Relevance Vector Machine for regression and comparaison with the SVR

Hadrien Hendrikx
School of Engineering (STI)
École Polytechnique
Fédérale de Lausanne
Lausanne, Switzerland
hadrien.hendrix@epfl.ch

Gregoire Gallois-Montbrun
School of Engineering (STI)
École Polytechnique
Fédérale de Lausanne
Lausanne, Switzerland
gregoire.gallois-montbrun@epfl.ch

Louis Faury
School of Engineering (STI)
École Polytechnique
Fédérale de Lausanne
Lausanne, Switzerland
louis.faury@epfl.ch

***Abstract*—This report, written in the context of a machine learning coding project, compares two sparse regression techniques.**
After introducing some theoretical aspects on Support Vector Regression (SVR) and the Relevance Vector Regression (RVR), we introduce two datasets (an artificial one well as a real one) that will serve us as baselines for experimental comparisons between the two methods.
We will then focus the comparison on a few key features. Namely, we will study the tradeoff found between generalization and sparsity in the two methods. Issues such as robustness (behavior far from data, dataset scavenging and data chunking), computational complexity and training / testing time will also be tackled.

I. Introduction

We hereinafter focus on regression tasks : from a dataset $\{X, \mathbf{t}\} \in \mathbb{R}^{N,d} \times \mathbb{R}^N$, we wish to learn a function f such that $f(X) \simeq \mathbf{t}$.
The simplest method to perform such a task is to consider that $f(\cdot)$ is a linear function of a non-linear mapping of the input x, corrupted by a normally distributed noise of precision β

$$t = w^T\phi(x) + \eta, \quad \eta \sim \mathcal{N}\left(\eta \,|\, 0, \beta^{-1}\right) \tag{1}$$

with $\phi : \mathbb{R}^d \to \mathbb{R}^m$ a non-linear transformation. Therefore, $p(t \,|\, w, \beta) \sim \mathcal{N}\left(t \,|\, w^T\phi(x), \beta^{-1}\right)$ and the *maximum-likelihood* solution is given by the minimizer of the opposite of the log-likelihood :

$$w_{MLE} = \underset{w}{\operatorname{argmin}} \left\{ \frac{1}{2} \sum_{n=1}^{N} \left\{ w^T\phi(x_n) - t_n \right\}^2 \right\} \tag{2}$$

To avoid overfitting, a regularizer term is often added to the cost function :

$$E(w) = \frac{1}{2} \sum_{n=1}^{N} \left\{ w^T\phi(x_n) - t_n \right\}^2 + \frac{\lambda}{2}\|w\|^2 \tag{3}$$

even if some more advanced methods (Bayesian learning for instance) can also be used to avoid prevent data overfit.

II. Theoretical background

A. Support Vector machine for Regression

The Support Vector machine for Regression (SVR) extends the SVM method for regression tasks.
To obtain a sparse solution, the likelihood term in (3) is replaced by an ε-*insensitive error function* (see [7]) denoted $E_\varepsilon(\cdot)$ with :

$$E_\varepsilon(y(x) - t) = \begin{cases} 0, & \text{if } |y(x) - t| < \varepsilon \\ |y(x) - t| < \varepsilon & \text{otherwise} \end{cases} \tag{4}$$

Therefore, the quantity to be minimized can be expressed as :

$$J(w) = \frac{C}{n} \sum_{n=1}^{N} E_\varepsilon(w^T\phi(x_n) - t_n) + \frac{1}{2}\|w\|^2 \tag{5}$$

where C is a regularization parameter.
As for the SVM, one can introduce *slack variables* in order to transform this optimization program into a Quadratic Programming (QP) problem (quadratic objective, linear constraints) :

$$\begin{aligned} \min_{w,b} \quad & \frac{1}{2}\|w\|^2 + \frac{C}{n} \sum_{i=1}^{n} \left(\xi_i + \hat{\xi}_i\right) \\ \text{s.t.} \quad & \xi, \hat{\xi} \geq 0 \qquad \text{(row wise)} , \\ & w^T\phi(x_i) + b + \varepsilon + \xi_i \geq t_i, \quad i = 1, \dots, n, \\ & w^T\phi(x_i) + b - \varepsilon - \hat{\xi}_i \leq t_i, \quad i = 1, \dots, n \end{aligned} \tag{6}$$

