basis functions. More specifically, the function producing the outputs is defined as

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

where a bias term w_0 is included and N data samples are considered. Further, $K(\cdot, \cdot)$ is a *kernel* function and one is associated with every combination of data points in the data set. This formulation is identical for both SVMs and RVMs, with the exception that the RVM is not limited to positive-definite kernels.

Generally, the possibility of noise in the data is accounted for by introducing a normally-distributed error term, ϵ:

$$t_n = y(\boldsymbol{x}_n; \boldsymbol{w}) + \epsilon_n \quad where \quad \epsilon \sim \mathcal{N}(0, \sigma^2) \tag{2}$$

Assuming independent outputs with the same distribution of noise, the complete data likelihood is a product of each target's probability. For regression, this amounts to a product of Gaussians while classification makes use of Bernoulli distributions with θ being the vector containing all parameters except weights:

$$p(\mathbf{t}|\mathbf{X}, \mathbf{w}, \theta) = \prod_{n=1}^{N} p(t_n|\mathbf{x}_n, \mathbf{w}, \theta) \tag{3}$$

RVM, as opposed to SVM, outputs not merely a decision but also a probability, giving the certainty of an estimation by making use of Bayesian Inference. Therefore, a prior is introduced over the parameter vector with *each* weight w_i having its own, separate hyper-parameter α_i.

$$p(\boldsymbol{w}|\boldsymbol{\alpha}) = \prod_{i}^{N+1} \mathcal{N}(w_i \mid 0, \alpha_i^{-1}) \tag{4}$$

This is known as the automatic relevance determination (ARD) prior and lends its name to the RVM. Maximizing the evidence will drive many of these hyper-parameters to infinity, resulting in their respective weights being peaked at zero and creating a sparse model, thus curtailing over-fitting. Given the Bayesian framework, hyper-priors are introduced over $\boldsymbol{\alpha}$ and β, where β is the precision σ^{-2}. These hyper-priors are generally formulated as Gamma distributions due to their conjugacy with Gaussian distributions for a known mean and unknown precision.

The parameter values for these priors can be set to small positive values to indicate uninformative priors or to zero to give improper and uniform hyper-priors, the latter simplifying the approximation technique. Given the definition of likelihood and hierarchical prior, the posterior is derived as a combination of the weight and hyper-parameter posterior. Derivation of the latter component cannot be performed analytically in full. However, evidence approximation methods allow the posterior to be solved with respect to the parameters of interest in an iterative fashion.

In case of regression, the posterior over the weights has a closed form solution, since it is a convolution of Gaussians with mean $\boldsymbol{\mu}$ and variance $\boldsymbol{\Sigma}$. This leaves the optimization of the hyper-parameters' posterior to be conducted using an approximation. Classification is similar to the regression framework. The linear combination of basis functions is transformed by a logistic sigmoid function to give target values in the range $[0, 1]$. The main complicating factor in considering classification is added complexity due to the inability to integrate out the weights analytically. As a result, no closed-form solutions for either the weight posterior or the marginal likelihood is available. This problem is addressed by using a Laplace approximation. The approximate marginal likelihood is subsequently maximized as in the regression case, leading to re-estimated values for α [1].

At the termination of the approximation process, probabilities of predictions for new, unseen data are evaluated using the weights and training data corresponding to the relevant vectors in combination with the optimized parameters θ_{MP}:

$$p(t^*|\boldsymbol{t}_{rv}, \boldsymbol{x}^*, \boldsymbol{X}_{rv}, \boldsymbol{w}, \theta_{MP}) \tag{5}$$

Again, these predictions follow a Gaussian distribution in the regression setting and a Bernoulli distribution in case of classification.

