Support vector machines

Chmiela Bartosz

University of Wrocław

25.04.2022

Contents

Introdution

- generalizations of linear decision boundaries,
- optimal separating hyperplanes for classes linearly separable,
- nonlinear boundaries for nonseparable classes,
- generalizations of Fisher's linear discriminant analysis.

The support vector classifier

Training data consists of N pairs $(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)$ with $x_i \in \mathbb{R}^p$ and $y_i \in \{-1, 1\}$. Define a hyperplane by

$$\left\{x : \ f(x) = x^T\beta + \beta_0 = 0\right\},$$

where β is a unit vector: $\|\beta\| = 1$. A classification rule induced by $f(x)$ is

$$G(x) = \operatorname{sign}\left[x^T\beta + \beta_0\right].$$

The support vector classifier

Since the classes are separable, we can find a function $f(x)$ with

$$\forall i \; y_i f(x_i) > 0.$$

Hence we are able to find the hyperplane that creates the biggest *margin* between the training points. The optimization problem

$$\max_{\beta, \beta_0, \|\beta\|=1} M$$
$$\text{s.t. } y_i(x_i^T \beta + \beta_0) \geq M, \; i = 1, \ldots, N,$$

captures this concept.

The support vector classifier

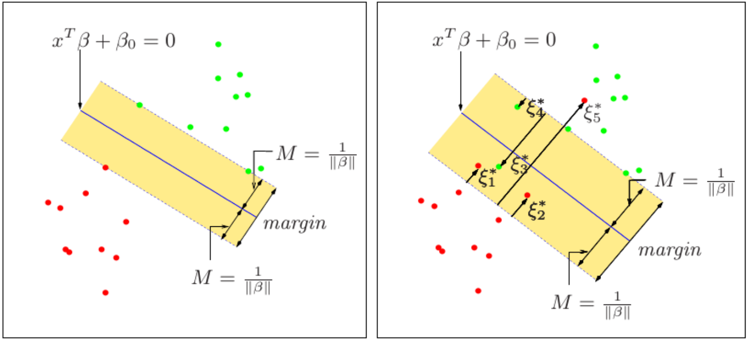


Figure: Support vector classifiers. The left panel shows the separable case. The right panel shows the nonseparable (overlap) case. Source: *The elements of statistical learning, T Hastie, R Tibshirani, JH Friedman, fig. 12.1*.

The support vector classifier

This problem can abe more conveniently rephrased as

$$\min_{\beta,\beta_0} \|\beta\|$$
$$\text{s.t. } y_i(x_i^T\beta + \beta_0) \geq 1, \; i = 1, \dots, N.$$

Note that $M = \frac{1}{\|\beta\|}$. This is a convex optimization problem.

The support vector classifier

Suppose now that the classes overlap in feature space. Define the slack variables $\xi - (\xi_1, \xi_2, \ldots, \xi_N)$. There are two natural ways to modify the constraint:

$$y_i(x_i^T\beta + \beta_0) \geq M - \xi_i,$$
$$\text{or}$$
$$y_i(x_i^T\beta + \beta_0) \geq M(1 - \xi_i),$$

where $\forall i\ \xi_i \geq 0,\ \sum_{i=1}^N \xi_i \leq \text{const}.$

The support vector classifier

- The value ξ_i in the constraint $y_i(x_i^T\beta + \beta_0) \geq M(1 - \xi_i)$ is the proportional amount by which the prediction is on the wrong side of its margin,
- misclassifications occur when $\xi_i > 1$,
- by bounding $\sum_{i=1}^N \xi_i$, we bound the total proportional amount of misclassifications,
- so when $\sum_{i=1}^N \xi_i < K$, the total number of training misclassifications are bounded at K.

