9 Regression Estimation

Overview

In this chapter, we explain the ideas underlying Support Vector (SV) machines for function estimation. We start by giving a brief summary of the motivations and formulations of an SV approach for regression estimation (Section 9.1), followed by a derivation of the associated dual programming problems (Section 9.2). After some illustrative examples, we cover extensions to linear programming settings and a ν-variant that utilizes a more convenient parametrization. In Section 9.6, we discuss some applications, followed by a summary (Section 9.7) and a collection of problems for the reader.

Prerequisites

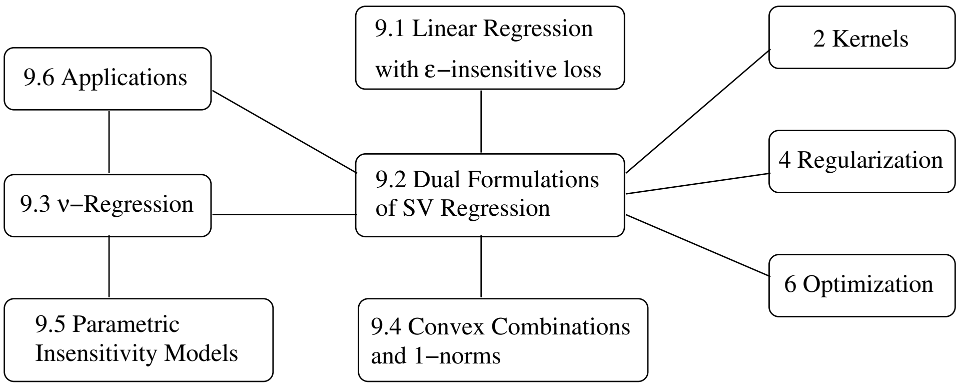
Although it is not strictly indispensable, we recommend that the reader first study the basics of the SVM classification algorithm, at least at the level of detail given in Chapter 1. The derivation of the dual (Section 9.2) is self-contained, but would benefit from some background in optimization (Chapter 6), especially in the case of the more advanced formulations given in Section 9.2.2. If desired, these can actually be skipped at first reading. Section 9.3 describes a modification of the standard SV regression algorithm, along with some considerations on issues such as robustness. The latter can be best understood within the context given in Section 3.4. Finally, Section 9.4 deals with linear programming regularizers, which were discussed in detail in Section 4.9.2.

9.1 Linear Regression with Insensitive Loss Function

SVMs were first developed for pattern recognition. As described in Chapter 7, they represent the decision boundary in terms of a typically small subset of all training examples — the Support Vectors. When the SV algorithm was generalized to the case of regression estimation (that is, to the estimation of *real-valued* functions, rather than just $\{\pm 1\}$-valued ones, as in the case in pattern recognition), it was crucial to find a way of retaining this feature. In order for the sparseness property to carry over to the case of SV Regression, Vapnik devised the so-called *ε-insensitive loss function* (Figure 1.8) [561],

ε-Insensitive Loss

$$|y - f(\mathbf{x})|_\varepsilon = \max\{0, |y - f(\mathbf{x})| - \varepsilon\}, \tag{9.1}$$


9.1 Linear Regression with ε−insensitive loss
2 Kernels
9.6 Applications
4 Regularization
9.2 Dual Formulations of SV Regression
9.3 ν−Regression
6 Optimization
9.5 Parametric Insensitivity Models
9.4 Convex Combinations and 1−norms

which does not penalize errors below some $\varepsilon \geq 0$, chosen a priori.[1] The rationale behind this choice is the following. In pattern recognition, when measuring the loss incurred for a particular pattern, there is a large area where we accrue zero loss: whenever a pattern is on the correct side of the decision surface, and does not touch the margin, it does not contribute any loss to the objective function (7.35). Correspondingly, it does not carry any information about the position of the decision surface — after all, the latter is computed by minimizing that very objective function. This is the underlying reason why the pattern does not appear in the SV expansion of the solution. A loss function for regression estimation must also have an insensitive zone; hence we use the ε-insensitive loss.

The regression algorithm is then developed in close analogy to the case of pattern recognition. Again, we estimate linear functions, use a $\|\mathbf{w}\|^2$ regularizer, and rewrite everything in terms of dot products to generalize to the nonlinear case. The basic SV regression algorithm, which we will henceforth call **ε-SVR**, seeks to estimate linear functions[2],

$$f(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle + b, \text{ where } \mathbf{w}, \mathbf{x} \in \mathcal{H}, b \in \mathbb{R}, \tag{9.2}$$

based on independent and identically distributed (iid) data,

$$(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m) \in \mathcal{H} \times \mathbb{R}. \tag{9.3}$$

Here, $\mathcal{H}$ is a dot product space in which the (mapped) input patterns live (i.e., the feature space induced by a kernel). The goal of the learning process is to find a

1. The insensitive zone is sometimes referred to as the *ε-tube*. Actually, this term is lightly misleading, as in multi-dimensional problems, the insensitive zone has the shape of a *slab* rather than a tube; in other words, the region between two parallel hyperplanes, differing in their y offset.
2. Strictly speaking, these should be called *affine* functions. We will not indulge in these fine distinctions. The crucial bit is that the part to which we apply the kernel trick is linear.

function f with a small risk (or test error) (cf. Chapter 3),

$$R[f] = \int c(f, \mathbf{x}, y)\, dP(\mathbf{x}, y), \tag{9.4}$$

where P is the probability measure which is assumed to be responsible for the generation of the observations (9.3), and c is a loss function, such as $c(f, \mathbf{x}, y) = (f(\mathbf{x}) - y)^2$, or one of many other possible choices (Chapter 3). The particular loss function for which we would like to minimize (9.4) depends on the specific regression estimation problem at hand. Note that this does not necessarily have to coincide with the loss function used in our learning algorithm. First, there might be additional constraints that we would like our regression estimation to satisfy, for instance that it have a sparse representation in terms of the training data — in the SV case, this is achieved through the insensitive zone in (9.1). Second, we cannot minimize (9.4) directly in any case, since we do not know P. Instead, we are given the sample (9.3), and we try to obtain a small risk by minimizing the regularized risk functional,

$$\frac{1}{2}\|\mathbf{w}\|^2 + C \cdot R^{\varepsilon}_{\text{emp}}[f]. \tag{9.5}$$

Regularized Risk Functional

Here,

$$R^{\varepsilon}_{\text{emp}}[f] := \frac{1}{m}\sum_{i=1}^{m} |y_i - f(\mathbf{x}_i)|_{\varepsilon} \tag{9.6}$$

measures the ε-insensitive training error, and C is a constant determining the trade-off with the complexity penalizer $\|\mathbf{w}\|^2$. In short, minimizing (9.5) captures the main insight of statistical learning theory, stating that in order to obtain a small risk, we need to control both training error and model complexity, by explaining the data with a simple model (Chapter 5).

A small $\|\mathbf{w}\|^2$ corresponds to a linear function (9.2) that is flat — in feature space. Note that in the case of pattern recognition, we use the same regularizer (cf. (7.35)); however, it corresponded to a large *margin* in this case. How does this difference arise? A related question, it turns out, is why SV regression requires an extra parameter ε, while SV pattern recognition does not.

Flatness vs. Margin

Let us try to understand how these seemingly different problems are actually identical.

Definition 9.1 (ε-margin) *Let $(E, \|.\|_E)$, $(G, \|.\|_G)$ be normed spaces, and $\mathcal{X} \subset E$. We define the ε-margin of a function $f : \mathcal{X} \to G$ as*

$$m_{\varepsilon}(f) := \inf\{\|x - x'\|_E \mid x, x' \in \mathcal{X}, \|f(x) - f(x')\|_G \geq 2\varepsilon\}. \tag{9.7}$$

Let us look at a 1-D toy problem (Figure 9.1). In pattern recognition, we are looking for a function which exceeds some constant ε (using the canonical hyperplanes of Definition 7.1, this constant is $\varepsilon = 1$) on the positive patterns, and which is smaller than $-\varepsilon$ on the negative patterns. The points where the function takes the values $\pm\varepsilon$ define the ε-margin in the space of the patterns (the x-axis in

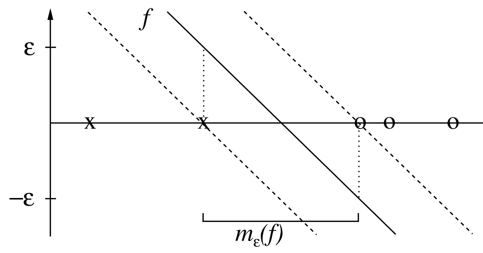


Figure 9.1 1D toy problem: separate 'x' from 'o'. The SV classification algorithm constructs a linear function $f(x) = \langle w, x\rangle + b$ satisfying the canonicality condition $\min\{|f(x)||x \in \mathcal{X}\} = 1$ (or equivalently, $\varepsilon = 1$). To maximize the margin $m_\varepsilon(f)$, we have to minimize $|w|$.

the plot). Therefore, the *flatter* the function f, the *larger* the classification margin. This illustrates why both SV regression and SV pattern recognition use the same regularizer $\|\mathbf{w}\|^2$, albeit with different effects. For more detail on these issues and on the ε-margin, cf. Section 9.8; cf. also [561, 418].

Primal Objective Function, ε-SVR

The minimization of (9.5) is equivalent to the following constrained optimization problem:

$$\underset{\mathbf{w}\in\mathcal{H},\boldsymbol{\xi}^{(*)}\in\mathbb{R}^m,b\in\mathbb{R}}{\text{minimize}} \quad \tau(\mathbf{w},\boldsymbol{\xi}^{(*)}) = \frac{1}{2}\|\mathbf{w}\|^2 + C\cdot\frac{1}{m}\sum_{i=1}^{m}(\xi_i+\xi_i^*), \tag{9.8}$$

$$\text{subject to } (\langle\mathbf{w},\mathbf{x}_i\rangle + b) - y_i \le \varepsilon + \xi_i, \tag{9.9}$$

$$y_i - (\langle\mathbf{w},\mathbf{x}_i\rangle + b) \le \varepsilon + \xi_i^*, \tag{9.10}$$

$$\xi_i^{(*)} \ge 0. \tag{9.11}$$

Here and below, it is understood that $i = 1, \ldots, m$, and that bold face Greek letters denote m-dimensional vectors of the corresponding variables; $^{(*)}$ is a shorthand implying both the variables with and without asterisks.