2 Methods

2.1 Regression

Our implementation of the RVM follows from Tipping's descriptions ([2], [3]). We used the direct, fixed-point iteration approach since Tipping argues that it gives faster convergence than an Expectation-Maximization

approach at the expense of technically not guaranteeing to locally maximize the objective function, although this is not an issue in practice as Tipping found. The parameters $\boldsymbol{\alpha}$ and σ^2 were initialized with small positive values. At each iteration, the mean and variance of the weights' posterior were derived using Equation 12 and 13 in Tipping's paper [2] and subsequently used to update $\boldsymbol{\alpha}$ and σ^2 (see Equation 16 and 17 in [2]). At the end of each iteration, the relevance vectors with $\boldsymbol{\gamma}$-values above a set threshold—we used 2.22×10^{-16}—are pruned out since their corresponding weights are sharply peaked at zero. Convergence was defined as when the maximum difference between current and previous $\boldsymbol{\alpha}$-values fell below 10^{-5}. At this point, the design matrix between each test data point and data training points corresponding to the relevant vectors was calculated and multiplied with the optimized weights as in Equation 21 in [2]. The result is the predictive mean while the predictive variance was calculated using Equation 22 in [2]. The implementation followed from these equations.

2.2 Classification

The implementation for RVM classification followed from our regression implementation. Again, the first step is to initialize the hyper-parameter vector $\boldsymbol{\alpha}$. At these $\boldsymbol{\alpha}$ values, the mode of the posterior distribution ($\mathbf{w_{MP}}$) is found, which can be accomplished with an iterative optimization technique. Our RVM classification implementation uses the Newton-CG method from scipy.optimize, which is effective since we know the form the Hessian (Equation 12 in [3]). Once the optimization is converged, the negative Hessian is the inverse of the covariance $\boldsymbol{\Sigma}$ of the Gaussian approximation to the posterior. From this, $\boldsymbol{\alpha}$ can be updated exactly as in the regression case, substituting $\boldsymbol{\mu}$ with $\boldsymbol{w}_{MP}$. Weights were pruned once their α-values were greater than 10^9 and convergence was defined as when the maximum change of all α_i-values was less than 10^{-3}.

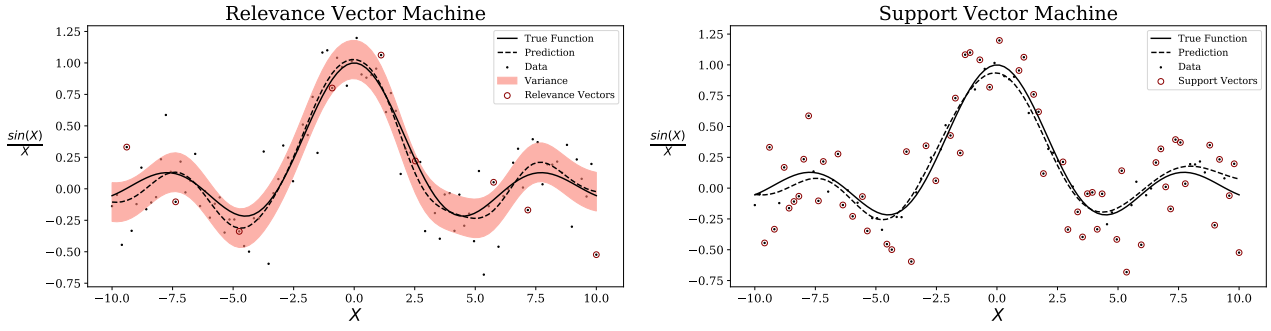
2.3 Comparison with SVM

As noted, RVM is built on the same framework as SVM. Both models are non-parametic models using kernel functions and thus have the same functional form. Therefore, SVM provides a comparable base line to investigate the arguments put forth by Tipping and is implemented using an opensource framework provided by *scikit-learn* [4]. It is a widely used, fast, off-the-shelf SVM that aids the verification of the results obtained by our RVM implementation as well as provides some insight into scaling properties. Whenever RVM is compared to SVM, the same kernel is used to facilitate fair comparisons while kernel parameters are optimized separately unless noted otherwise. Further, the remaining free parameters of SVM, such as the regularization parameter C and ϵ-insensitivity parameter are evaluated using cross validation to guarantee improved performance and a fairer comparison.