The support vector classifier

Again, we can define $M = \frac{1}{||\beta||}$, and write in the equivalent form

$$\min_{\beta,\beta_0} ||\beta||$$
$$\text{s.t. } y_i(x_i^T\beta + \beta_0) \geq 1 - \xi_i,\ i = 1, \ldots, N,$$
$$\text{and } \xi_i \geq 0,\ \sum_{i=1}^{N} \xi_i \leq \text{const}.$$

This is the usual way the *support vector classifier* is defined for the nonseparable case.
By nature of this criterion, points well inside their class do not play a big role.

The support vector classifier

Computationally it is convenient to re-express in the equivalent form

$$\min_{\beta,\beta_0} \frac{1}{2}\|\beta\|^2 + C\sum_{i=1}^{N}\xi_i$$
$$\text{s.t. } y_i(x_i^T\beta + \beta_0) \geq 1 - \xi_i,\ \xi_i \geq 0\ \forall i,$$

where the "cost" parameter C replaces the bounding constant. The separable case corresponds to $C = \infty$.

The support vector classifier

The Lagrange (primal) function is

$$L_p = \frac{1}{2}\|\beta\|^2 + C\sum_{i=1}^{N}\xi_i - \sum_{i=1}^{N}\alpha_i\left[y_i\left(x_i^T\beta + \beta_0\right) - (1-\xi_i)\right] - \sum_{i=1}^{N}\mu_i\xi_i,$$

which we minimize w.r.t β, β_0 and ξ_i.

The support vector classifier

Setting the respective derivatives to zero, we get

$$\beta = \sum_{i=1}^{N} \alpha_i y_i x_i,$$

$$0 = \sum_{i=1}^{N} \alpha_i y_i,$$

$$\alpha_i = C - \mu_i, \forall i,$$

as well as the positivity constraints $\alpha_i, \mu_i, \xi_i \geq 0, \ \forall i$.

The support vector classifier

By substituting into, we obtain the Lagrangian (Wolfe) dual objective function

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j x_i^T x_j,$$

which gives a lower bound on the objective function.

The support vector classifier

We maximize L_D s.t. $0 \leq \alpha_i \leq C$ and $\sum_{i=1}^{N} \alpha_i y_i = 0$.

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j x_i^T x_j,$$

In addition to equations obtained from derivatives, the Karush-Kuhn-Tucker conditions include the constraints

$$\begin{aligned} \alpha_i \left[y_i \left(x_i^T \beta + \beta_0 \right) - (1 - \xi_i) \right] &= 0, \\ \mu_i \xi_i &= 0, \\ \alpha_i \left[y_i \left(x_i^T \beta + \beta_0 \right) - (1 - \xi_i) \right] &\geq 0, \end{aligned}$$

Together these equations uniquely characterize the solution to the primal and dual problem.

The support vector classifier

We see that the solution for β has the form

$$\hat{\beta} = \sum_{i=1}^{N} \hat{\alpha}_i y_i x_i,$$

with nonzero coefficients $\hat{\alpha}_i$ only for those observations i for which the constraints are exactly met. These observations are called the *support vectors*, since $\hat{\beta}$ is represented in terms of them alone.

The support vector classifier

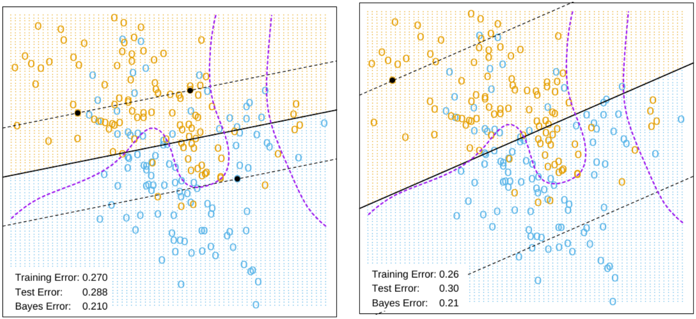
Maximizing the dual L_D is a simpler convex quadratic programming problem than the primal L_P, and can be solved with standard techniques.

