The Relevance Vector Machine by Michael E. Tipping (2000)

KTH Royal Institute of Technology
Group Project: DD2434 Machine Learning Advanced

Authors: Sandra Picó Oristrell, Miquel Corominas Larsson,
Anna Hedström, Evangelia Gogoulou
Supervisor: (Dr) Pawel Herman

January 21, 2018

This work is a reference implementation of Michael E. Tipping's (2000) original paper on relevance vector machine (RVM) [4], which in many ways have been viewed as a Bayesian analogue to the Support Vector Machine (SVM). In our attempts to replicate the method and apply it on the same data sets, we achieved comparable results. However, in a subset of the data sets we noted subtle deviations, which triggered additional exploration of the potential sources of uncertainty in the model. Thus, in efforts of empirically evaluate this, we first identified the primary uncertainties (being the input scale parameters of the chosen kernel functions) and thereafter tested how different values of the input scale parameters, would effect the performance of the RVM (defined as prediction error rate and number of relevance vectors), being optimised by implementing a cross-validation. In line with previous literature, our findings indicate that the RVM is inherently sensitive to changes in such scale parameters, and that considerable performance gains can be achieved when optimising those. In the light of our re-implementation, we also critically examine Tipping's arguments in favour of the RVM in comparison with the SVM.

Keywords: RVM; SVM; Sparse Kernel Machines; Bayesian Statistics

1 Introduction

Since the article was originally published, the RVM has unarguably become the Bayesian cousin of the SVM. While the techniques share many characteristics for regression and classification, several advantages to avoid the limitations of SVMs has been put forward in favour of RVM. For example, to oppose the SVM's frequentist treatment of providing points estimates/ decisions as outputs, that comes in forms of a single weight vector maximising a certain objective function, the RVM produces posterior probabilities after the optimisation procedure. Essentially this means that in the spirit of a true Bayesian, one instead generates a distribution of weights, where the coefficients of the kernel expansion, that is the vector of parameters, are being treated as random variables (read more about this in the Discussion section).

The interestingness of the RVM arguably also lies in the other principal advantages that the Tipping article presents. These foremost include the sparsity of the solution and the fact that one do not need to carefully select a complexity parameter C (in avoidance of overfitting). In general terms, the sparsity is obtained by the RVM's ability to more quickly prune the basis vectors against some pre-determined threshold, which according to Tipping make it run much faster on test data while maintaining comparable generalisation errors with SVM. In RVM, one incrementally update the set of basis vectors until convergence, whereas in SVM, the number of support vectors tend to grow linearly with the size of the training set. Interestingly, in contrast to SVM where the non-zero weights (or support vectors) typically lies in close to the decision boundary, RVM offers more ”prototypical” examples (termed relevance vectors) and often lie in midst of ”apparent” clusters.

In a wider context, one might credit the development of RVM to the current surge in a probabilistic ”Bayesian” perspective in the machine learning community, e.g, as exemplified by Murphy (2012), Bishop (2006). The idea to propagate uncertainty, explicitly incorporate a prior and more generally, search for indicators of the plausibility of a model is undeniably an attractive proposition as such.

1.1 Scope, Objectives and Structure

Scope, Objectives. For this report, we aim to reproduce the work of Tipping [3] [4] and moreover discuss and critically examine identified sources of uncertainty. In the light of this, our focus have been to empirically carry out tests to investigate the magnitude to which the RVM is sensitive to different initialisations of the input scale parameter, $gamma$. To achieve this, we reported error rate and number of vectors as a basis for discussion.

Structure. The report is organised as follows: We first introduce the reader to the Methods and Implementation section, where we formulate and comment on the algorithmic implementation of the RVM. We thereafter describe and expound on the obtained results, in the light of the influence of input scale parameters (that are arbitrarily chosen and not incrementally obtained through the system). Finally, in the Discussion section we reflect on possible extensions and future improvements, and conclude giving some insight on the accomplished work.

2 Methods and Implementation

2.1 Algorithmic formulation

We have implemented the RVM as described in the original paper. For overview, the algorithmic formulation of both RVM Classification and Regression have been presented on the next page.