3 Results

3.1 Regression on a synthetic dataset

To validate the implemented RVM algorithm, the method was deployed on a toy example of a noisy sinc(x) function, similar to the regression examples in [2]. Comparing Figure 1 to Figure 2 in [2] shows strong similarities and thus validates the implementation. Additionally, comparing to results obtained from SVM verifies the higher sparsity property of RVM.



9 Relevance Vectors; MSE* = 0.0032

60 Support Vectors; MSE* = 0.0030

Figure 1: RVM vs. SVM on sinc(x) data set with $N = 100$ data points and noise ϵ drawn from $\mathcal{N}(0, 0.2)$ using radial basis function kernel with length-scale $\gamma = 0.35$. SVM uses 5-fold cross-validation with 30 data points for validation to estimate free parameters. *MSE refers to Mean Squared Error.

3.2 Classification on a synthetic dataset

Our implementation was validated on a synthetic dataset in which one class was sampled from a single Gaussian and slightly overlapped with a second class sampled from a mixture of Gaussians, similar to the toy example in [2]. Additionally, the results were compared to the use of an SVM on the same problem. Comparing Figure 2 to Figure 3 in [2] we see very similar results, validating the implementation. Compared to the SVM implementation, one can immediately see two major advantages of the RVM: the high degree of sparsity (4 versus 36 vectors) as well as probabilities—indicated by contours—rather than a hard decision boundary.

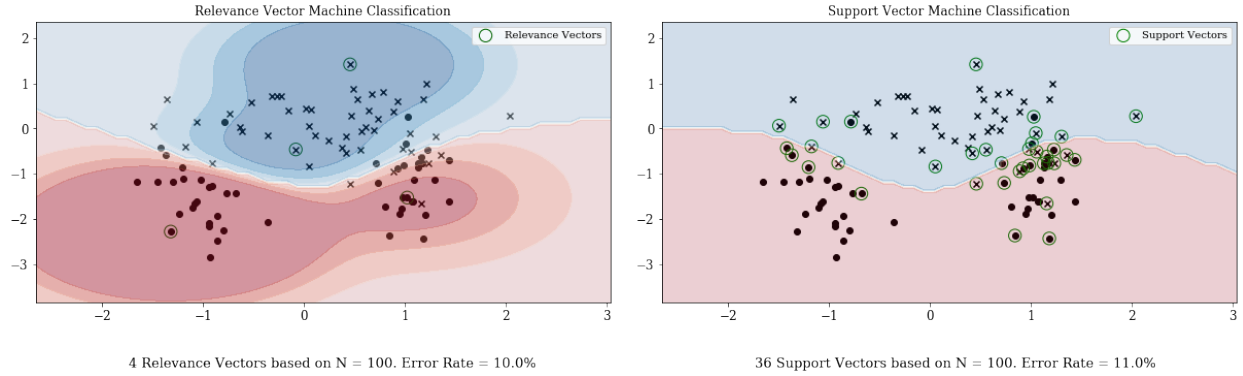


Figure 2: RVM and SVM classification performance on a synthetic dataset with $N = 100$ consisting of two classes, one generated by a single Gaussian ($\times$) and the other by a mixture of two Gaussians ($\bullet$). Both methods used a radial basis function kernel with $\gamma = 1$. The remaining free parameters for SVM were determined via 5-fold cross validation with 30 samples for validation.