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j x_i^T x_j,$$

$$L_p = \frac{1}{2} \|\beta\|^2 + C \sum_{i=1}^{N} \xi_i - \sum_{i=1}^{N} \alpha_i \left[y_i \left(x_i^T \beta + \beta_0 \right) - (1 - \xi_i) \right]$$

$$- \sum_{i=1}^{N} \mu_i \xi_i,$$

The support vector classifier



$C = 10000$ $C = 0.01$

Figure: The linear support vector boundary for the mixture data example with two overlapping classes, for two different values of C. The broken purple curve in the background is the Bayes decision boundary. Source: *The elements of statistical learning, T Hastie, R Tibshirani, JH Friedman, fig. 12.2*.

Support vector machines and kernels

- enlarge the feature space using basis expansions,
- select basis functions $h_m(x),\ m = 1, \ldots, M$,
- fit the SV classifier using input features $h(x_i) = (h_1(x_i), \ldots, h_M(x_i)),\ i = 1, \ldots, N$,
- produce the (nonlinear) function $\hat{f}(x) = h(x)^T \hat{\beta} + \hat{\beta}_0$,
- classifier is $\hat{G}(x) = \mathrm{sign}(\hat{f}(x))$,
- the SVM classifier is an extension of this idea,
- dimension of the enlarged space is allowed to get very large.

Computing the SVM for classification

We can represent the optimization problem and its solution in a special way that only involves the input features via inner products,

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \langle h(x_i), h(x_j) \rangle,$$

and the solution function $f(x)$ can be written:

$$f(x) = h(x)^T \beta + \beta_0 = \sum_{i=1}^{N} \alpha_i y_i \langle h(x), h(x_i) \rangle + \beta_0.$$

As before, given α_i, β_0 can be determined by solving $y_i f(x_i) = 1$.

Computing the SVM for classification

So both of these equations involve $h(x)$ only through inner products and require only knowledge of the kernel function

$$K(x, x') = \langle h(x), h(x') \rangle.$$

K should be a symmetric positive (semi-) definite function. Three popular choices for K in the SVM literature are

- dth-degree polynomial: $K(x, x') = (1 + \langle x, x' \rangle)^d$,
- Radial basis: $K(x, x') = \exp(-\gamma \|x - x'\|^2)$,
- Neural network $K(x, x') = \tanh(\kappa_1 \langle x, x' \rangle + \kappa_2)$.

Computing the SVM for classification

Consider for example a feature space with two inputs X_1 and X_2, and a polynomial kernel of degree 2. Then

$$
\begin{aligned}
K(X, X') &= \big(1 + \langle X, X' \rangle\big)^2 = \\
&= \big(1 + X_1 X_1' + X_2 X_2'\big)^2 = \\
&= 1 + 2X_1X_1' + 2X_2X_2' + \big(X_1X_1'\big)^2 + \big(X_2X_2'\big)^2 + \\
&+ 2X_1X_1'X_2X_2'.
\end{aligned}
$$

Computing the SVM for classification

Then $M = 6$, and if we choose $h_1(X) = 1$, $h_2(X) = \sqrt{2}X_1$, $h_3(X) = \sqrt{2}X_2$, $h_4(X) = X_1^2$, $h_5(X) = X_2^2$, $h_6(X) = \sqrt{2}X_1X_2$, then

$$K(X, X') = \langle h(X), h(X') \rangle.$$

and we see that the solution can be written

$$\hat{f}(x) = \sum_{i=1}^{N} \hat{\alpha}_i y_i K(x, x_i) + \hat{\beta}_0.$$

Computing the SVM for classification

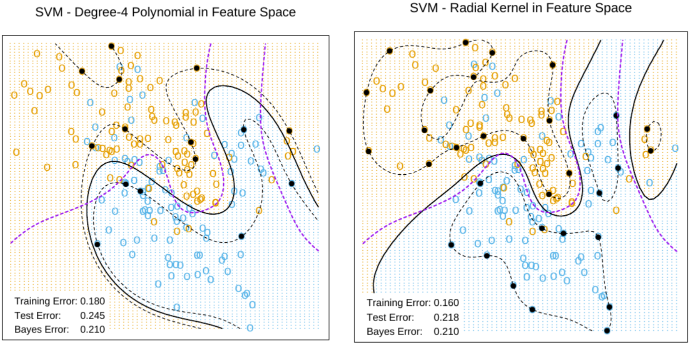


Figure: Two nonlinear SVMs for the mixture data. The left plot uses a 4th degree polynomial kernel, the right a radial basis kernel (with $\gamma = 1$). Source: *The elements of statistical learning, T Hastie, R Tibshirani, JH Friedman, fig. 12.3*.