Algorithm 1 Pseudo code Regression

```
procedure REGRESSIONTRAINING($Xtrain$, $ytrain$, $gamma$)
    Define size($alpha$)=size($Xtrain$)+1 and Initialize $alpha$ with 1
    Define size($mu$)=size($Xtrain$)+1 and Initialize $mu$ with 0
    Initialize $sigma$ with 50
    Define size($idx$)=size($alpha$) and Initialize $idx$ with true
    Define $alphaThreshold = 10^9$
    for iteration $i$ do
        Update $idx = indexes(alpha < alphaThreshold)$
        Compute $mu[idx]$
        Compute $alpha[idx]$
        Compute $sigma$
    end for
    $Xrel = Xtrain[idx[1 ::]$
    $yrel = y[index[1 ::]]$
    return $mu[idx]$, $sigma$, $alpha[idx]$, $Xrel$, $yrel$
```

The exclusive reference of the algorithm presented above is Tipping's work on RVMs [3] [4]

Algorithm 2 Pseudo code Classification

```
procedure CLASSIFICATIONTRAINING($Xtrain$, $ytrain$,$gamma$)
    Define size($alpha$)=size($Xtrain$)+1 and Initialize $alpha$ with 1
    Define size($weights$)=size($Xtrain$)+1 and Initialize $weights$ with 0
    Define size($idx$)=size($Xtrain$)+1 and Initialize $idx$ with [0...N]
    Define $alphaThreshold = 10^{12}$
    for iteration $i$ do
        Compute $alpha[idx]$
        Compute $Phi$, $Phi(Xn) = [1, K(xn, x1).....K(xn, xN]$
        Compute Penalised logistic log-likelihood
        for iteration $j$ do
            Compute $Hessian$
            Compute $Sigma$
            Compute Second order Newton's method
        end for
        Update $idx = indexes(alpha < alphaThreshold)$
        Update $weights$
    end for
    return $idx$, $weights$
```

Newton's method: In order to find $\mathbf{w}_{MP}$ we used the Second-order Newton method. This method is an iterative method used to find the roots of a function. However,the steps followed to achieve this method for the classification tasks are the following ones:

1. Compute the gradient: $\nabla f(\mathrm{x}_n)$

2. Compute: $\lambda * \mathbf{H}[f(\mathrm{x}_n)]^{-1} * \nabla f(\mathrm{x}_n)$

Review of Tools. This work has been scripted in Python. While we decisively implemented the main RVM algorithm and its associated kernels from scratch, we naturally also utilised commonly used libraries such as **numpy**, **math**, **sklearn** for purposes such as cross validation optimisation and implementation of the SVM. For the variational nature of the data sets in raw form (some were .asc files, others .txt), efforts on matrix manipulation were made for importing the data using **numpy**.

2.2 Experimental Data and Tests

Data. The data sets utilised for this work is identical to that of Tipping [3] [4]. For regression, besides the Boston housing- (http://lib.stat.cmu.edu/datasets/boston. I) and Friedman functions- data, we also generated data in accordance to the Noisy sinc (with Gaussian noise and Uniform noise). For the classification tasks we worked with two-class problems which included a synthetic data set, the Pima Diabetes data set (http://www.stats.ox.ac.uk/pub/PRNN/), as well as the Banana-, Breast Cancer-, Titanic-, Waveform-, German- and Image- data sets by (http://ida.first.gmd.de/ raetsch/).

Tests. While most of the parameters in the RVM are incrementally learned by the iterative nature of the algorithm itself, we soon gleaned some understanding that other indirect parameters would influence the performance of the RVM that needed to be tested. Thus, in effort to systematically optimise the seemingly arbitrary choice of the input scale parameters for the (Gaussian) kernel function, we created a loop for different values of the width for Gaussian kernel using cross-validation. By finding what seems to be the suitable parameter for the kernel we could then also consider and report its relative influence on the overall performance.

3 Findings and Analysis

3.1 Benchmark Comparison

The results are obtained through an average of 10 datasets (similar to Tipping's implementation). In all cases the Gaussian RBF Kernel is used and its scaling parameter $gamma$ is tuned by using 5 fold cross-validation. In classification, the values of $gamma$ parameter tested range from 0.5 to 5.0 with step 0.5, while the range of $gamma$ values in regression is from 1 to 10 with step 1.