3.3 Performance on Selected Problems

The RVM model was also deployed on a selection of classification and regression problems in order to further validate the implementation by comparing with the results presented in [2] and evaluate properties when compared to SVM. Different kernels and combinations of parameters were tested to find the best performing setting for each dataset and model using training and validation sets. The choice of kernel however, was the same for each model given a dataset. The results presented in the tables were averaged over repeated experiments on the same data due to slight variations in convergence for classification, stemming from the implementation of the Laplace approximation. Additionally, the regression datasets were scaled such that the inputs and outputs were in the range $[0, 1]$ as the time to perform the kernel calculations for the SVM was prohibitive.

		Error*		Vectors	
Dataset	$N_{tr} \times d$	SVM	RVM	SVM	RVM
Boston Housing	300×13	0.24	0.23	293	18
Friedman 1	300×10	0.10	0.13	193	44
Friedman 2	300×4	0.00015	0.00012	101	12
Friedman 3	300×4	0.06	0.07	191	48
Normalized Mean		1.00	1.07	1.00	0.16

Table 1: Regression; *Mean Squared Error

		Error*		Vectors	
Dataset	$N_{tr} \times d$	SVM	RVM	SVM	RVM
Pima Diabetes	500×7	28%	23%	161	2.8
Breast cancer	300×30	7%	11%	70	3
Banana	300×2	32%	34%	137	5.5
Waveform	500×21	15%	18.5%	91	19
Normalized Mean		1.00	1.17	1.00	0.08

Table 2: Classification; *Misclassification Rate

From Table 1 and 2 we can see that RVM and SVM give comparable results in terms of accuracy, with classification performing slightly worse. The main point of interest, however, is the dramatic increase in sparsity for the RVM compared to the SVM. We should also note that the presented results are similar to the results presented in [2], keeping in mind differences in the datasets such as size and dimensionality. Our classification RVM seemed to perform slightly worse relative to the SVM but with a much higher degree of sparsity compared to Tipping. It is possible in this case that we over-pruned the weights and as a result sacrificed some accuracy. In summary, our implementation of the RVM confirms Tipping's argument that the RVM achieves similar levels of accuracy as the SVM while maintaining a much sparser model.

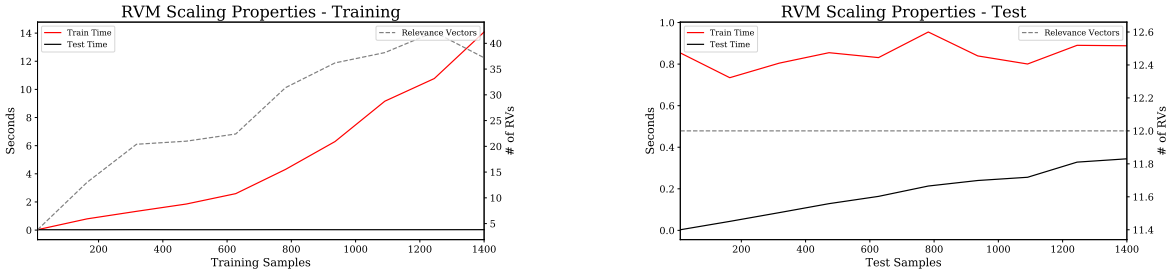
4 Discussion

4.1 Scaling Properties

RVMs constrain the use of basis function more strongly than SVMs by virtue of their probabilistic framework. This reduces the complexity of the decision function since only relevant vectors and corresponding weights