9.2 Dual Problems

9.2.1 ε-Insensitive Loss

The key idea is to construct a Lagrangian from the objective function and the corresponding constraints, by introducing a dual set of variables. It can be shown that this function has a saddle point with respect to the primal and dual variables at the solution; for details see Chapter 6. We define a Lagrangian,

$$\begin{aligned} L := {} & \frac{1}{2}\|\mathbf{w}\|^2 + \frac{C}{m}\sum_{i=1}^{m}(\xi_i+\xi_i^*) - \sum_{i=1}^{m}(\eta_i\xi_i + \eta_i^*\xi_i^*) \\ & - \sum_{i=1}^{m}\alpha_i(\varepsilon+\xi_i+y_i-\langle\mathbf{w},\mathbf{x}_i\rangle - b) \\ & - \sum_{i=1}^{m}\alpha_i^*(\varepsilon+\xi_i^*-y_i+\langle\mathbf{w},\mathbf{x}_i\rangle + b), \end{aligned} \tag{9.12}$$

where the dual variables (or Lagrange multipliers) in (9.12) have to satisfy positivity constraints,

$$\alpha_i^{(*)}, \eta_i^{(*)} \geq 0. \tag{9.13}$$

It follows from the saddle point condition (Chapter 6) that the partial derivatives of L with respect to the primal variables $(\mathbf{w}, b, \xi_i, \xi_i^*)$ have to vanish for optimality;

$$\partial_b L = \sum_{i=1}^m (\alpha_i - \alpha_i^*) = 0, \tag{9.14}$$

$$\partial_{\mathbf{w}} L = \mathbf{w} - \sum_{i=1}^m (\alpha_i^* - \alpha_i)\mathbf{x}_i = 0, \tag{9.15}$$

$$\partial_{\xi_i^{(*)}} L = \tfrac{C}{m} - \alpha_i^{(*)} - \eta_i^{(*)} = 0. \tag{9.16}$$

Substituting (9.14), (9.15), and (9.16) into (9.12) yields the dual optimization problem,

$$\begin{array}{ll} \underset{\boldsymbol{\alpha}^{(*)} \in \mathbb{R}^m}{\text{maximize}} & \begin{cases} -\frac{1}{2} \sum\limits_{i,j=1}^m (\alpha_i^* - \alpha_i)(\alpha_j^* - \alpha_j) \langle \mathbf{x}_i, \mathbf{x}_j \rangle \\ -\varepsilon \sum\limits_{i=1}^m (\alpha_i^* + \alpha_i) + \sum\limits_{i=1}^m y_i(\alpha_i^* - \alpha_i), \end{cases} \\ \text{subject to} & \sum\limits_{i=1}^m (\alpha_i - \alpha_i^*) = 0 \text{ and } \alpha_i, \alpha_i^* \in [0, C/m]. \end{array} \tag{9.17}$$

In deriving (9.17), we eliminate the dual variables η_i, η_i^* through condition (9.16). Eq. (9.15) can be rewritten as

$$\mathbf{w} = \sum_{i=1}^m (\alpha_i^* - \alpha_i)\mathbf{x}_i, \quad \text{thus} \quad f(\mathbf{x}) = \sum_{i=1}^m (\alpha_i^* - \alpha_i) \langle \mathbf{x}_i, \mathbf{x} \rangle + b. \tag{9.18}$$

SV Expansion

This is the familiar *SV expansion*, stating that $\mathbf{w}$ can be completely described as a linear combination of a subset of the training patterns $\mathbf{x}_i$.

Note that just as in the pattern recognition case, the complete algorithm can be described in terms of dot products between the data. Even when evaluating $f(\mathbf{x})$, we need not compute $\mathbf{w}$ explicitly. This will allow the formulation of a nonlinear extension using kernels.

Computing the Offset b

So far we have neglected the issue of computing b. The latter can be done by exploiting the Karush-Kuhn-Tucker (KKT) conditions (Chapter 6). These state that at the point of the solution, the product between dual variables and constraints has to vanish;

$$\begin{aligned} \alpha_i(\varepsilon + \xi_i - y_i + \langle \mathbf{w}, \mathbf{x}_i \rangle + b) &= 0, \\ \alpha_i^*(\varepsilon + \xi_i^* + y_i - \langle \mathbf{w}, \mathbf{x}_i \rangle - b) &= 0, \end{aligned} \tag{9.19}$$

and

$$\begin{aligned} (\tfrac{C}{m} - \alpha_i)\xi_i &= 0, \\ (\tfrac{C}{m} - \alpha_i^*)\xi_i^* &= 0. \end{aligned} \tag{9.20}$$

This allows us to draw several useful conclusions.

- First, only examples $(\mathbf{x}_i, y_i)$ with corresponding $\alpha_i^{(*)} = C/m$ can lie outside the

ε-insensitive tube (i.e., $\xi_i^{(*)} > 0$) around f.

- Second, we have $\alpha_i \alpha_i^* = 0$. In other words, there can never be a set of dual variables α_i, α_i^* which are both simultaneously nonzero (cf. Problem 9.1).
- Third, for $\alpha_i^{(*)} \in (0, C/m)$ we have $\xi_i^{(*)} = 0$, and furthermore the second factor in (9.19) must vanish. Hence b can be computed as follows:

$$\begin{aligned} b &= y_i - \langle \mathbf{w}, \mathbf{x}_i \rangle - \varepsilon \quad \text{for } \alpha_i \in (0, C/m), \\ b &= y_i - \langle \mathbf{w}, \mathbf{x}_i \rangle + \varepsilon \quad \text{for } \alpha_i^* \in (0, C/m). \end{aligned} \tag{9.21}$$

Theoretically, it suffices to use any Lagrange multiplier in $(0, C/m)$, If given the choice between several such multipliers (usually there are many multipliers which are not 'at bound,' meaning that they do not equal 0 or C/m), it is safest to use one that is not too close to 0 or C/m.
Another way of computing b will be discussed in the context of interior point optimization (cf. Chapter 10). There, b turns out to be a by-product of the optimization process. See also [291] for further methods to compute the constant offset.

- A final note must be made regarding the *sparsity* of the SV expansion. From (9.19) it follows that the Lagrange multipliers may be nonzero only for $|f(\mathbf{x}_i) - y_i| \geq \varepsilon$; in other words, for all examples inside the ε-tube (the shaded region in Figure 1.8) the α_i, α_i^* vanish. This is because when $|f(\mathbf{x}_i) - y_i| < \varepsilon$ the second factor in (9.19) is nonzero, hence α_i, α_i^* must be zero for the KKT conditions to be satisfied. Therefore we have a sparse expansion of $\mathbf{w}$ in terms of $\mathbf{x}_i$ (we do not need all $\mathbf{x}_i$ to describe $\mathbf{w}$). The examples that come with nonvanishing coefficients are called *Support Vectors*. It is geometrically plausible that the points inside the tube do not contribute to the solution: we could remove any one of them, and still obtain the same solution, therefore they cannot carry any information about it.

9.2.2 More General Loss Functions

We will now consider loss functions $c(\mathbf{x}, y, f(\mathbf{x}))$ which for fixed $\mathbf{x}$ and y are convex in $f(\mathbf{x})$. This requirement is chosen as we want to ensure the existence and uniqueness (for strict convexity) of a minimum of optimization problems (Chapter 6). Further detail on loss functions can be found in Chapter 3; for now, we will focus on how, given a loss function, the optimization problems are derived.

For the sake of simplicity, we will additionally assume c to be symmetric, to have (at most) two (for symmetry) discontinuities at $\pm\varepsilon, \varepsilon \geq 0$ in the first derivative, and to be zero in the interval $[-\varepsilon, \varepsilon]$. All loss functions from table 3.1 belong to this class. Hence c will take on the form

$$c(\mathbf{x}, y, f(\mathbf{x})) = \tilde{c}(|y - f(\mathbf{x})|_\varepsilon). \tag{9.22}$$

Note the similarity to Vapnik's ε-insensitive loss. It is rather straightforward to extend this special choice to more general convex loss functions: for nonzero loss functions in the interval $[-\varepsilon, \varepsilon]$, we use an additional pair of slack variables. Furthermore we might choose different loss functions $\tilde{c}_i$, $\tilde{c}_i^*$ and different values of ε_i, ε_i^* for each example. At the expense of additional Lagrange multipliers in the dual formulation, additional discontinuities can also be dealt with. In a manner analogous to (9.8), we arrive at a convex minimization problem [512] (note that for ease of notation, we have absorbed the sample size in C; cf. (9.8)):

$$\begin{array}{ll} \underset{\mathbf{w}\in\mathcal{H},\boldsymbol{\xi}^{(*)}\in\mathbb{R}^m,b\in\mathbb{R}}{\text{minimize}} & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^m (\tilde{c}(\xi_i)+\tilde{c}(\xi_i^*)), \\ \text{subject to} & \left\{\begin{array}{lll} \langle\mathbf{w},\mathbf{x}_i\rangle + b - y_i & \le & \varepsilon+\xi_i, \\ y_i - \langle\mathbf{w},\mathbf{x}_i\rangle - b & \le & \varepsilon+\xi_i^*, \\ \xi_i,\xi_i^* & \ge & 0. \end{array}\right. \end{array} \tag{9.23}$$

Again, by standard Lagrange multiplier techniques, using exactly the same reasoning as in the $|\cdot|_\varepsilon$ case, we can compute the dual optimization problem. In some places, we will omit the indices $_i$ and * to avoid tedious notation. This yields

$$\begin{array}{ll} \underset{\boldsymbol{\alpha}^{(*)}\in\mathbb{R}^m}{\text{maximize}} & \left\{\begin{array}{l} -\frac{1}{2}\sum_{i,j=1}^m (\alpha_i^*-\alpha_i)(\alpha_j^*-\alpha_j)\langle\mathbf{x}_i,\mathbf{x}_j\rangle \\ +\sum_{i=1}^m (y_i(\alpha_i^*-\alpha_i)-\varepsilon(\alpha_i^*+\alpha_i)) \\ +C\sum_{i=1}^m (T(\xi_i^*)+T(\xi_i)), \end{array}\right. \\ \text{where} & \left\{\begin{array}{lll} \mathbf{w} & = & \sum_{i=1}^m (\alpha_i^*-\alpha_i)\mathbf{x}_i, \\ T(\xi) & := & \tilde{c}(\xi)-\xi\partial_\xi\tilde{c}(\xi), \end{array}\right. \\ \text{subject to} & \left\{\begin{array}{lll} \sum_{i=1}^m (\alpha_i-\alpha_i^*) & = & 0, \\ \alpha & \le & C\partial_\xi\tilde{c}(\xi), \\ \xi & = & \inf\{\xi\,|\,C\partial_\xi\tilde{c}\ge\alpha\}, \\ \alpha,\xi & \ge & 0. \end{array}\right. \end{array} \tag{9.24}$$

Let us consider the examples of Table 3.1 as special cases. We will explicitly show for two examples how (9.24) can be further simplified to reduce it to a form that is practically useful. In the ε-insensitive case, where $\tilde{c}(\xi)=|\xi|$, we get

$$T(\xi) = \xi - \xi\cdot 1 = 0. \tag{9.25}$$

We can further conclude from $\partial_\xi\tilde{c}(\xi)=1$ that

$$\xi = \inf\{\xi\,|\,C\ge\alpha\} = 0 \text{ and } \alpha\in[0,C]. \tag{9.26}$$

In the case of piecewise polynomial loss, we have to distinguish two different cases: $\xi\le\sigma$ and $\xi>\sigma$. In the first case we get

$$T(\xi) = \frac{1}{p\sigma^{p-1}}\xi^p - \frac{1}{\sigma^{p-1}}\xi^p = -\frac{p-1}{p}\sigma^{1-p}\xi^p, \tag{9.27}$$

and $\xi = \inf\{\xi\,|\,C\sigma^{1-p}\xi^{p-1}\ge\alpha\} = \sigma C^{-\frac{1}{p-1}}\alpha^{\frac{1}{p-1}}$; thus

$$T(\xi) = -\frac{p-1}{p}\sigma C^{-\frac{p}{p-1}}\alpha^{\frac{p}{p-1}}. \tag{9.28}$$

Table 9.1 Terms of the convex optimization problem depending on the choice of the loss function.