Once the problem solved, predictions are made using :

$$y(x) = \sum_{n=1}^{N} (a_n - \hat{a}_n) k(x, x_n) + b \tag{7}$$

where we introduced the kernel $k(x, x') = \phi(x)^T\phi(x')$. The coefficients $\{a_n\}$ and $\{\hat{a}_n\}$ actually are Lagrange multipliers for the QP problem (6), and provide a *sparse* solution - only a few data points are used for regression. Those are called *support vectors*, and are so that $a_n \neq 0$ or $\hat{a}_n \neq 0$ - in other words, those that lie on the boundary or outside of the ε-tube defined by the loss function in equation (4).
As for the SVM, one can adopt a ν-SVR formulation (see

[5]) to have a *lower-bound control* on the number of retained support vectors.

B. Relevance Vector machine for Regression

The SVR therefore provides a useful tool for obtaining sparse regression machine. However, it suffers from a *number of limitations*, such as an output representation as decision rather than posterior probability, the need to estimate hyper-parameters (kernel width, penalization parameter) via *held-out methods* (like cross-validation), or the need for the kernel to be a Mercer kernel type (positive definite).

The Relevance Vector Machine for regression is a *Bayesian sparse kernel technique* that shares many of the SVR's characteristics while avoiding its limitations. It instantiates a model intended to mirror the structure of the SVR :

$$y(x) = \sum_{n=1}^{N+1} w_n k(x, x_n) \tag{8}$$

where the bias b is included in the predictor w and $k(\cdot, \cdot)$ is an arbitrary kernel (not necessarily positive definite). Assuming i.i.d data sample with Gaussian noise of precision β, the likelihood writes :

$$p(\mathbf{t} \,|\, X, w, \beta) = \prod_{n=1}^{N} \mathcal{N}\left(t_n \,|\, y(x_n), \beta\right) \tag{9}$$

The predictor w is given a centered Gaussian prior distribution :

$$p(w \,|\, \alpha) = \prod_{i=1}^{N+1} \mathcal{N}\left(w \,\middle|\, 0, \alpha_i^{-1}\right) \tag{10}$$

introducing a separate precision parameter α_i for each weight parameter w_i.

This leads to a Gaussian posterior distribution over w :

$$\begin{aligned} p(w \,|\, \mathbf{t}, X, \alpha, \beta) &= p(\mathbf{t} \,|\, w, X, \beta) p(w \,|\, \alpha) \\ &= \mathcal{N}\left(w \,|\, m, \Sigma\right) \end{aligned} \tag{11}$$

where

$$\begin{aligned} m &= \beta \Sigma \boldsymbol{\phi}^T t \\ \Sigma &= \left(A + \beta \boldsymbol{\phi}^T \boldsymbol{\phi}\right)^{-1}, \qquad A = \text{diag}(\alpha_i) \end{aligned} \tag{12}$$

In a full Bayesian approach, α and β are both given prior distributions. However, this leads to intractable computations when computing predictions. The use of *empirical Bayes* solves this problem, by approximating α and β by their maximum-a-posteriori value (also known as the *evidence approximation* or type-2 maximum likelihood).

As a result of approximation, a proportion of parameters α_i are driven to infinite values, constraining the corresponding weights w_i to have 0 mean and infinite precision, and hence are set to 0. The resulting predictions are therefore sparse in datapoints, and the inputs $\{x_n\}$ corresponding to non-zero weights are called *relevance vectors*. Once the optimal values α^* and β^* found, the predictive distribution over y can therefore be computed using α^* and β^*.

The sparsity analysis of the RVR leads to a practical algorithm for optimizing the hyper-parameters that has significant speed advantages, and is referred to as *automatic relevance determination*. The full algorithm and its justification can be found in [1] and [6].