are needed to predict new data points. Computing time for prediction is thus generally lower ([5], [6]). However, during the process of learning weights, RVM faces a computational cost since it involves optimizing a nonconvex function as opposed to SVM. This behaviour has been widely studied in literature (e.g [7]). Therefore, training can approximately scale in the cube of the number of relevant basis functions. Specifically, there is a further speed distinction for RVM when considering regression or classification problems. While in both cases we perform an evidence approximation to determine the hyper-parameters, the classification case requires an additional approximation for the weight posterior since we cannot integrate analytically over the weights. This is not the case for SVM. Additionally, the Type-II Maximum Likelihood procedure used by RVM has scaling properties that worsen as the number of classes in polychotomous problems increases [8]. Modifications of RVMs have been proposed to alleviate these shortcomings. Further, the choice of an iterative approximation algorithm considerably influences RVM's speed of convergence. As noted by Tipping [2], Fixed-Point Iteration is faster than an EM approach, although the former does not guarantee local maximisation of the objective function. Another approach named Sparse Bayesian Learning is significantly faster [9]. Finally, SVMs require cross-validation or similar methods to curtail overfitting while RVMs do not due to their probabilistic framework. Hence, training SVMs requires dividing available data into training and validation sets in order to estimate the optimal free parameters, such as regularization and ϵ-insensitivity parameters. Depending on the extent of possible values under consideration or the technique applied, the training phase may be more computationally intensive than for a comparable RVM set-up. However, some versions of RVM do use separate optimization algorithms for kernel parameters (e.g [10]) and hence SVM's disadvantage in this regard is reduced. Summarizing, the comparable sparsity of RVM gives it a computational edge over SVM when predicting new input. However, it is less clear whether SVM's computational advantage of optimizing convex functions is sufficient to balance out its need for cross-validation, even in the regression setting.

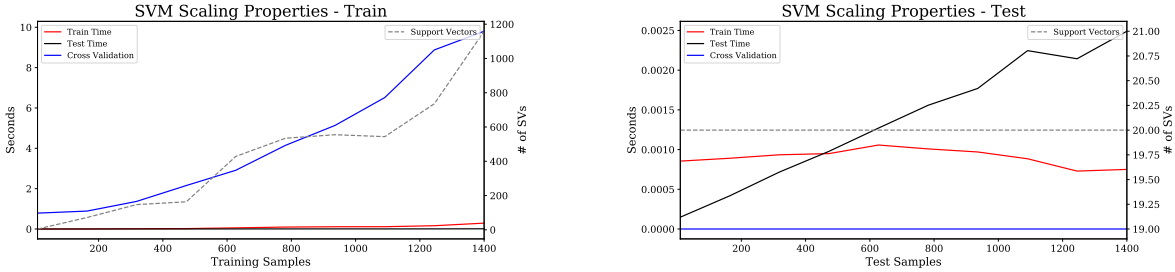
To investigate the scaling properties in a practical setting, an experiment was conducted using data created by $\text{sinc}(x)$ and using radial basis function kernel with a fixed length-scale. The resulting comparisons are not just a product of theoretical differences, but also due to programming differences, such as vectorization and the partial use of more efficient programming languages in the *scikit-learn* version of SVM [4]. However, some interesting aspects are nonetheless observable. First, as noted earlier, cross-validation can take a considerable amount of computing time, especially the larger the range of values under consideration for each parameter. During the experiment, only 10 values for each parameter C and ϵ-insensitivity were considered. Though several authors state that RVM scales worse than SVM ([2], [7]), we find that within a small range of sizes for training data (below $N = 1,400$), RVM using First-Point Iteration can outperform a state-of-the-art SVM when considering the need for cross validation. Second, the test time for the RVM increases much more slowly than for the SVM as the number of test points increases. Third, the ratio of training to testing time shows that bottle neck of RVM lies indeed in the training phase. Lastly, Figure 4a verifies that the number of support vectors grows rapidly with the number of data points available for training. This increase is less pronounced for RVM due to its sparsity properties.



(a) Scaling of training data

(b) Scaling of test data

Figure 3: RVM scaling properties averaged over 5 repetitions



(a) Scaling of training data

(b) Scaling of test data

Figure 4: SVM scaling properties averaged over 5 repetitions

4.2 Interpreting Probabilistic Outputs

As a result of its Bayesian approach, the RVM naturally gives rise to probabilistic outputs; we have a prior of weight parameters, and the Type-II Maximum Likelihood approach for hyper-parameter values allows the generation of a predictive distribution for new inputs. On the other hand, the SVM gives "hard" binary decision. This limitation of SVMs can be especially problematic in contexts where there are asymmetric classification costs or if a classifier is making a small decision which must then be combined with other outputs for an overall decision [11]. When SVMs first started gaining traction within the Machine Learning community many researchers had difficulty placing them within a probabilistic framework. Some even went so far as to believe that maximum margin SVM learning and probabilistic models constituted two separate sub-domains of machine learning [12].