Description of tables. As seen in the respective table for regression and classification, we computed the mean error and number of support vectors for each dataset. Regarding classification, the error is computed in terms of the number of misclassified points (%), while the mean error in regression refers to the mean root square error. For each data set, the number of training examples (N) and the number of input variables (d) are given. Note that the absolute differences in the table are calculated with respect to the original work of Tipping (2000) and not with regards to the differences between the SVM and RVM in our re-implementation.

3.1.1 REGRESSION

			- Gamma -		**- RMSE -**			**- Vectors -**		
Data set	**N**	**d**	**SVM**	**RVM**	**SVM**	**RVM**	**Diff**	**SVM**	**RVM**	**Diff**
Sinc (Gaussian noise)	100	1	1.0	1.0	0.077	0.058	0.268	15.0	7.0	-0.3
Sinc (Uniform noise)	100	1	1.0	1.0	0.073	0.050	0.137	15.0	10.0	-3.0
Friedman #1	240	10	1.0	2.0	3.468	5.438	-2.638	229.0	219.0	-159
Friedman #2	240	4	3.0	1.0	405.84	387.129	3117.8	240.0	238.0	-231.1
Friedman #3	240	4	4.0	4.0	0.341	0.38	-0.3636	201.0	125.0	-113.5
Boston Housing	481	13	1.0	5.0	8.38	7.694	-0.234	481.0	470.0	-431.0

Table 1: Comparison of Regression Results

From the benchmark table above it can be concluded that the RVM Regression method is capable of producing sparse regression models. In most of the datasets explored, the quality of the solution obtained with RVM Regression is better than the one obtained with SVM, in terms of both regression error and sparsity. The main reason behind the negative results obtained in the case of Friedman #1 and Friedman #3 is limited computational resources. The computational complexity is significantly increased due to the big number of iterations needed for the convergence of the RVM Regression algorithm, combined with the big number of input parameters tested for each dataset.

Decision Boundaries. Figure 1 is an example of the the relevance vector machine applied to a synthetic data set, showing the decision boundary and the data points. The relevance vectors are indicated by circles. From this one can conclude that the the RVM gives a much sparser model compared to that of a corresponding application of a SVM.

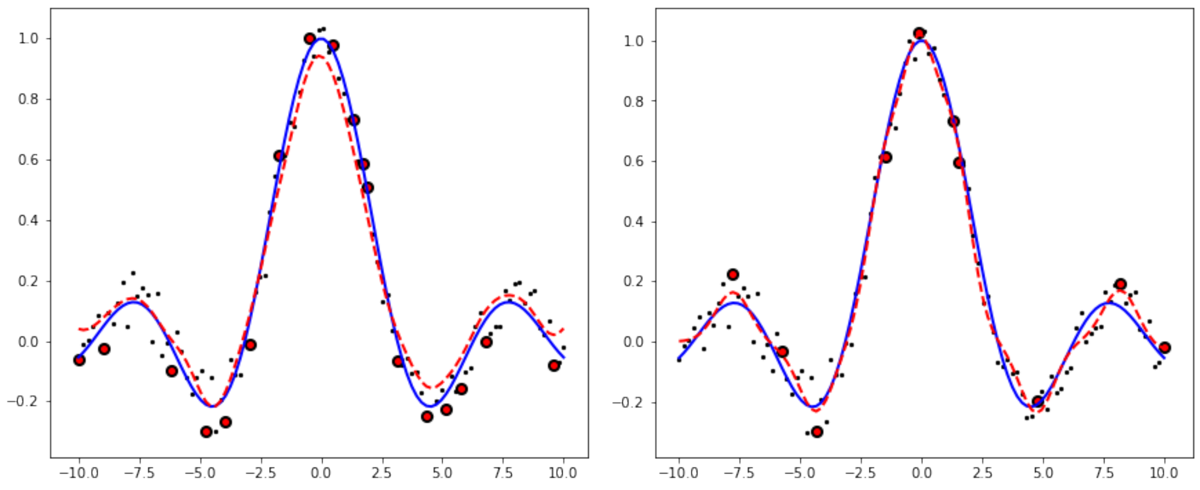

Figure 1: On the left, support, and on the right, relevance vector approximations to the sinc(x) function, which true value is represented with a continuous blue line. 100 data points have been sampled from the function with added random uniform noise, which are represented as small black dots. The approximation to the true function given by the support and relevance vectors (represented as red dots with black borders) is shown as a discontinuous red line. It is clear that for a similar approximation to the true function, support vector regression requires a higher number of support vectors than the ones needed for relevance vector regression.