Furthermore, the RVR provides a posterior distribution over the predictor, leading to the following *predictive distribution* (unlike the SVR which only provide a posterior decision) :

$$\begin{aligned} p(t \,|\, x, X, \mathbf{t}, \alpha, \beta) &= \int_w p(t \,|\, w, x, \beta) p(w \,|\, X, \mathbf{t}, \alpha) dw \\ &= \mathcal{N}\left(t \,\middle|\, m^T \phi(x), \phi(x)^T \Sigma \phi(x)\right) \end{aligned} \tag{13}$$

with m and Σ the first moments of the posterior (see (12))

C. Theoretical method comparaison

1) Complexity: When comparing the SVR's and the RVR's complexity, one must distinguish complexity at training time and at testing time.

Training the SVR sums up in solving a large quadratic-programing (QP) problem. A popular approach to do that implies breaking up the initial QP into smaller problems, solvable analytically, and is called *Sequential Minimization Optimization* (SMO). It requires a linear (with respect to the datapoints) amount of memory, and scales between linear and quadratic complexity in the training set size (see [4]). At test time, the complexity is linear in the number of support vectors.

Training a RVM involves optimizing a non-convex function. For a model with M basis functions, the RVM requires the inversion of a $M \times M$ matrix, which requires from $O(M^{2.7})$ to $O(M^3)$, which is as we just saw larger than the SVR's cost. However, parameters are determined automatically and in one run when training a RVR, while the hyper-parameters typically need several runs (f-fold cross-validation) to be estimated. At testing time, the RVR's complexity grows linearly with the number of relevance vectors.

2) Performance: We just named one of the major pros of using RVR - there is no need for using held-out methods to estimate hyper-parameters, as they are automatically determined through automatic relevance detection (expect for parametric basis functions). Also, it has been shown [1], [4] that RVR leads to *sparser solutions, without loss of generalization abilities* (on the contrary, the RVR usually performs better than the SVR). We will try to observe this observations in the experiments derived hereinafter.

III. Experimental results

A. Datasets presentation

1) Artificial dataset: The goal of using an artificial dataset is to be able to produce visual outputs to evaluate the performance of the two methods on a simple regression problem, hence the need to work in a one dimensional dataset.

The dataset was generated from the $sinc(\cdot)$ function, on the interval $[-5, 5]$. Points were randomly sampled from this interval, applied the $sinc$ function and added a normally distributed noise. For reasons we'll tackle later, no outliers were added in the dataset.

The following table sums up the dataset characteristics.

Dimension	Points	Support	Noise variance	Outlier
1	100	[-5,5]	0.01	No

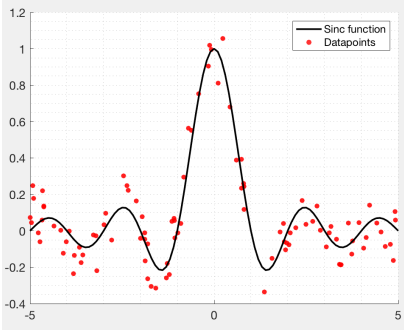

Fig. 1: The 1D artificial dataset

2) Real dataset: This dataset, known as the Airfoils Self-Noise dataset provides informations about design of various airfoils and physical environment (free stream velocity, angle of attack, chord length, frequency, suction side thickness) in order to predict the resulting sound pressure level in dB (see [**?**] for full details). The following table sums up the dataset characteristics.

Dimension	Points
5	1503

B. Implementation

We hereinafter describe different computations and results obtained on the artificial data with both the SVR and the RVR. Discussion on the two methods advantages and drawback will be held in the next section.

1) Support Vector Regression: We implemented both the ε-SVR and the ν-SVR on the artificial dataset. We used the `libsvm` library (see [2]) with a radial-basis function kernel :

$$\forall x, x', \quad k(x, x') = \exp\left(-\frac{\|x - x'\|^2}{\sigma^2}\right) \tag{14}$$

We therefore have to tune three hyper-parameters : σ (kernel width), C (penalization factor) and ν (resp. ε) for the ν-SVR (resp. ε-SVR). Figures (4) and (3) display the regressive function obtained for three given parameters for both methods on the artificial dataset. For the ν-SVR, the corresponding ε-insensitive tube was computed thanks to the values of the final Lagrange multipliers (a Lagrange multiplier strictly inferior to C indicates a point laying on the ε-tube).