In order to address this major limitation of the SVM, researchers have attempted not only to get posterior probabilities from SVMs via post-processing, but also to interpret SVMs within a larger probabilistic framework. One of the first attempts to extract posterior outputs from a SVM was performed by Platt [11] and involves training a SVM normally and then training the parameters of an additional sigmoid function to map the SVM outputs into probabilities. This method yields posterior probabilities while maintaining the sparseness of the model, unlike previous probabilistic kernel machines which are not sparse. Platt notes that this method requires additional hold-out and cross-validation procedures in order to determine the values of the parameters for the sigmoid. Though this approach is common, Tipping argues that the estimates are unreliable and that sigmoid output is not necessarily a good approximation of the posterior probability [2]. In particular, he states that the output from an SVM cannot be a reliable model of the 'inverse sigmoid' of the posterior probability. While the RVM does a good job approximating the true log-odds for membership in overlapping classes, the SVM fails to capture the shape of the log-odds and thus is over-confident about its predictions near the decision boundary. Having more accurate estimates of the posterior probability—especially near the decision boundary—can be critical for many applications.

More fundamentally, perhaps, is the issue that Platt's approach fails to place the SVM within a probabilistic framework [12]. Many researchers have pointed out the correspondence between SVMs and Gaussian Processes, leading to the interpretation of SVMs as *maximum a posteriori* solutions with Gaussian Process priors [13]. This framework of the SVM kernel as a prior over functions on the input space both provides insight into SVM kernels and makes it possible to use Bayesian methods. In particular, it provides methods on how to tune hyper-parameters—including C and kernel parameters—since it allows the definition of a properly normalized evidence. In addition, it enables one to obtain predictive class probabilities, though the procedure to do so can involve a computationally expensive Monte Carlo procedure [14]. Although there are ways of both obtaining probabilistic outputs from an SVM and interpreting the SVM within a probabilistic framework, the RVM lends itself much more naturally to probabilistic inferences. Applications in which the posterior probability is critical—such as medical diagnosis—might therefore be better suited to the RVM.

4.3 Extensions and Applications of RVM

Given the advantageous and disadvantageous characteristics of RVM outlined in previous sections, several modifications and novel implementations exist.

Polychotomous Classification

To address the scaling properties of Type-II Maximum Likelihood for polychotomous classification problems, multiclass multi-kernel relevance vector machines (mRVMs) are introduced to estimate target probabilities by means of multinomial probit likelihood [8]. A further advantage of this modified framework is the ability to use different kernels for a set of heterogeneous features that are then combined in a convex fashion to make predictions. This is different from SVMs, where heuristically combining multiple classifiers, such as directed acyclic graph SVM or one-against-one methods, are the most practically suitable approaches [15].

Scaling

Tipping and Faul propose an improved convergence algorithm named Sequential Sparse Bayesian Learning, which aims at reducing the required training time of RVMs by iteratively adding relevant vectors instead of pruning them [9]. Hence, the deletion of basis functions occurs analytically but contributes to the increased greediness of the algorithm, which according to the authors did not materialize in a reduction of prediction quality. Overall, the authors find a considerable speed increase of roughly 17-fold in some instances.

Another approach of speeding up computations is to reduce the amount of relevant features or weights beforehand. Wei et al. implement a computationally simpler, linear RVM classifier to eliminate obvious, non-relevant background pixels of breast cancer mammograms before running a more complex RVM [5]. Using this set-up, the authors reduced the time for classifying new mammograms almost 5-fold compared to a single RVM and roughly 33-fold compared to a SVM. While this approach did not reduce the computation time for training, it did allow quicker real-time classifications.