	ε	α	$CT(\alpha)$
ε-insensitive	$\varepsilon \neq 0$	$\alpha \in [0, C]$	0
Laplacian	$\varepsilon = 0$	$\alpha \in [0, C]$	0
Gaussian	$\varepsilon = 0$	$\alpha \in [0, \infty)$	$-\frac{1}{2}C^{-1}\alpha^2$
Huber's robust loss	$\varepsilon = 0$	$\alpha \in [0, C]$	$-\frac{1}{2}\sigma C^{-1}\alpha^2$
Polynomial	$\varepsilon = 0$	$\alpha \in [0, \infty)$	$-\frac{p-1}{p}C^{-\frac{1}{p-1}}\alpha^{\frac{p}{p-1}}$
Piecewise polynomial	$\varepsilon = 0$	$\alpha \in [0, C]$	$-\frac{p-1}{p}\sigma C^{-\frac{1}{p-1}}\alpha^{\frac{p}{p-1}}$

In the second case ($\xi \geq \sigma$) we have

$$T(\xi) = \xi - \sigma\frac{p-1}{p} - \xi = -\sigma\frac{p-1}{p}, \tag{9.29}$$

and $\xi = \inf\{\xi \,|\, C \geq \alpha\} = \sigma$; hence $\alpha \in [0, C]$. These two cases can be combined to yield

$$\alpha \in [0, C] \text{ and } T(\alpha) = -\frac{p-1}{p}\sigma C^{-\frac{p}{p-1}}\alpha^{\frac{p}{p-1}}. \tag{9.30}$$

Table 9.1 contains a summary of the various conditions on α, and formulas for $T(\alpha)$ for different loss functions.[3] Note that the maximum slope of $\tilde{c}$ determines the region of feasibility of α, meaning that $s := \sup_{\xi \in \mathbb{R}^+} \partial_\xi \tilde{c}(\xi) < \infty$ leads to compact intervals $[0, Cs]$ for α. This means that the influence of a single pattern is bounded, leading to robust estimators (cf. Chapter 3 and Proposition 9.4 below). We also observe experimentally that the performance of an SVM depends on the loss function used [376, 515, 95].

A cautionary remark is necessary regarding the use of loss functions other than the ε-insensitive loss. Unless $\varepsilon \neq 0$, we lose the advantage of a sparse decomposition. This may be acceptable in the case of few data, but will render the prediction step rather slow otherwise. Hence we have to trade off a potential loss in prediction accuracy with faster predictions. Note, however, that this issue could be addressed using reduced set algorithms like those described in Chapter 18, or sparse decomposition techniques [513]. In a Bayesian setting, Tipping [539] recently showed how the squared loss function can be used without sacrificing sparsity, cf. Section 16.6.

3. The table displays $CT(\alpha)$ instead of $T(\alpha)$, since the former can be plugged directly into the corresponding optimization equations.

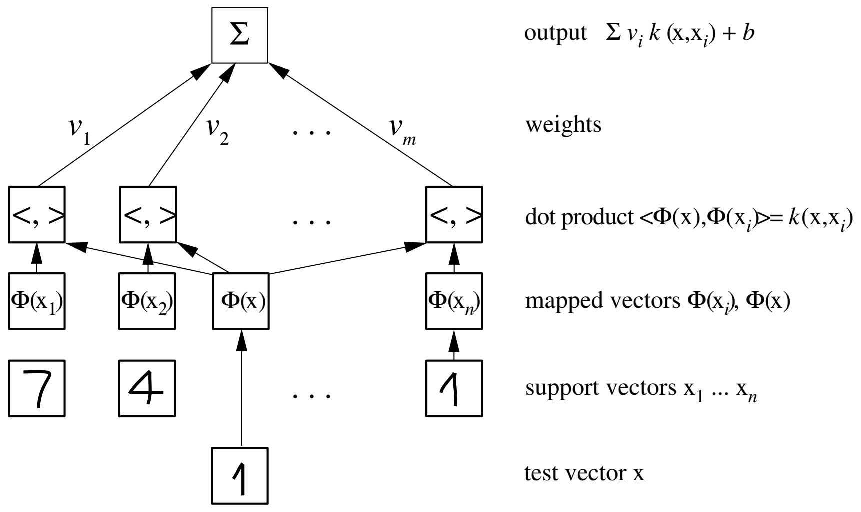

output $\Sigma\, v_i\, k\,(x,x_i) + b$
weights
dot product $<\Phi(x),\Phi(x_i)> = k(x,x_i)$
mapped vectors $\Phi(x_i)$, $\Phi(x)$
support vectors $x_1 \ldots x_n$
test vector x
Σ
v_1
v_2
v_m
$<,>$
$\Phi(x_1)$
$\Phi(x_2)$
$\Phi(x)$
$\Phi(x_n)$

Figure 9.2 Architecture of a regression machine constructed using the SV algorithm. In typical regression applications, the inputs would not be visual patterns. Nevertheless, in this example, the inputs are depicted as handwritten digits.

9.2.3 The Bigger Picture

Let us briefly review the basic properties of the SV algorithm for regression, as described so far. Figure 9.2 contains a graphical overview of the different steps in the regression stage. The input pattern (for which a prediction is to be made) is mapped into feature space by a map Φ. Then dot products are computed with the images of the training patterns under the map Φ. This corresponds to evaluating kernel functions $k(x_i, x)$. Finally, the dot products are added up using the weights $\nu_i = \alpha_i^* - \alpha_i$. This, plus the constant term b, yields the final prediction output. The process described here is very similar to regression in a neural network, with the difference that in the SV case, the weights in the input layer are a subset of the training patterns.

The toy example in Figure 9.3 demonstrates how the SV algorithm chooses the flattest function among those approximating the original data with a given precision. Although requiring flatness only in *feature* space, we observe that the functions are also *smooth* in *input* space. This is due to the fact that kernels can be associated with smoothness properties via regularization operators, as explained in more detail in Chapter 4.

Finally, Figure 9.4 shows the relation between approximation quality and sparsity of representation in the SV case. The lower the precision required for approximating the original data, the fewer SVs are needed to encode this data. The non-SVs are redundant — even without these patterns in the training set, the SVM

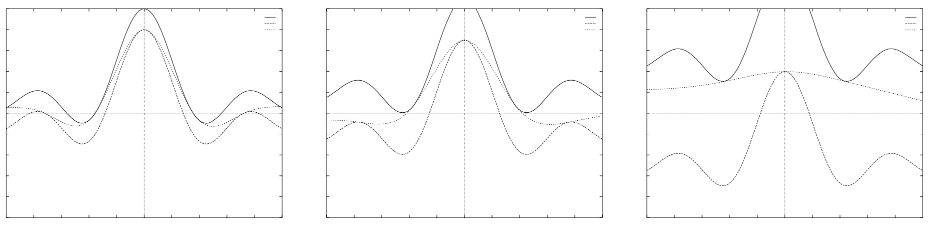


Figure 9.3 From top to bottom: approximation of the function $\operatorname{sinc} x$ with precisions $\varepsilon = 0.1, 0.2$, and 0.5. The solid top and dashed bottom lines indicate the size of the ε-tube, here drawn around the target function $\operatorname{sinc} x$. The dotted line between them is the regression function.

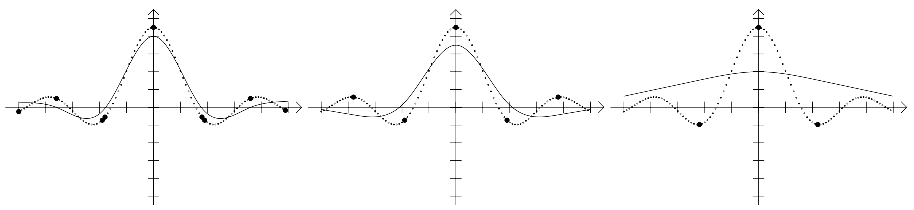


Figure 9.4 Left to right: regression (solid line), data points (small dots) and SVs (big dots) for an approximation of $\operatorname{sinc} x$ (dotted line) with $\varepsilon = 0.1, 0.2$, and 0.5. Note the decrease in the number of SVs.

would have constructed exactly the same function f. We might be tempted to use this property as an efficient means of data compression, namely by storing only the support patterns, from which the estimate can be reconstructed completely. Unfortunately, this approach turns out not to work well in the case of noisy high-dimensional data, since for moderate approximation quality, the number of SVs can be rather high [572].

9.3 ν-SV Regression

The parameter ε of the ε-insensitive loss is useful if the desired accuracy of the approximation can be specified beforehand. In some cases, however, we just want the estimate to be as accurate as possible, without having to commit ourselves to a specific level of accuracy a priori. We now describe a modification of the ε-SVR algorithm, called ν-SVR, which automatically computes ε [481].

To estimate functions (9.2) from empirical data (9.3) we proceed as follows. At each point $\mathbf{x}_i$, we allow an error ε. Everything above ε is captured in slack variables $\xi_i^{(*)}$, which are penalized in the objective function via a regularization constant C, chosen a priori. The size of ε is traded off against model complexity and slack variables via a constant $\nu \geq 0$:

$$\underset{\mathbf{w}\in\mathcal{H},\boldsymbol{\xi}^{(*)}\in\mathbb{R}^m,\varepsilon,b\in\mathbb{R}}{\text{minimize}} \quad \tau(\mathbf{w},\boldsymbol{\xi}^{(*)},\varepsilon) = \frac{1}{2}\|\mathbf{w}\|^2 + C\cdot\left(\nu\varepsilon + \frac{1}{m}\sum_{i=1}^{m}(\xi_i+\xi_i^*)\right), \tag{9.31}$$

$$\text{subject to } (\langle \mathbf{w},\mathbf{x}_i\rangle + b) - y_i \leq \varepsilon + \xi_i, \tag{9.32}$$

$$y_i - (\langle \mathbf{w},\mathbf{x}_i\rangle + b) \leq \varepsilon + \xi_i^*, \tag{9.33}$$

$$\xi_i^{(*)} \geq 0, \quad \varepsilon \geq 0. \tag{9.34}$$

Primal Problem ν-SVR

For the constraints, we introduce multipliers $\alpha_i^{(*)}, \eta_i^{(*)}, \beta \geq 0$, and obtain the Lagrangian,

$$L(\mathbf{w},b,\boldsymbol{\alpha}^{(*)},\beta,\boldsymbol{\xi}^{(*)},\varepsilon,\boldsymbol{\eta}^{(*)}) = \tag{9.35}$$

$$\frac{1}{2}\|\mathbf{w}\|^2 + C\nu\varepsilon + \frac{C}{m}\sum_{i=1}^{m}(\xi_i+\xi_i^*) - \beta\varepsilon - \sum_{i=1}^{m}(\eta_i\xi_i + \eta_i^*\xi_i^*)$$

$$-\sum_{i=1}^{m}\alpha_i(\xi_i + y_i - \langle \mathbf{w},\mathbf{x}_i\rangle - b + \varepsilon) - \sum_{i=1}^{m}\alpha_i^*(\xi_i^* + \langle \mathbf{w},\mathbf{x}_i\rangle + b - y_i + \varepsilon).$$