2) Relevance Vector Regression: A similar work was conducted with the RVR, by using the `SparseBayes` library written by Mike Tipping. The same kernel (RBF) was used. Therefore, the only hyper-parameter to be fixed is the kernel's width σ (since automatic relevance detection automatically finds the best β and α). For regression, we plot the mean of the *predictive distribution* as well as it's standard deviation (see (13)).

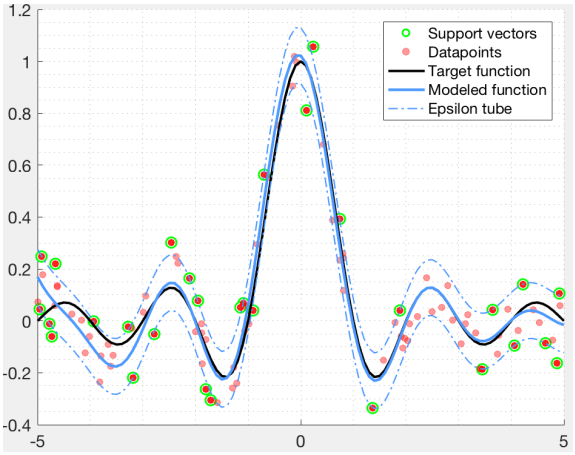

Fig. 2: ν-SVR with : $C = 100$, $\nu = 0.2$, $\sigma = 1.5$

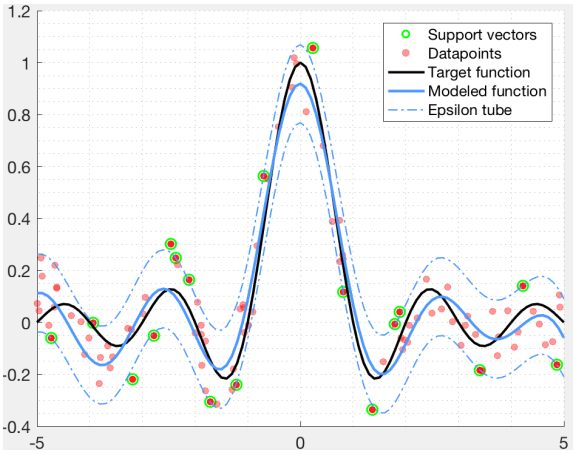

Fig. 3: ε-SVR with : $C = 50$, $\varepsilon = 0.15$, $\sigma = 2$

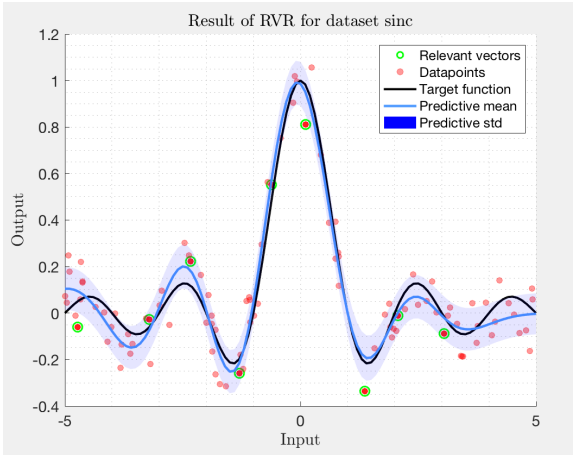

Fig. 4: RVR with : $\sigma = 1$

C. Cross-validation

1) A sparse regression evaluative metric: To infer the best hyper-parameters for all three algorithms (ϵ-SVR, ν-SVR and RVR), grid-search cross-validation is used. The metric we used to evaluate performance of the algorithms is derived from the BIC (Bayesian Information Criterion, see [1]). Among the properties of this metric is the ability to evaluate a compromise between complexity and likelihood of different models. We want to reproduce this characteristic to evaluate the performance versus sparsity tradeoff found by different models. The following lines intent to motivate the metric we chose to use.