3.1.2 CLASSIFICATION

			- Gamma -		**- Mean Error -**			**- Vectors -**		
Data set	**N**	**d**	**SVM**	**RVM**	**SVM**	**RVM**	**Diff**	**SVM**	**RVM**	**Diff**
Pima Diabetes	200	8	120.5	129.0	32.2	21.09	1.49	180	3.0	1.0
Ripley's Synth	100	2	0.5	0.5	11.03	13.46	4.16	47.5	6.6	2.6
Breast Cancer	200	9	2.2	4.05	31.8	34.4	4.5	181.9	8.7	2.4
Titanic	150	3	1.05	1.6	22.5	22.7	0.3	74.3	60.8	4.5
Waveform	400	21	7.0	7.35	33.0	12.9	2	400.0	13.6	1.0
German	700	20	10	13.2	26.8	26.2	4	490	17.6	5.0

Table 2: Comparison of Classification Results

In the previous table, the reader should note that there are three data sets not missing in comparison with Tipping's original work. Specifically it was the Banana, Image and U.S.P.S data set. The reason they are not included is because of the simple fact that our computer was not able to make a the cross-validation necessary for the different values of $gamma$ (through an average of 10 data sets with N = 1300). Regarding Banana data set, we found a problem in the data and we obtained a math error. And, finally, we didn't find U.S.P.S data set in the website specified on Tipping's paper.

Decision Boundaries. To compare the results as achieved in the paper, we again produced similar graphics of the decision boundary for the SVM and RVM with regards to classification. The following figures 3, 2 are examples of the the relevance vector machine applied to the synthetic- and Banana data-set, showing the decision boundary, data points and support- and relevance vectors as indicated by circles. From this, one can conclude that the the RVM gives a much sparser model compared to that of a corresponding application of a SVM. Nevertheless, regarding the Banana data-set, we can verify that the *gamma* used to generate this graph is not the optimal one. The purpose of this graph is to demonstrate the importance of the *gamma* parameter in the acquisition of satisfactorily results. This can be verified through observing the graphs as explained.