To minimize (9.31), we have to find the saddle point of L, meaning that we minimize over the primal variables $\mathbf{w},\varepsilon,b,\xi_i^{(*)}$ and maximize over the dual variables $\alpha_i^{(*)},\beta,\eta_i^{(*)}$. Setting the derivatives with respect to the primal variables equal to zero yields the four equations

$$\mathbf{w} = \sum_i(\alpha_i^* - \alpha_i)\mathbf{x}_i, \tag{9.36}$$

$$C\cdot\nu - \sum_i(\alpha_i + \alpha_i^*) - \beta = 0, \tag{9.37}$$

$$\sum_{i=1}^{m}(\alpha_i - \alpha_i^*) = 0, \tag{9.38}$$

$$\tfrac{C}{m} - \alpha_i^{(*)} - \eta_i^{(*)} = 0. \tag{9.39}$$

As in Section 9.2, the $\alpha_i^{(*)}$ are nonzero in the *SV expansion* (9.36) only when a constraint (9.32) or (9.33) is precisely met.

Substituting the above four conditions into L leads to the dual optimization problem (sometimes called the Wolfe dual). We will state it in the kernelized form: as usual, we substitute a kernel k for the dot product, corresponding to a dot product in some feature space related to input space via a nonlinear map Φ,

$$k(x,x') = \langle \Phi(x),\Phi(x')\rangle = \langle \mathbf{x},\mathbf{x}'\rangle\,. \tag{9.40}$$

ν-SVR Dual Program

Rewriting the constraints, and noting that $\beta,\eta_i^{(*)} \geq 0$ do not appear in the dual, we arrive at the ν-SVR Optimization Problem: for $\nu \geq 0, C > 0$,

$$\underset{\boldsymbol{\alpha}^{(*)}\in\mathbb{R}^m}{\text{maximize}}\, W(\boldsymbol{\alpha}^{(*)}) = \sum_{i=1}^{m}(\alpha_i^* - \alpha_i)y_i - \frac{1}{2}\sum_{i,j=1}^{m}(\alpha_i^* - \alpha_i)(\alpha_j^* - \alpha_j)k(x_i,x_j), \tag{9.41}$$

$$\text{subject to } \sum_{i=1}^{m}(\alpha_i - \alpha_i^*) = 0, \tag{9.42}$$

$$\alpha_i^{(*)} \in \left[0, \tfrac{C}{m}\right], \tag{9.43}$$

$$\sum_{i=1}^{m}(\alpha_i + \alpha_i^*) \leq C \cdot \nu. \tag{9.44}$$

The regression estimate then takes the form (cf. (9.2), (9.36), (9.40))

$$f(x) = \sum_{i=1}^{m}(\alpha_i^* - \alpha_i)k(x_i, x) + b, \tag{9.45}$$

where b (and ε) can be computed by taking into account that (9.32) and (9.33) become equalities with $\xi_i^{(*)} = 0$ for points with $0 < \alpha_i^{(*)} < C/m$, due to the KKT conditions. Here, substitution of $\sum_j(\alpha_j^* - \alpha_j)k(x_j, x)$ for $\langle \mathbf{w}, \mathbf{x} \rangle$ is understood, cf. (9.36), (9.40). Geometrically, this amounts to saying that we can compute the thickness and vertical position of the tube by considering some points that sit exactly on the edge of the tube.[4]

We now show that ν has an interpretation similar to the case of ν-SV pattern recognition (Section 7.5). This is not completely obvious: recall that in the case of pattern recognition, we introduced ν to replace C. In regression, on the other hand, we introduced it to replace ε.

Before we give the result, the following observation concerning ε is helpful. If $\nu > 1$, then necessarily $\varepsilon = 0$, since it does not pay to increase ε. This can be seen either from (9.31) — the slacks are "cheaper" — or by noting that for $\nu \geq 1$, (9.43) implies (9.44), since $\alpha_i \alpha_i^* = 0$ for all i (9.58). Therefore, (9.44) is redundant, and all values $\nu \geq 1$ are actually equivalent. Hence, we restrict ourselves to $0 \leq \nu \leq 1$.

If $\nu < 1$, we mostly find $\varepsilon > 0$. It is still possible that $\varepsilon = 0$, for instance if the data are noise-free and can be perfectly interpolated with a low capacity model. The case $\varepsilon = 0$ is not what we are interested in: it corresponds to plain L_1-loss regression.

Below, we will use the term *errors* to refer to training points lying outside the tube, and the term *fraction* of errors/SVs to denote the relative numbers of errors/SVs; that is, these respective quantities are divided by m. In this proposition, we define the *modulus of absolute continuity* of a function f as the function $\epsilon(\delta) := \sup \sum_i |f(b_i) - f(a_i)|$, where the supremum is taken over all disjoint intervals (a_i, b_i) with $a_i < b_i$ satisfying $\sum_i(b_i - a_i) < \delta$. Loosely speaking, the condition on the conditional density of y given x asks that it be absolutely continuous 'on average.'

ν-Property

Proposition 9.2 *Suppose ν-SVR is applied to some data set and the resulting ε is nonzero. The following statements hold:*

4. Should it occur, for instance due to numerical problems, that it is impossible to find two non-bound SVs at the two edges of the tube, then we can replace them by the SVs which are closest to the tube. The SV closest to the top of the tube can be found by minimizing $y_i - \langle \mathbf{w}, \mathbf{x}_i \rangle$ over all points with $\alpha_i^* > 0$; similarly, for the bottom SVs we minimize $\langle \mathbf{w}, \mathbf{x}_i \rangle - y_i$ over the points with $\alpha_i > 0$. We then proceed as we would with the non-bound SVs, cf. Problem 9.16.

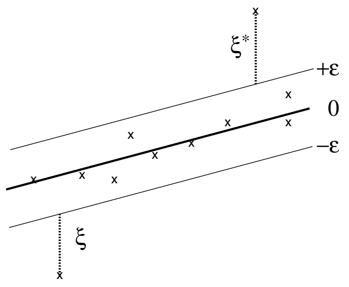


Figure 9.5 Graphical depiction of the ν-trick. Imagine increasing ε, starting from 0. The first term in $\nu\varepsilon + \frac{1}{m}\sum_{i=1}^{m}(\xi_i + \xi_i^*)$ (cf. (9.31)) increases proportionally to ν, while the second term decreases proportionally to the fraction of points outside of the tube. Hence, ε grows as long as the latter fraction is larger than ν. At the optimum, it must therefore be $\leq \nu$ (Proposition 9.2, (i)). Next, imagine decreasing ε, starting from some large value. Again, the change in the first term is proportional to ν, but this time, the change in the second term is proportional to the fraction of SVs (even points *on* the edge of the tube contribute). Hence, ε shrinks as long as the fraction of SVs is smaller than ν, leading eventually to Proposition 9.2, (ii).

(i) ν is an upper bound on the fraction of errors.

(ii) ν is a lower bound on the fraction of SVs.

(iii) Suppose the data (9.3) were generated iid from a distribution $P(x, y) = P(x)P(y|x)$, with $P(y|x)$ continuous and the expectation of the modulus of absolute continuity of its density satisfying $\lim_{\delta \to 0} \mathbf{E}_x[\epsilon(\delta)] = 0$. With probability 1, asymptotically, ν equals both the fraction of SVs and the fraction of errors.

The proposition shows that $0 \leq \nu \leq 1$ can be used to control the number of errors. Since the constraint (9.42) implies that (9.44) is equivalent to $\sum_i \alpha_i^{(*)} \leq C\nu/2$, we conclude that Proposition 9.2 actually holds separately for the upper and the lower edge of the tube, with $\nu/2$ each. As an aside, note that by the same argument, the number of SVs at the two edges of the standard ε-SVR tube asymptotically agree.

The proof of Proposition 9.2 can be found in Section A.2. In its stead, we use a graphical argument that should make the result plausible (Figure 9.5). For further information on the ν-trick in a more general setting, cf. Section 3.4.3.

Connection ν-SVR / ε-SVR

Let us briefly discuss how ν-SVR relates to ε-SVR (Section 9.1). Both algorithms use the ε-insensitive loss function, but ν-SVR *automatically* computes ε. From a Bayesian viewpoint, this automatic adaptation of the loss function can be interpreted as adapting the error model, controlled by the hyperparameter ν (cf. Chapter 3). Comparing (9.17) (substitution of a kernel for the dot product is understood) and (9.41), we note that ε-SVR requires an additional term $-\varepsilon\sum_{i=1}^{m}(\alpha_i^* + \alpha_i)$, which, for fixed $\varepsilon > 0$, encourages some of the $\alpha_i^{(*)}$ to be 0. Accordingly, the constraint (9.44), which appears in ν-SVR, is not needed. The primal problems (9.8) and (9.31) differ in the term $\nu\varepsilon$. If $\nu = 0$, then the optimization can grow ε arbitrarily large, hence zero empirical risk can be obtained even when all α are zero.

In the following sense, ν-SVR includes ε-SVR. Note that in the general case, using kernels, $\bar{\mathbf{w}}$ is a vector in feature space.

Proposition 9.3 *If ν-SVR leads to the solution $\bar{\varepsilon}, \bar{\mathbf{w}}, \bar{b}$, then ε-SVR with ε set a priori to $\bar{\varepsilon}$, and the same value of C, has the solution $\bar{\mathbf{w}}, \bar{b}$.*

Proof If we minimize (9.31), then fix ε and minimize only over the remaining variables, the solution does not change. ■

Connection to Robust Estimators

Using the ε-insensitive loss function, only the patterns outside the ε-tube enter the empirical risk term, whereas the patterns closest to the actual regression have zero loss. This does not mean that it is only the 'outliers' that determine the regression. In fact, the contrary is the case:

Proposition 9.4 (Resistance of SV Regression) *Using Support Vector Regression with the ε-insensitive loss function (9.1), local movements of target values of points outside the tube do not influence the regression.*

Proof Shifting y_i locally does not change the status of (x_i, y_i) as being a point outside the tube. The dual solution $\boldsymbol{\alpha}^{(*)}$ then remains feasible; which is to say it satisfies the constraints (the point still has $\alpha_i^{(*)} = C/m$). In addition, the primal solution, with ξ_i transformed according to the movement of x_i, is also feasible. Finally, the KKT conditions are still satisfied, as $\alpha_i^{(*)} = C/m$. Thus (Chapter 6), $\boldsymbol{\alpha}^{(*)}$ remains the solution. ■

The proof relies on the fact that everywhere outside the tube, the upper bound on the $\alpha_i^{(*)}$ is the same. This, in turn, is precisely the case if the loss function increases linearly outside the ε-tube (cf. Chapter 3 for requirements for robust loss functions). Inside, a range of functions is permissible, provided their first derivative is smaller than that of the linear part.