The general formula of the BIC metric is:

$$BIC = -2ln(L) + kln(N) \tag{15}$$

where L corresponds to the likelihood of the model, k is the number of parameters of the model and N is the number of data-points. In the particular case we wish to tackle (measuring a trade-off between regression performance and sparsity), the BIC appears to be a reasonable metric, although some adjustments are needed to adapt it to a regression task.

In the case of RVR and SVR, the number of parameters k is directly proportional to the number of relevant vectors (respectively support vectors). Assuming independent, normally distributed noise, the likelihood can directly be related to the Mean Square Error (MSE) :

$$ln(L) = -\frac{N}{2\sigma^2} MSE \tag{16}$$

where σ^2 corresponds to the variance of the dataset's noise. Therefore a reasonable metric to measure the goodness of a sparse regression model would be

$$BICSR = N\frac{MSE}{\sigma^2} + N_{SV} ln(N) \tag{17}$$

where N_{SV} is the number of relevance (or support) vectors, and $BICSR$ holds for Bayesian Information Criterion for Sparse Regression. Minimizing this metric indeed implies minimizing the MSE while linearly penalizing the number of support vectors. Some other kind of penalization (quadratic, exponential) could also be used. The discussion of the choice of this penalization, as well as the validation of the well behavior of this metric is held in (IV).

We wish to emphasize the fact that this metric is not the commonly used BIC metric, but rather an adaptation of it to sparse regression (and therefore a measure of trade-off between performance (MSE) and sparsity).

2) Grid-search cross-validation: In the following, we performed 50-fold cross-validation with a 0.75 training-test ratio, using the $BICSR$ previously defined.

Figures (5) and (6) display grid search results for both ϵ-SVR and ν-SVR in the form of heat-maps. As expected both selected models are almost the same therefore showing the equivalence between both approaches. Both lead to a total of 14 support vectors.

Figure (7) displays grid-search results for the RVR over kernel width σ, the only hyperparameter for this method.

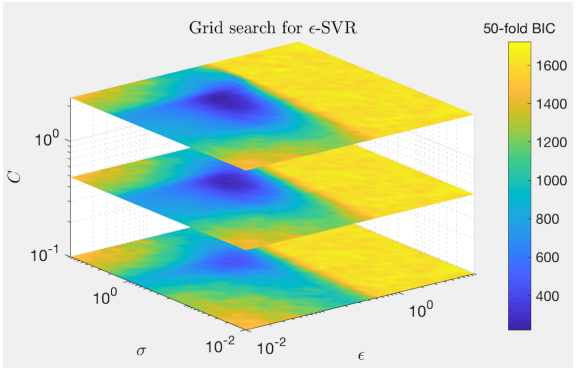

Fig. 5: Grid search for $\epsilon - SVR$ over ϵ, C and σ for sinc dataset Best hyperparameters: $\epsilon = 0.17$, $C = 2.40$, $\sigma = 1.17$

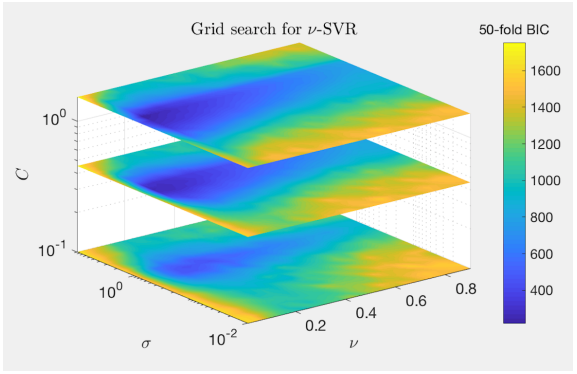

Fig. 6: Grid search for $\nu - SVR$ over ν, C and σ for sinc dataset Best hyperparameters: $\nu = 0.057$, $C = 1.52$, $\sigma = 1.13$