Applications

One interesting use of RVM is the spatial prediction of landslides with the support of the Cuckoo Search Optimization (CSO) [10]. As opposed to Tipping's proposition, kernel parameters are optimized using CSO. However, computation times for training and testing are not discussed or compared to an RVM that optimizes kernel parameters by itself. The results of the authors suggest that RVM is a capable tool for spatial predictions, even compared to Artificial Neural Networks due to yielding probabilistic interpretation, but note that the unbalanced classification problem makes generalizations to the real-world difficult.

Camps-Valls et al. use RVM to relate light reflected by the ocean's surface to chlorophyll concentration [7]. They employ a modification of the standard RVM wherein each basis function has its own kernel parameters constraint by biophysical knowledge. Hence, making use of RVM's property to use more customizable kernels and estimate parameters itself. Compared to SVM and Neural Networks, the modified RVM provides as accurate results with less biased estimators and higher sparsity. However, the authors note the much higher computational time required for RVM in the training phase.

RVMs have also been used for clinical data. Researchers classified eyes as healthy or as having glaucoma using both RVMs and SVMs [16]. They found that the performance of both methods were similar and an improvement upon existing classification methods. They note, however, that a binary decision may not be appropriate in such a context since it is hard to have a meaningful cutoff between patients with and without glaucoma. They also found probabilistic estimates from RVMs to be more reliable than those obtained from SVMs via post-processing, providing some confirmation of Tipping's argument concerning probabilistic outputs from both methods.

4.4 Summary of RVM

Advantages

1. As seen, RVMs typically result in much sparser models, leading to faster computation times when predicting new data points. In the case of regression, this amounts to $\mathcal{O}(M)$ for predicting the mean and $\mathcal{O}(M^2)$ for predicting the variance of an input, where M refers to the number of relevance vectors which are typically considerably fewer than support vectors for a SVM. Further, sparsity also relates to generalization and regularization. Simpler models are less prone to overfitting.

2. Predictions from RVMs are probabilistic, whereas extensions of the SVM to give probabilistic outputs can be unreliable.

3. SVMs require a computationally expensive cross-validation procedure to estimate free parameters including kernel parameters. Several propositions exist to reduce the need for computational resources. Keerthi [17] analyzes the use of the radius/margin bound iterative technique. The hyper-parameter vector is optimized using gradient-based or conjugate-based techniques by finding the radius of the smallest sphere containing weighted kernel values. This approach leads to a significant reduction in the time compared to cross validation. Several other heuristics for evaluating the performance of a set of hyper-parameters exist, such as Xi-Alpha Bound or Approximate Span-Bound [18].

4. Kernels for SVMs must satisfy Mercer's condition and thus be positive-definite, constraining the application of potentially useful kernels. Further, RVMs facilitate direct optimization of kernel parameters. This eliminates the necessity to find the optimal value using cross-validation or some other technique, even in the case of RVMs. However, it should be noted that solutions in this particular case depend on the chosen initial parameters and how the updates are interleaved [19]. Further, computations are very complex and other validation methods might be faster.

Disadvantages

1. RVM's training process involves the optimization of non-convex functions and is thus more computationally demanding than for the maximum-margin classifier SVM. Improvements in the iterative algorithm, such as the introduction of Sequential Sparse Bayesian Learning algorithm have alleviated some of the shortcomings [9].

2. RVM's hyper-parameters are parameters of a Gaussian Process covariance function, which is degenerate as so far as it assigns lower predictive uncertainty further away from training data samples [19]. This is because the prior over weights implies a weighted linear combination of basis functions. Away from these basis function, a localized prior assigns no variance outside of these localized basis functions and thus away from training data points. This attribute persists in the posterior and uncertainty far away from training samples is further reduced by the model's sparsity. To address this, the authors propose RVM* by adding one basis function centered on a given test data point. Therefore, an additional α_i and w_i is introduced and evaluated. This evaluation requires all training samples and leads to a non-sparse model. While the authors focus on regressions, they acknowledge the issue exists in a similar fashion in the classification case.