In the case of ν-SVR with ε-insensitive loss, the above proposition implies that essentially, the regression is a generalization of an estimator for the mean of a random variable which

(a) throws away the largest and smallest examples (a fraction $\nu/2$ of either category — in Section 9.3, it is shown that the sum constraint (9.42) implies that Proposition 9.2 can be applied separately for the two sides, using $\nu/2$); and

(b) estimates the mean by taking the average of the two extremal ones of the remaining examples.

Trimmed Mean

This resistance to outliers is close in spirit to robust estimators like the *trimmed mean*. In fact, we could get closer to the idea of the trimmed mean, which first throws away the largest and smallest points and then computes the mean of the remaining points, by using a quadratic loss inside the ε-tube. This would leave us with Huber's robust loss function (see Table 3.1).

Note, moreover, that the parameter ν is related to the breakdown point of the corresponding robust estimator [251]. As it specifies the fraction of points which may be arbitrarily bad outliers, ν is related to the fraction of some arbitrary distribution that may be added to a known noise model without leading to a failure of the estimator.

Finally, we add that by a simple modification of the loss function (cf. [594]), namely weighting the slack variables $\boldsymbol{\xi}^{(*)}$ above and below the tube in the target

Table 9.2 Asymptotic behavior of the fraction of errors and SVs.
The ε found by ν-SV regression is largely independent of the sample size m. The fraction of SVs and the fraction of errors approach $\nu = 0.2$ from above and below, respectively, as the number of training examples m increases (cf. Proposition 9.2).

m	10	50	100	200	500	1000	1500	2000
ε	0.27	0.22	0.23	0.25	0.26	0.26	0.26	0.26
fraction of errors	0.00	0.10	0.14	0.18	0.19	0.20	0.20	0.20
fraction of SVs	0.40	0.28	0.24	0.23	0.21	0.21	0.20	0.20

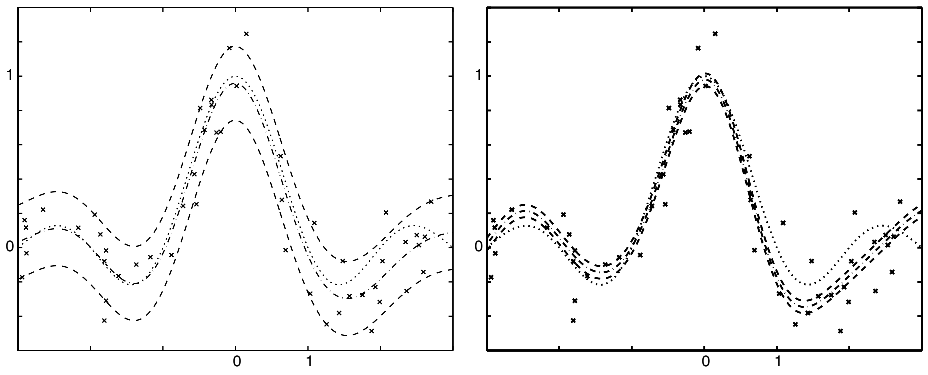


Figure 9.6 ν-SV regression with $\nu = 0.2$ (left) and $\nu = 0.8$ (right). The larger ν allows more points to lie outside the tube (see Section 9.3). The algorithm automatically adjusts ε to 0.22 (left) and 0.04 (right). Shown are the sinc function (dotted), the regression f and the tube $f \pm \varepsilon$.

function (9.31) by 2λ and $2(1-\lambda)$ respectively, with $\lambda \in [0,1]$, we can estimate generalized *quantiles*. The argument proceeds as follows. Asymptotically, all patterns have multipliers at bound (cf. Proposition 9.2). The parameter λ, however, changes the upper bounds in the box constraints applying to the two different types of slack variables to $2C\lambda/m$ and $2C(1-\lambda)/m$, respectively. The equality constraint (9.38) then implies that $(1-\lambda)$ and λ give the fractions of points (of those which are outside the tube) which lie on the top and bottom of the tube, respectively.

Quantile

Experiments

Let us now look at some experiments. We start with a toy example, which involves estimating the regression of a noisy sinc function, given m examples (x_i, y_i), with x_i drawn uniformly from $[-3,3]$, and $y_i = \sin(\pi x_i)/(\pi x_i) + \upsilon_i$. The υ_i were drawn from a Gaussian with zero mean and variance σ^2, and we used the RBF kernel $k(x,x') = \exp(-|x-x'|^2)$, $m = 50, C = 100, \nu = 0.2$, and $\sigma = 0.2$. Standard deviation error bars were computed from 100 trials. Finally, the *risk* (or test error) of a regression estimate f was computed with respect to the sinc function without noise, as $\frac{1}{6}\int_{-3}^{3} |f(x) - \sin(\pi x)/(\pi x)|\, dx$. Results are given in Table 9.2 and Figures 9.6–9.12.

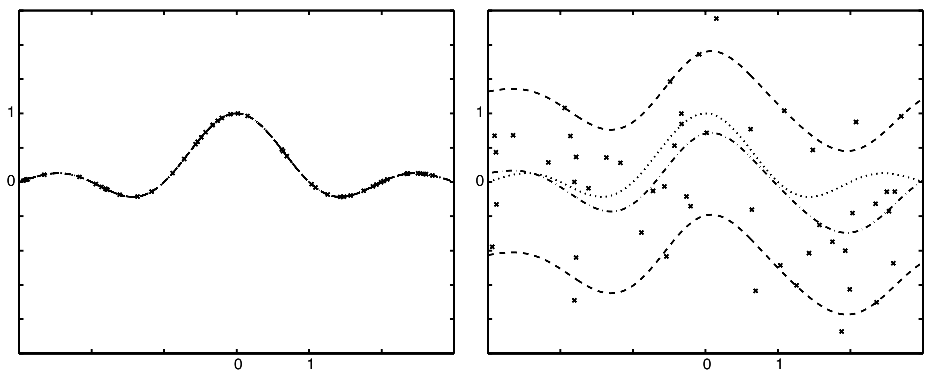


Figure 9.7 ν-SV regression on data with noise $\sigma = 0$ (left) and $\sigma = 1$ (right). In both cases, $\nu = 0.2$. The tube width automatically adjusts to the noise (top: $\varepsilon = 0$, bottom: $\varepsilon = 1.19$).

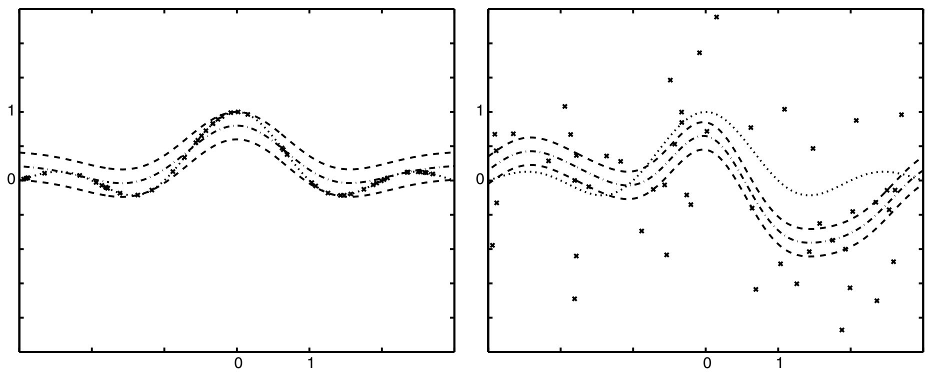


Figure 9.8 ε-SV regression (Section 9.2) on data with noise $\sigma = 0$ (left) and $\sigma = 1$ (right). In both cases, $\varepsilon = 0.2$ — this choice, which has to be specified a priori, is ideal for neither case: in the upper figure, the regression estimate is biased; in the lower figure, ε does not match the external noise [510].

9.4 Convex Combinations and ℓ_1-Norms

All the algorithms presented so far involve convex, and at best, quadratic programming. Yet we might think of reducing the problem to a case where linear programming techniques can be applied. This can be done in a straightforward fashion [591, 517] for both SV pattern recognition and regression. The key is to replace the original objective function by

$$R_{\text{reg}}[f] := \frac{1}{m}\|\boldsymbol{\alpha}\|_1 + C \cdot R_{\text{emp}}[f] \tag{9.46}$$

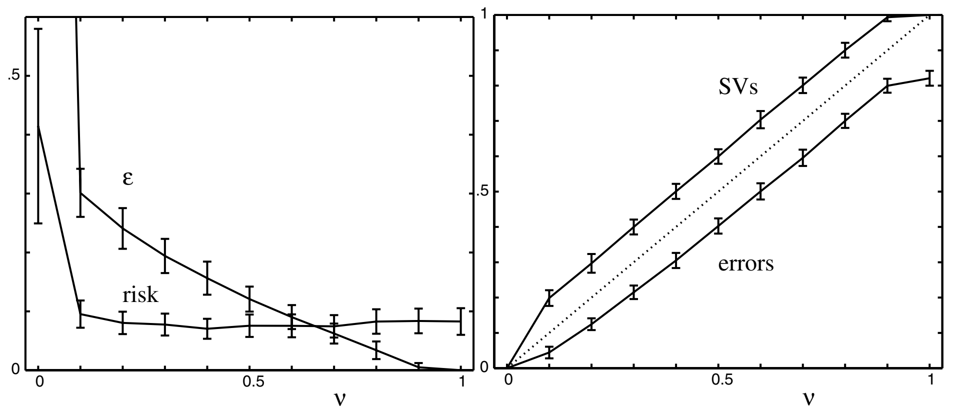


Figure 9.9 ν-SVR for different values of the error constant ν. Notice how ε decreases when more errors are allowed (large ν), and that over a large range of ν, the test error (risk) is insensitive to changes in ν.