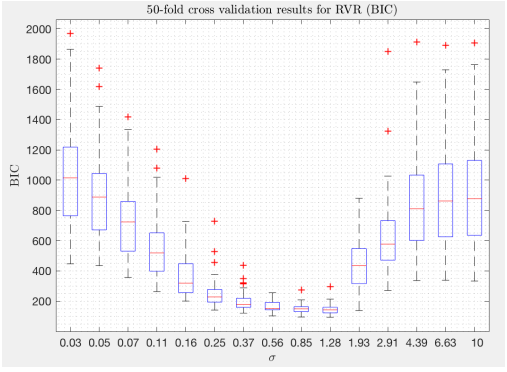

Fig. 7: Grid search for RVR over σ for sinc dataset. Best hyperparameter: $\sigma = 1.28$

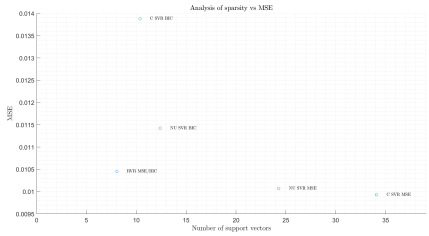

Fig. 8: Evaluation of the tradeoff between raw performance (MSE) and sparsity (number of non-zero coefficients). Points correspond to the best hyperparameters for each method and each metric.

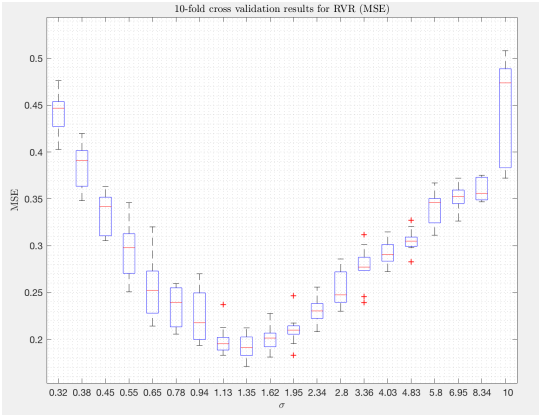

Fig. 9: Grid search for RVR over σ for airfoils dataset. Best hyperparameter: $\sigma = 1.25$

D. Real dataset

For this dataset, the lack of information concerning dataset noise prevented to use BIC-like metrics for finding best hyperparameters. We therefore performed grid search using 10-fold cross validation over Mean Square Error for the different algorithms, after standardization of data. Such metric only considers fitting of the different models with the data without taking model complexity into account. Due to higher dimensionality and to an higher number of points, training test ratio was reduced to 0.5.

Results of grid search over σ, the kernel width for Relevance Vectors Regression are displayed on figure (9). Optimal kernel width therefore corresponds to $\sigma = 1.25$ leading to an MSE of 0.19. Number of retrieved relevance vectors is 94.

IV. Discussion

V. Conclusion

References

[1] Christopher M Bishop. Pattern recognition. *Machine Learning*, 128:1–58, 2006.

[2] Chih-Chung Chang and Chih-Jen Lin. Libsvm: A library for support vector machines. *ACM Trans. Intell. Syst. Technol.*, 2(3):27:1–27:27, May 2011.

[3] Kelwin Fernandes, Pedro Vinagre, and Paulo Cortez. A proactive intelligent decision support system for predicting the popularity of online news. In *Portuguese Conference on Artificial Intelligence*, pages 535–546. Springer, 2015.

[4] John Platt. Sequential minimal optimization: A fast algorithm for training support vector machines. 1998.

[5] Bernhard Schölkopf, Alex J Smola, Robert C Williamson, and Peter L Bartlett. New support vector algorithms. *Neural computation*, 12(5):1207–1245, 2000.

[6] Michael Tipping. Relevance vector machine, October 14 2003. US Patent 6,633,857.

[7] Vladimir N Vapnik. The nature of statistical learning theory. 1995.