5 Conclusion

In this report, we have successfully implemented the Relevance Vector Machine proposed by Tipping on a range of toy problems and datasets for both classification and regression while investigating its advantages and disadvantages. RVM can be an effective alternative to the SVM. Particularly for scenarios in which the training data is relatively small and uncertainty estimates for prediction are required, the advantages of the RVM over the SVM are clear. In the context of "big data" or in applications for which prediction probabilities are not a priority, the RVM is likely to be a poor choice due to its increased need for computing resources. Since Tipping's publication of the first RVM, several interesting applications have been put forward that often addressed the model's limitations in a novel way, increasing its usability.

References

[1] C. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[2] M.E. Tipping. Sparse bayesian learning and the relevance vector machine. *Journal of machine learning research*, 1(Jun):211–244, 2001.

[3] M.E. Tipping. The relevance vector machine. *Advances in Neural Information Processing Systems*, 12:652–658, 2000.

[4] F. Pedregosa et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[5] L. Wei et al. Relevance vector machine for automatic detection of clustered microcalcifications. *IEEE transactions on medical imaging*, 24(10):1278–1285, 2005.

[6] B. Demir and S. Erturk. Hyperspectral image classification using relevance vector machines. *IEEE Geoscience and Remote Sensing Letters*, 4(4):586–590, 2007.

[7] G. Camps-Valls et al. Retrieval of oceanic chlorophyll concentration with relevance vector machines. *Remote Sensing of Environment*, 105(1):23–33, 2006.

[8] T. Damoulas et al. Inferring sparse kernel combinations and relevance vectors: an application to subcellular localization of proteins. In *Machine Learning and Applications, 2008. ICMLA'08. Seventh International Conference on*, pages 577–582. IEEE, 2008.

[9] M.E. Tipping and A.C. Faul. Fast marginal likelihood maximisation for sparse bayesian models. In *AISTATS*, 2003.

[10] N. Hoang and D. Tien Bui. A novel relevance vector machine classifier with cuckoo search optimization for spatial prediction of landslides. *Journal of Computing in Civil Engineering*, 30(5):04016001, 2016.

[11] J. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in Large Margin Classifiers*, 10:61–74, 1999.

[12] B. Schölkopf A. Zien, V. Franc. Support vector machines as probabilistic models. *International Conference on Machine Learning*, pages 665–672, 2011.

[13] P. Sollich. Probabilistic methods for support vector machines. *Neural Information Processing Systems*, 1999.

[14] P. Sollich. Bayesian methods for support vector machines: Evidence and predictive class probabilities. *Machine Learning*, 46:21–52, 2002.

[15] C. Hsu and C. Lin. A comparison of methods for multiclass support vector machines. *IEEE transactions on Neural Networks*, 13(2):415–425, 2002.

[16] C. Bowd et al. Relevance vector machine and support vector machine classifier analysis of scanning laser polarimetry retinal nerve fiber layer measurements. *Invest Ophthalmol Vis Sci*, 46(4):1322–1329, 2005.

[17] S. Keerthi. Efficient tuning of svm hyperparameters using radius/margin bound and iterative algorithms. *IEEE Transactions on Neural Networks*, 13(5):1225–1229, 2002.

[18] K. Duan, S. Keerthi, and A.N. Poo. Evaluation of simple performance measures for tuning svm hyperparameters. *Neurocomputing*, 51:41–59, 2003.

[19] C.E. Rasmussen and J. Quinonero-Candela. Healing the relevance vector machine through augmentation. In *Proceedings of the 22nd international conference on Machine learning*, pages 689–696. ACM, 2005.