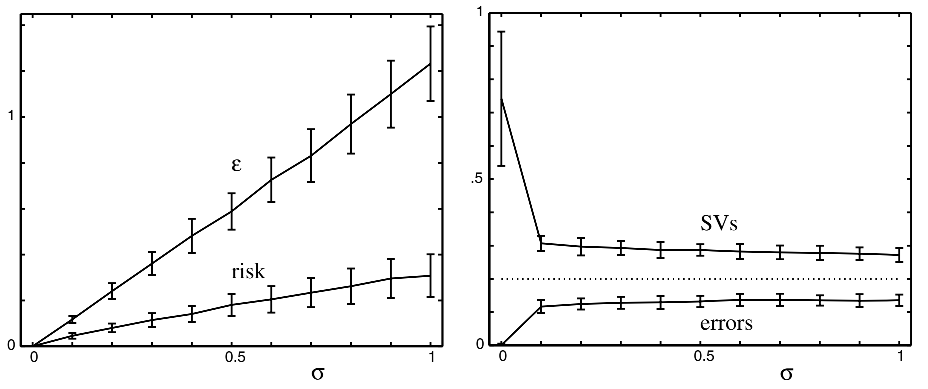


Figure 9.10 ν-SVR for different values of the noise σ. The tube radius ε increases linearly with σ (largely due to the fact that both ε and the $\xi_i^{(*)}$ enter the loss function linearly). Due to the automatic adaptation of ε, the number of SVs and of points outside the tube (errors) is largely independent of σ, except for the noise-free case $\sigma = 0$.

where $\|\boldsymbol{\alpha}\|_1 = \sum_{i=1}^m |\alpha_i|$ denotes the ℓ_1 norm in coefficient space. Using the SV kernel expansion (9.18),

$$f(x) = \sum_{i=1}^{m} \alpha_i k(x_i, x) + b, \tag{9.47}$$

this translates to the objective function

$$R_{\text{reg}}[f] = \frac{1}{m}\sum_{i=1}^{m} |\alpha_i| + \frac{C}{m}\sum_{i=1}^{m} c(x_i, y_i, f(x_i)). \tag{9.48}$$

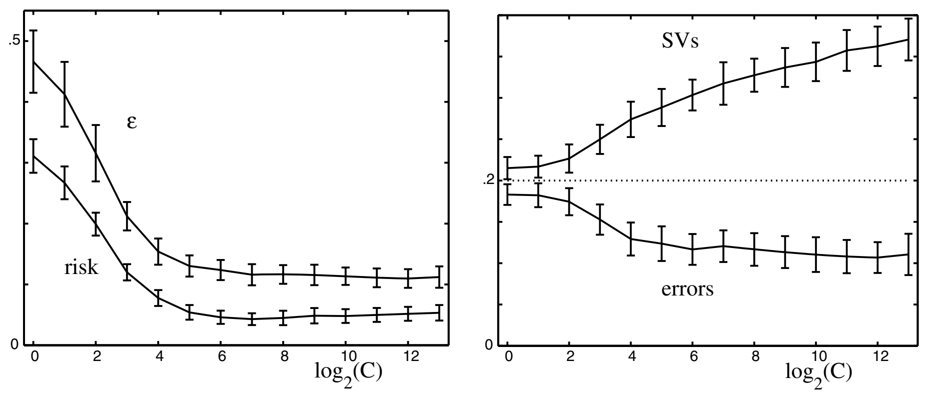


Figure 9.11 ν-SVR for different values of the constant C. The left graph shows that ε decreases when the regularization is decreased (large C). Only very little, if any, overfitting occurs. In the right graph, note that ν upper bounds the fraction of errors, and lower bounds the fraction of SVs (cf. Proposition 9.2). The bound gets looser as C increases — this corresponds to a smaller number of examples m relative to C (cf. Table 9.2).

ε-LP Regression Objective Function

For the ε-insensitive loss function, this leads to a linear programming problem. For other loss functions, the problem remains a quadratic or general convex one. Therefore we limit ourselves to the derivation of the linear programming problem in the case of the $|\cdot|_\varepsilon$ loss function. Reformulating (9.48) yields

$$\begin{array}{ll} \underset{\boldsymbol{\alpha}^{(*)},\boldsymbol{\xi}^{(*)}\in\mathbb{R}^m,b\in\mathbb{R}}{\text{minimize}} & \frac{1}{m}\sum_{i=1}^{m}(\alpha_i+\alpha_i^*)+\frac{C}{m}\sum_{i=1}^{m}(\xi_i+\xi_i^*), \\ \text{subject to} & \left\{\begin{array}{lll} \sum_{j=1}^{m}(\alpha_j-\alpha_j^*)k(x_j,x_i)+b-y_i & \leq & \varepsilon+\xi_i, \\ y_i-\sum_{j=1}^{m}(\alpha_j-\alpha_j^*)k(x_j,x_i)-b & \leq & \varepsilon+\xi_i^*, \\ \alpha_i,\alpha_i^*,\xi_i,\xi_i^* & \geq & 0. \end{array}\right. \end{array} \tag{9.49}$$

Unlike the SV case, the transformation into its dual does not give any improvement in the structure of the optimization problem. Hence it is best to minimize $R_{\text{reg}}[f]$ directly, which can be achieved using a linear optimizer (see [130, 336, 555]).

Weston et al. [591] use a similar LP approach to estimate densities on a line. We may even obtain bounds on the generalization error [505] which exhibit better rates than in the SV case [606]; cf. Chapter 12.

ν-LP Regression

We conclude this section by noting that we can combine these ideas with those presented in the previous section, and construct a ν-LP regression algorithm [517]. It differs from the previous ν-SV algorithm in that we now minimize

$$R_{\text{reg}}+C\nu\varepsilon=\frac{1}{m}\sum_{i=1}^{m}|\alpha_i|+CR_{\text{emp}}^{\varepsilon}[f]+C\nu\varepsilon. \tag{9.50}$$

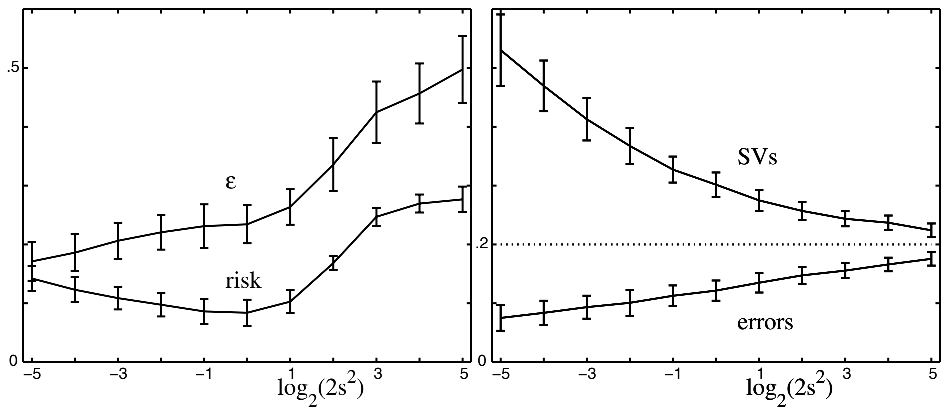


Figure 9.12 ν-SVR for different values of the Gaussian kernel width $2s^2$, using $k(x, x') = \exp(-|x - x'|^2/(2s^2))$. Using a kernel that is too wide results in underfitting; moreover, since the tube becomes too rigid as $2s^2$ gets larger than 1, the ε which is needed to accomodate a fraction $(1 - \nu)$ of points increases significantly. In the plot on the right, it can again be seen that the speed of the uniform convergence responsible for the asymptotic statement given in Proposition 9.2 depends on the capacity of the underlying model. Increasing the kernel width leads to smaller covering numbers (Chapter 12) and therefore faster convergence.

The goal here is not only to achieve a small training error (with respect to ε), but also to obtain a solution with a small ε. Rewriting (9.50) as a linear program yields

$$\begin{aligned}
&\underset{\boldsymbol{\alpha}^{(*)},\boldsymbol{\xi}^{(*)}\in\mathbb{R}^m, b,\varepsilon\in\mathbb{R}}{\text{minimize}} \quad \frac{1}{m}\sum_{i=1}^{m}(\alpha_i + \alpha_i^*) + \frac{C}{m}\sum_{i=1}^{m}(\xi_i + \xi_i^*) + C\nu\varepsilon, \\
&\text{subject to} \quad \begin{cases} \sum_{j=1}^{m}(\alpha_j - \alpha_j^*)k(x_j, x_i) + b - y_i & \leq \varepsilon + \xi_i, \\ y_i - \sum_{j=1}^{m}(\alpha_j - \alpha_j^*)k(x_j, x_i) - b & \leq \varepsilon + \xi_i^*, \\ \alpha_i, \alpha_i^*, \xi_i, \xi_i^*, \varepsilon & \geq 0. \end{cases}
\end{aligned} \tag{9.51}$$

The difference between (9.50) and (9.49) lies in the objective function, and the fact that ε has now become a variable of the optimization problem.

The ν-property (Proposition 9.2) also holds for ν-LP regression. The proof is analogous to the ν-SV case, and can be found in [517].

9.5 Parametric Insensitivity Models

In Section 9.3, we generalized ε-SVR by estimating the width of the tube rather than taking it as given a priori. What we retained, however, is the assumption that the ε-insensitive zone has a tube shape. We now go one step further and use parametric models of arbitrary shape [469]. This can be useful in situations where the noise depends on x (this is called heteroscedastic noise).

Let $\{\zeta_q^{(*)}\}$ (here and below, $q = 1, \ldots, p$ is understood) be a set of $2p$ positive functions on the input space $\mathcal{X}$. Consider the following quadratic program: for given $\nu_1^{(*)}, \ldots, \nu_p^{(*)} \geq 0$,

$$\underset{\mathbf{w}\in\mathcal{H},\boldsymbol{\xi}^{(*)}\in\mathbb{R}^m,\boldsymbol{\varepsilon}^{(*)}\in\mathbb{R}^p,b\in\mathbb{R}}{\text{minimize}} \quad \tau(\mathbf{w}, \boldsymbol{\xi}^{(*)}, \boldsymbol{\varepsilon}^{(*)}) = \|\mathbf{w}\|^2/2 + C \cdot \left(\sum_{q=1}^{p}(\nu_q\varepsilon_q + \nu_q^*\varepsilon_q^*) + \frac{1}{m}\sum_{i=1}^{m}(\xi_i + \xi_i^*)\right), \tag{9.52}$$

$$\text{subject to} \quad (\langle \mathbf{w}, \Phi(x_i)\rangle + b) - y_i \leq \sum_{q=1}^{p} \varepsilon_q \zeta_q(x_i) + \xi_i, \tag{9.53}$$

$$y_i - (\langle \mathbf{w}, \Phi(x_i)\rangle + b) \leq \sum_{q=1}^{p} \varepsilon_q^* \zeta_q^*(x_i) + \xi_i^*, \tag{9.54}$$

$$\xi_i^{(*)} \geq 0, \;\; \varepsilon_q^{(*)} \geq 0. \tag{9.55}$$

A calculation analogous to that in Section 9.3 shows that the Wolfe dual consists of maximizing (9.41) subject to (9.42), (9.43), and, instead of (9.44), the modified constraints,

$$\sum_{i=1}^{m} \alpha_i^{(*)} \zeta_q^{(*)}(x_i) \leq C \cdot \nu_q^{(*)}. \tag{9.56}$$

which are still linear in $\boldsymbol{\alpha}^{(*)}$. In the toy experiment shown in Figure 9.13, we use a simplified version of this optimization problem, where we drop the term $\nu_q^*\varepsilon_q^*$ from the objective function (9.52), and use ε_q and ζ_q in (9.54). By this, we render the problem symmetric with respect to the two edges of the tube. In addition, we use $p = 1$. This leads to the same Wolfe dual, except for the last constraint, which becomes (cf. (9.44))

$$\sum_{i=1}^{m} (\alpha_i + \alpha_i^*)\zeta(x_i) \leq C \cdot \nu. \tag{9.57}$$

Note that the optimization problem of Section 9.3 can be recovered using the constant function $\zeta \equiv 1$.[5]

The advantage of this setting is that since the same ν is used for both sides of the tube, the computation of ε and b is straightforward: for instance, by solving a linear system, using two conditions such as those described following (9.45). Otherwise, general statements become cumbersome: the linear system can have a zero determinant, depending on whether the functions $\zeta_p^{(*)}$, evaluated on the x_i with $0 < \alpha_i^{(*)} < C/m$, are linearly dependent. The latter occurs, for instance, if we use constant functions $\zeta^{(*)} \equiv 1$. In this case, it is pointless to use two different

5. Observe the similarity to semiparametric SV models (Section 4.8) where a modification of the expansion of f leads to similar additional constraints. The important difference in the present setting is that the Lagrange multipliers α_i and α_i^* are treated equally, and not with different signs as in semiparametric modelling.