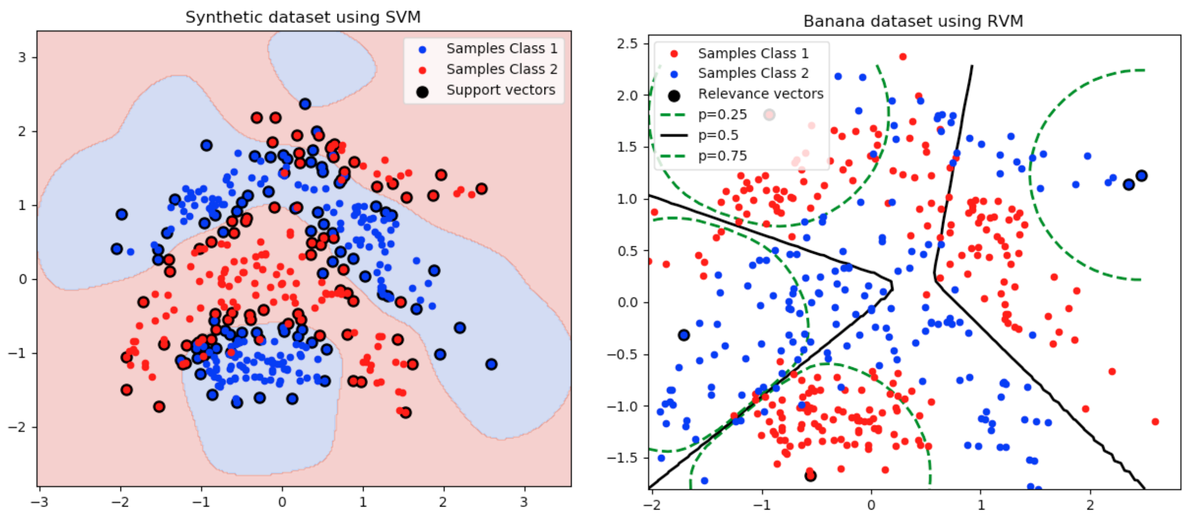

Figure 2: SVM and RVM: Decision Boundary of Banana Dataset

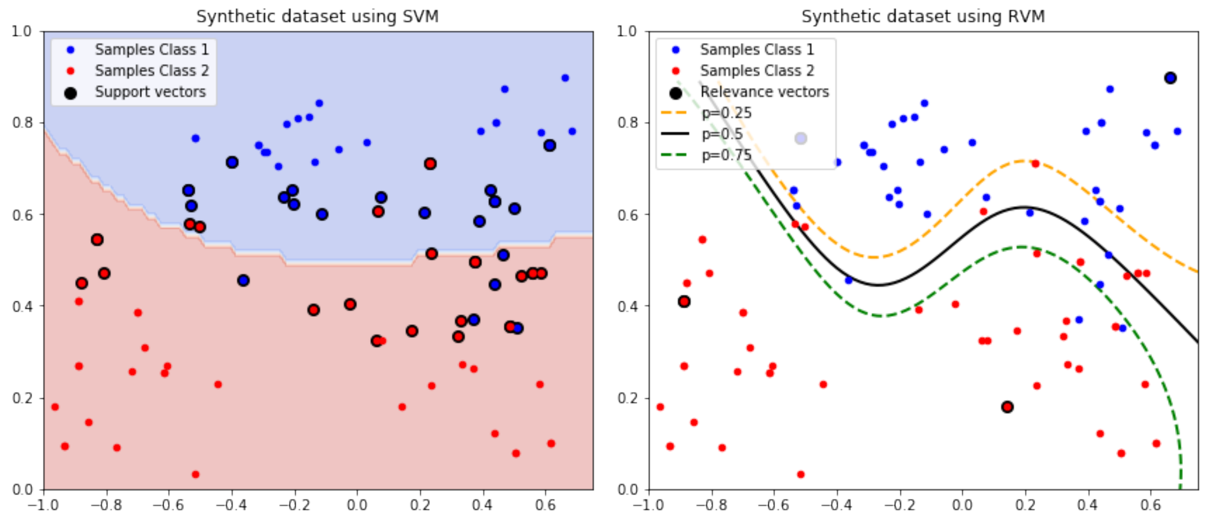

Figure 3: SVM (left) and RVM (right) classifiers on the Ripley's synthetic dataset. Data points for each class are represented as blue/red dots, with the support/relevance vectors indicated as dots with black borders. In the SVM plot, the decision boundary can be observed as the intersection between the two coloured zones. In the RVM plot, the decision boundary is indicated with a black line. Furthermore, confidence decision boundaries are additionally indicated with dashed lines.

3.2 Evaluation of Input Scale Parameters

To theoretically explain the existing differences in our reproduction of Tipping's work, we implemented tests for the input scale parameter *gamma* on each data set i (as motivated in the Methods and Implementation section) so to allow us to consider the influence of some identified/ hypothesised inherent sources of uncertainty in the model.

As displayed by the relationship between the *gamma* of the kernel and the error in the figure 4 below, there are huge performance gains to accumulate given a suitable value of the parameter. In the same vein, these findings also highlight the great extent to which the RVM is inherently sensitive to user's ability to conduct sound parameter tuning. Moreover this implies that in cases when they are not appropriately set, performance will deteriorate.

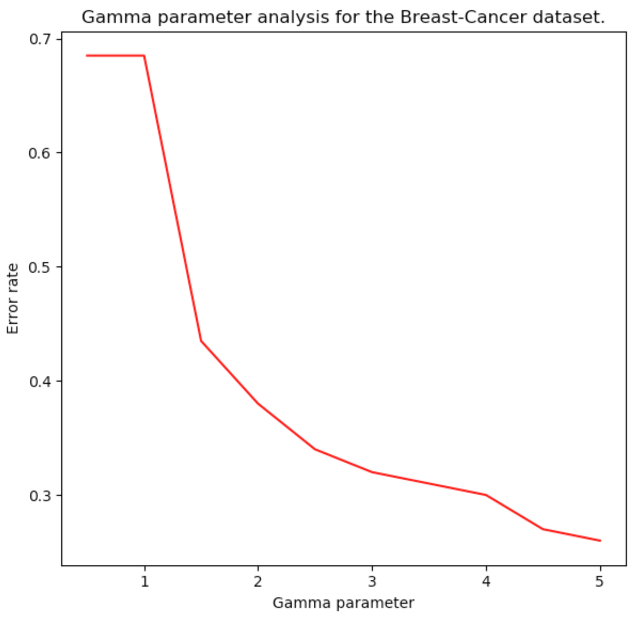

Figure 4: Gamma Parameter Analysis on the Breast-Cancer data set. The figure provides an approximate representation on the Breast-cancer data set on how ranges of the kernel width $gamma$, from very narrow width to large, influences the performance of the RVM, as measured in mean error rate (%). While this does not necessarily imply that we found the "optimal" width in terms of the model accuracy, it at least incentives efforts to optimise the parameters given the dramatic effects that the input scale parameters have.