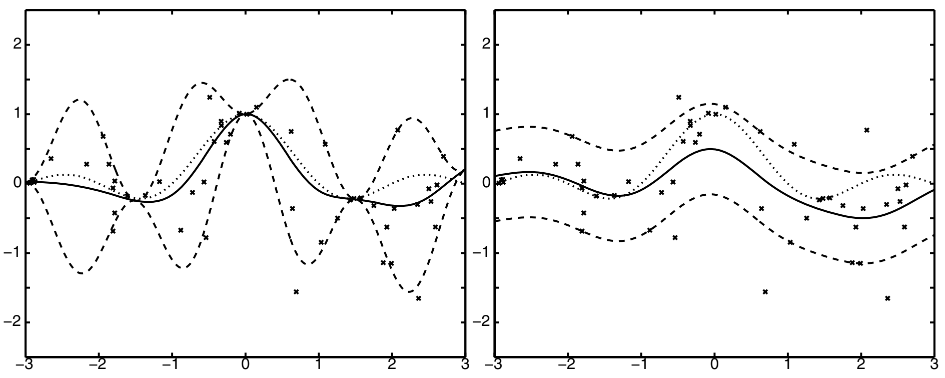


Figure 9.13 Toy example, using prior knowledge about an x-dependence of the noise. Additive noise ($\sigma = 1$) was multiplied by the function $\sin^2((2\pi/3)x)$. *Left:* the *same* function was used as ζ in the context of a parametric insensitivity tube (Section 9.5). *Right:* ν-SVR with standard tube.

values ν, ν^*, since the constraint (9.42) then implies that *both* sums $\sum_{i=1}^m \alpha_i^{(*)}$ are bounded by $C \cdot \min\{\nu, \nu^*\}$. We conclude this section by giving, without proof, a generalization of Proposition 9.2 to the optimization problem with constraint (9.57):

Proposition 9.5 *Suppose we run the above algorithm on a data set with the result that* $\varepsilon > 0$. *Then*

(i) $\frac{\nu m}{\sum_i \zeta(x_i)}$ *is an upper bound on the fraction of errors.*

(ii) $\frac{\nu m}{\sum_i \zeta(x_i)}$ *is an upper bound on the fraction of SVs.*

(iii) Suppose the data (9.3) were generated iid from a distribution $P(x, y) = P(x)P(y|x)$, *with* $P(y|x)$ *continuous and the expectation of its modulus of continuity satisfying* $\lim_{\delta \to 0} \mathbf{E}\epsilon(\delta) = 0$. *With probability 1, asymptotically, the fractions of SVs and errors equal* $\nu \cdot (\int \zeta(x)\, d\tilde{P}(x))^{-1}$, *where* $\tilde{P}$ *is the asymptotic distribution of SVs over* x.

Figure 9.13 gives an illustration of how we can make use of parametric insensitivity models. Using the proper model, the estimate gets much better. In the parametric case, we used $\nu = 0.1$ and $\zeta(x) = \sin^2((2\pi/3)x)$, which, due to $\int \zeta(x)\, dP(x) = 1/2$, corresponds to our standard choice $\nu = 0.2$ in ν-SVR (cf. Proposition 9.5). Although this relies on the assumption that the SVs are uniformly distributed, the experimental findings are consistent with the asymptotes predicted theoretically: for $m = 200$, we got 0.24 and 0.19 for the fraction of SVs and errors, respectively.

Table 9.3 Results for the Boston housing benchmark; *top:* ν-SVR, *bottom:* ε-SVR. Abbreviation key: MSE: Mean squared errors, STD: standard deviation thereof (100 trials), Errors: fraction of training points outside the tube, SVs: fraction of training points which are SVs.

ν	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
automatic ε	2.6	1.7	1.2	0.8	0.6	0.3	0.0	0.0	0.0	0.0
MSE	9.4	8.7	9.3	9.5	10.0	10.6	11.3	11.3	11.3	11.3
STD	6.4	6.8	7.6	7.9	8.4	9.0	9.6	9.5	9.5	9.5
Errors	0.0	0.1	0.2	0.2	0.3	0.4	0.5	0.5	0.5	0.5
SVs	0.3	0.4	0.6	0.7	0.8	0.9	1.0	1.0	1.0	1.0

ε	0	1	2	3	4	5	6	7	8	9	10
MSE	11.3	9.5	8.8	9.7	11.2	13.1	15.6	18.2	22.1	27.0	34.3
STD	9.5	7.7	6.8	6.2	6.3	6.0	6.1	6.2	6.6	7.3	8.4
Errors	0.5	0.2	0.1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
SVs	1.0	0.6	0.4	0.3	0.2	0.1	0.1	0.1	0.1	0.1	0.1

9.6 Applications

Boston Housing Benchmark

Empirical studies using ε-SVR have shown excellent performance on the widely used Boston housing regression benchmark set [529]. Due to Proposition 9.3, the only difference between ν-SVR and standard ε-SVR lies in the fact that different parameters, ε vs. ν, have to be specified a priori. We now describe how the results obtained on this benchmark set change with the adjustment of parameters ε and nu. In our experiments, we kept all remaining parameters fixed, with C and the width $2s^2$ in $k(x, x') = \exp(-\|x - x'\|^2/(2s^2))$ chosen as in [482]: we used $2s^2 = 0.3 \cdot N$, where $N = 13$ is the input dimension, and $C/m = 10 \cdot 50$ (the original value of 10 was corrected since in the present case, the maximal y-value is 50 rather than 1). We performed 100 runs, where each time the overall set of 506 examples was randomly split into a training set of $m = 481$ examples and a test set of 25 examples (cf. [529]). Table 9.3 shows that over a wide range of ν (recall that only $0 \leq \nu \leq 1$ makes sense), we obtained performances which are close to the *best* performances that can be achieved using a value of ε selected a priori by looking at the test set.[6] Finally, note that although we did not use validation techniques to select the optimal values for C and $2s^2$, the performances are state of the art: Stitson et al. [529] report an MSE of 7.6 for ε-SVR using ANOVA kernels (cf. (13.13) in Section 13.6), and 11.7 for Bagging regression trees. Table 9.3 also shows that in this real-world application, ν can be used to control the fractions of SVs and errors.

Time Series Prediction

Time series prediction is a field that often uses regression techniques. The stan-

6. For a theoretical analysis of how to select the asymptotically optimal ν for a given noise model, cf. Section 3.4.4.

dard method for writing a time series prediction problem in a regression estimation framework is to consider the time series as a dynamical system and to try to learn an attractor. For many time series $z(t)$, it is the case that if $N \in \mathbb{N}$ and $\tau > 0$ are chosen appropriately, then $z(t)$ can be predicted rather well from $(z(t-\tau), \ldots, z(t-N\tau)) \in \mathbb{R}^N$. We can thus consider a regression problem where the training set consists of the inputs $(z(t-\tau), \ldots, z(t-N\tau))$ and outputs $z(t)$, for a number of different values of t. Several characteristics of time series prediction make the problem hard for this naive regression approach. First, time series are often nonstationary — the regularity underlying the data changes over time. As a consequence, training examples that are generated as described above become less useful if they are taken from the distant past. Second, the different training examples are not iid, which is one of the assumptions on which the statistical learning model underlying SV regression is based.

Nevertheless, excellent results have been obtained using SVR in time series problems [376, 351]. In [376], a record result was reported for a widely studied benchmark dataset from the Santa Fe Institute. The study combined an ε-SVR with a method for segmenting the data, which stem from a time series that switches between different dynamics. SVR using ε-insensitive loss or Huber loss was found to significantly outperform all other results on that benchmark. Another benchmark record, on a different problem, has recently been achieved by [97]. To conclude, we note that SV regression has also successfully been applied in black-box system identification [216].

9.7 Summary

In this chapter, we showed how to generalize the SV algorithm to regression estimation. The generalization retains the sparsity of the solution through use of a SV expansion, exploits the kernel trick, and uses the same regularizer as its pattern recognition and single-class counterparts. We demonstrated how to derive the dual problems for a variety of loss functions, and we described variants of the algorithm. The LP-variant uses a different regularizer, which leads to sparse expansions in terms of patterns which no longer need to lie on the edge of the ε-tube; the ν-variant uses the same regularizer, but makes the loss function adaptive. The latter method has the advantage that the number of outliers and SVs can be controlled by a parameter of the algorithm, and, serendipitously, that the ε-parameter, which can be hard to set, is abolished.

Several interesting topics were omitted from this chapter, such as Density Estimation with SVMs [591, 563]. In this case, we make use of the fact that distribution functions are monotonically increasing, and that their values can be predicted with variable confidence which is adjusted by selecting different values of ε in the loss function. We also omitted the topic of Dictionaries, as introduced in the context of wavelets by [104] to allow a large class of basis functions to be considered simultaneously, for instance kernels with different widths. In the standard SV case, this can only be achieved by defining new kernels as linear combinations of differently scaled kernels. This is due to the fact that once a regularization operator is chosen, the solution minimizing the regularized risk function has to expanded into the corresponding Green's functions of P^*P (Chapter 4). In these cases, a possible way out is to resort to the LP version (Section 9.4). A final area of research left out of this chapter is the problem of estimating the values of functions at given test points, sometimes referred to as transduction [103].

9.8 Problems

9.1 (Product of SVR Lagrange Multipliers [561] •) *Show that for $\varepsilon > 0$, the solution of the SVR dual problem satisfies*

$$\alpha_i \alpha_i^* = 0 \tag{9.58}$$

for all $i = 1, \ldots, m$. Prove it either directly from (9.17), or from the KKT conditions.

Show that for $\varepsilon = 0$, we can always find a solution which satisfies (9.58) and which is optimal, by subtracting $\min\{\alpha_i, \alpha_i^*\}$ *from both multipliers.*

Give a mechanical interpretation of this result, in terms of forces on the SVs (cf. Chapter 7).

9.2 (SV Regression with Fewer Slack Variables ••) *Prove geometrically that in SV regression, we always have $\xi_i \xi_i^* = 0$. Argue that it is therefore sufficient to just introduce slacks ξ_i and use them in both (9.9) and (9.10). Derive the dual problem and show that it is identical to (9.17) except for a modified constraint $0 \le \alpha_i + \alpha_i^* \le C$. Using the result of Problem 9.1, prove that this problem is equivalent to(9.10).*

Hint: although the number of slacks is half of the original quantity, you still need both α_i and α_i^* to deal with the constraints.

9.3 (ν-Property from the Primal Objective Function •) *Try to understand the ν-property from the primal objective function (9.31). Assume that at the point of the solution, $\varepsilon > 0$, and set $(\partial/\partial\varepsilon)\tau(\mathbf{w}, \varepsilon)$ equal to 0.*

9.4 (One-Sided Regression ••) *Consider a situation where you are seeking a flat function that lies above all of the data points; that is, a regression that only measures errors in one direction. Formulate an SV algorithm by starting with the linear case and later introducing kernels. Generalize to the soft margin case, using the ν-trick. Discuss the applicability of such an algorithm. Also discuss how this algorithm is related to ν-SVR using different values of ν for the two sides of the tube.*

9.5 (Basis Pursuit ••) *Formulate a basis pursuit variant of SV regression, where, starting from zero, SVs are added iteratively in a greedy way (cf. [577]).*

9.6 (SV Regression with Hard Constraints •) *Derive dual programming problems for variants of ε-SVR and ν-SVR where all points are required to lie inside the ε-tubes (in* *other words, without slack variables ξ_i). Discuss how they relate to the problems that can be obtained from the usual duals by letting the error penalization C tend to infinity.*

9.7 (Modulus of Continuity vs. Margin •) *Discuss how the ε-margin (Definition 9.1) is related (albeit not identical) to the* modulus of continuity *of a function: given $\delta > 0$, the latter measures the largest difference in function values which can be obtained using points within a distance δ in E.*

9.8 (Margin of Continuous Functions [481] •) *Give an example of a continuous function f for which $m_\varepsilon(f)$ is zero.*[7]

9.9 (Margin of Uniformly Continuous Functions [481] •) *Prove that $m_\varepsilon(f)$ (Definition 9.1) is positive for all $\varepsilon > 0$ if and only if f is uniformly continuous.*[8]

9.10 (Margin of Lipschitz-Continuous Functions [481] •) *Prove that if f is Lipschitz-continuous, meaning that if there exists some $L > 0$ such that for all $x, x' \in E$, $\|f(x) - f(x')\|_G \leq L \cdot \|x - x'\|_E$, then $m_\varepsilon \geq \frac{2\varepsilon}{L}$.*

9.11 (SVR as Margin Maximization [481] •) *Suppose that E (Definition 9.1) is endowed with a dot product $\langle .,. \rangle$ (generating the norm $\|.\|_E$). Prove that for linear functions (9.2), the margin takes the form $m_\varepsilon(f) = \frac{2\varepsilon}{\|\mathbf{w}\|}$. Argue that for fixed $\varepsilon > 0$, maximizing the margin thus amounts to minimizing $\|\mathbf{w}\|$, as done in SV regression with hard constraints.*

9.12 (ε-Margin and Canonical Hyperplanes [481] •) *Specialize the setting of Problem 9.11 to the case where $\mathcal{X} = \{x_1, \ldots, x_m\}$, and show that $m_1(f) = \frac{2}{\|\mathbf{w}\|}$ is equal to (twice) the margin defined for Vapnik's* canonical hyperplane *(Definition 7.1). Argue that the parameter ε is superfluous in pattern recognition.*

9.13 (SVR for Vector-Valued Functions [481] ∘∘∘) *Assume $E = \mathbb{R}^N$. Consider linear functions $f(x) = Wx + b$, with W being an $N \times N$ matrix, and $b \in \mathbb{R}^N$. Give a lower bound on $m_\varepsilon(f)$ in terms of a matrix norm* compatible *[247] with $\|.\|_E$, using the solution of Problem 9.10.*

Consider the case where the matrix norm is induced by $\|.\|_E$, which is to say there exists a unit vector $\mathbf{z} \in E$ such that $\|W\mathbf{z}\|_E = \|W\|$. Give an exact expression for $m_\varepsilon(f)$.

Show that for the Hilbert-Schmidt norm $\|W\|_2 = \sqrt{\sum_{i,j=1}^N W_{ij}^2}$, which is compatible with the vector norm $\|.\|_2$, the problem of minimizing $\|W\|$ subject to separate constraints for each output dimension separates into N regression problems.

7. A function $f : E \to G$ is called *continuous* if for every $\delta > 0$ and $x \in E$, there exists an $\epsilon > 0$ such that *for all* $x' \in E$ satisfying $\|x - x'\|_E < \epsilon$, we have $\|f(x) - f(x')\|_G < \delta$.
8. A function $f : E \to G$ is called *uniformly continuous* if for every $\delta > 0$ there exists an $\epsilon > 0$ such that *for all* $x, x' \in E$ satisfying $\|x - x'\|_E < \epsilon$, we have $\|f(x) - f(x')\|_G < \delta$.

Try to conceive more interesting cases where the regression problems are coupled.[9]

9.14 (Multi-Class Problems ○○○) *Try to generalize $m_\varepsilon(f)$ to multi-class classification problems, and use it to conceive useful margin maximization algorithms for this case.*

9.15 (SV Regression With Overall ε-Insensitive Loss ●●) *Instead of (9.5), consider the objective function*

$$\frac{1}{2}\|\mathbf{w}\|^2 + C \cdot \frac{1}{m}\left|\sum_{i=1}^{m} |y_i - f(\mathbf{x}_i)|\right|_\varepsilon . \tag{9.59}$$

Note that this allows for an overall *ℓ_1 error of ε which is "for free." Therefore, poor performance on some of the points can, to some extent, be compensated for by high accuracies on other points (Figure 9.14). Show that this leads to the kernelized dual problem of maximizing*

$$W(\boldsymbol{\alpha}^{(*)}, \beta) = \sum_{i=1}^{m}(\alpha_i^* - \alpha_i)y_i - \beta\varepsilon - \frac{1}{2}\sum_{i,j=1}^{m}(\alpha_i^* - \alpha_i)(\alpha_j^* - \alpha_j)k(\mathbf{x}_i, \mathbf{x}_j), \tag{9.60}$$

subject to

$$\sum_{i=1}^{m}(\alpha_i - \alpha_i^*) = 0, \;\; 0 \le \alpha_i^{(*)} \le \frac{\beta}{m}, \;\; 0 \le \beta \le C. \tag{9.61}$$

Hint: Introduce slacks $\eta_i^{(*)} \ge 0$ which measure the deviation at each point, and put an ε-insensitive constraint on their sum. Introduce another slack $\xi \ge 0$ for allowing violations of that constraint, and penalize $C\xi$ in the primal objective function.[10]

9.16 (Computation of ε and b in ν-SVR ●) *Suppose i and j are the indices of two points such that $0 < \alpha_i < C/m$ and $0 < \alpha_j^* < C/m$ ("in-bound SVs"). Compute ε and b by exploiting that the KKT conditions imply (9.32) and (9.33) become equalities with $\xi_i = 0$ and $\xi_j^* = 0$.*

9.17 (Parametric ν-SVR Dual ●●) *Derive the dual optimization problem of ν-SVR with parametric loss models (Section 9.5).*

9.18 (Parametric ν-Property ●●●) *Prove Proposition 9.5.*

9.19 (Heteroscedastic Noise ●●●) *Combine ν-SVR using parametric tubes with a variance (e.g., [488]) or quantile estimator (Section 9.3) to construct a SVR algorithm that can deal with heteroscedastic noise.*

---

9. Cf. also Chapter 4, where it is shown that under certain invariance conditions, the regularizer has to act on the output dimensions separately and identically (that is, in a scalar fashion). In particular it turns out that under the assumption of quadratic homogeneity and permutation symmetry, the Hilbert-Schmidt norm is the only admissible norm.
10. This problem builds on joint work with Bob Williamson.

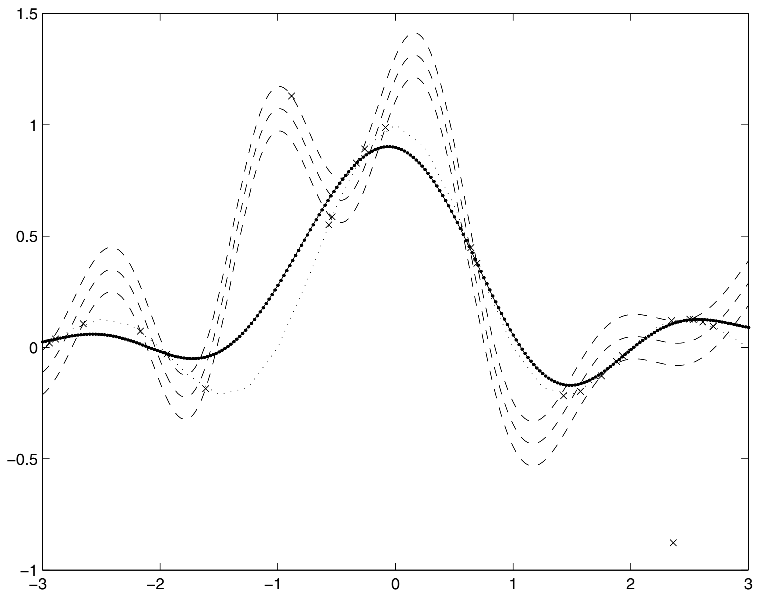


Figure 9.14 Solid line: SVR with overall ε-insensitive loss, dashed lines: standard ε-SVR, with $\pm\varepsilon$ tube. $m = 25$ data points were generated as follows: x-values drawn from uniform distribution over $[-3, 3]$, y-values computed from $y = \sin(\pi x)/(\pi x)$; to create two outliers, the y values of two random points were changed by ± 1. The SV machine parameters were $\varepsilon = 0.1, C = 10000$. For this value of C, which is essentially the hard margin case, the original SVM does a poor job as it tries to follow the outliers. The alternative approach, with a value of ε adjusted such that the overall ℓ_1 error is the same as before, does better, as it is willing to "spend" a large part of its ε on the outliers.

9.20 (SV Regression using ν and ε ∘∘∘) *Try to come up with a formulation of SV regression which uses ν and ε rather than C and ε (ε-SVR) or C and ν (ν-SVR).*

9.21 (ν-SV Regression with Huber's Loss Function ∘∘∘) *Try to generalize ν-SV regression to use loss functions other than the ε-insensitive one, such as the Huber loss, which is quadratic inside the ε-tube and linear outside (cf. Chapter 3).*

Study the relationship between ν and the breakdown point of the estimator [251].

9.22 (Relationship to "Almost Exact Interpolation" ∘∘∘) *Discuss the relationship between SVR and Powell's algorithm for interpolation with thin plate spline kernels [422]. Devise a variant of Powell's algorithm that uses the ε-insensitive loss.*