3.3 Quality of Results.

To validate our work in respect to the original paper presented by Tipping (2000; 2001) we computed the differences, as found in tables 3.1.1 and 3.1.2. By comparing the mean error results with focus on the RVM for each data set we could conclude, more generally, that for regression; the average error across data sets were 66.7, with max error as 387.129 and min error being 0.050. The reason for the relatively large deviation could possibly be ascribed the fact that the number of input scale parameters that could be tested is huge. With that being said, cross-validation is not the best option when it comes to optimising this parameter. The problem of adapting the input scale parameter has more importantly risen in the case of Friedman 1 data set, where multiple input scale parameters exist. Moreover, for classification, the average error across data sets were +-1.25, with max error as +-4.16 and min error being 0.3. While this arguably could be seen as satisfactorily results, in order to lay out possible reasons for the deviations found, we also conducted some additional exploration with regards to the influence of the input scale parameters. Please see the next subsection for further discussion.

Additionally, a limitation was encountered due to the lack of powerful hardware to compute extensive grid-search optimisation of parameters. This is a contributing reason to the difference of results obtained compared to Tipping's paper [3].

4 Discussion

4.1 Conclusions

In this paper, we have presented, tested and evaluated the Relevance Vector Machine with respect to the influence of the input scale parameters of the kernel functions, namely $gamma$. The results obtained are arguably well aligned with the original paper, given that one incorporates the possible reasons for the deviations as outlined (recall that these have been ascribed to the identified uncertainties in the model (caused by insufficient technical details in original paper) and a lack of computational resources, especially for the Image data set).

That being said, from our findings it is clear that the method of RVM could be preferable in several regards: first, RVM is capable of producing highly sparse models, second, it has an ability to give estimates of the posterior probability for classification, and third, that most parameters (else than $gamma$) are iteratively (and automatically) generated in the model. However, throughout the re-implementation process we also faced some potential disadvantages with the RVM model, most principally being; that it has relatively long training times compared with the SVM. When the training examples number increases, this makes the algorithm as presented by Tipping less practical, given that the algorithm requires an inverse operation of O(M^3) complexity and O(M^2) memory storage (with M the number of basis functions).

Interestingly here, while Tipping put forward arguments that this long running time is nevertheless to be ”offset” by the ”avoidance of cross-validation runs to set the model complexity parameters” - we would humbly like to disagree at this point. As we in practice experienced the sensitivity of the RVM with respect to the input scale parameter and even so, were able to demonstrate the magnitude of this influence on its performance, see e.g., 4, we would lastly want to highlight the importance of this. Since there is clear necessity to optimise the input scale parameter gamma through cross-validation to actually get satisfactorily results, it is arguably problematic that there is such an understatement of it in the main paper.

4.2 Extensions and Improvements

Thus far, we have considered the RVM for binary classification problems. The formulation of the RVM classifier as a probabilistic generalised linear model implies that the extension of the proposed RVM classifier to multiple-class cases is straightforward. In that case, RVM method would constitute an important solution to a huge variety of classification problems. Another aspect of RVMs that is worth exploring is the tuning of the input scale parameters. As concluded in the Results section, cross-validation is not an option when optimising multiple scale parameters due to high computational complexity. The introduction of various optimisation techniques existing in the bibliography would dramatically improve the performance of the RVM method.

References

[1] C.M. Bishop. *Pattern recognition and machine learning*. 2006.

[2] K.P. Murphy. *Machine Learning: A probabilistic perspective* 2012.

[3] Michael E.Tipping. *Sparse Bayesian Learning and the Relevance Vector Machine* Journal of Machine Learning Research 1. 2001.

[4] Michael E.Tipping. *The Relevance Vector Machine*. Advances in Neural Information Processing Systems 12. 2000.