The SVM as a penalization model

With $f(x) = h(x)^T \beta + \beta_0$, consider the optimization problem

$$\min_{\beta, \beta_0} \sum_{i=1}^{N} [1 - y_i f(x_i)]_+ + \frac{\lambda}{2} \|\beta\|^2$$

where the subscript "+" indicates positive part. This has the form *loss* + *penalty*. When $\lambda = 1/C$ then the solution is the same as in the beginning.

The SVM as a penalization model

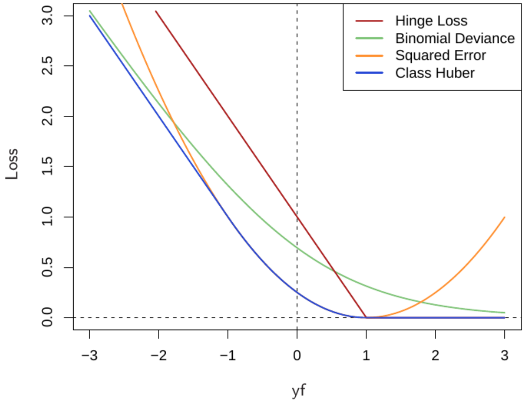


Figure: The support vector loss function (hinge loss), compared to the negative log-likelihood loss (binomial deviance) for logistic regression, squared-error loss, and a "Huberized" version of the squared hinge loss. Source: *The elements of statistical learning, T Hastie, R Tibshirani, JH Friedman, fig. 12.4*.

Support vector machines for regression

We first discuss the linear regression model

$$f(x) = x^T \beta + \beta_0,$$

and then handle nonlinear generalizations. To estimate β, we consider minimization of

$$H(\beta, \beta_0) = \sum_{i=1}^{N} V\left(y_i - f(x_i)\right) + \frac{\lambda}{2}\|\beta\|^2.$$

where V is error measure.

Support vector machines for regression

ε-insensitive error measure

$$V_\varepsilon(r) = \begin{cases} 0 & \text{if } |r| < \varepsilon, \\ |r| - \varepsilon & \text{otherwise,} \end{cases}$$

error measure used in robust regression in statistics

$$V_H(r) = \begin{cases} r^2/2, & \text{if } |r| \leq c, \\ c|r| - c^2/2, & |r| > c, \end{cases}$$

Support vector machines for regression

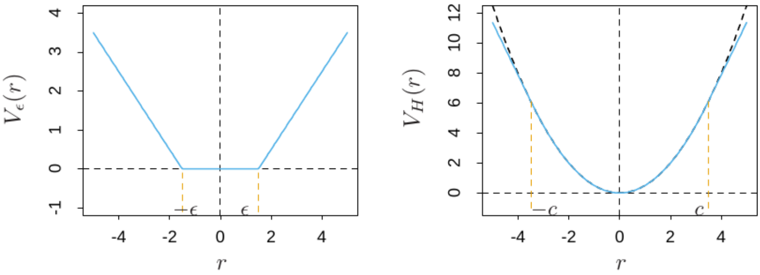


Figure: The left panel shows the ε-insensitive error function used by the support vector regression machine. The right panel shows the error function used in Huber's robust regression (blue curve). Beyond $|c|$, the function changes from quadratic to linear. Source: *The elements of statistical learning, T Hastie, R Tibshirani, JH Friedman, fig. 12.8*.

Support vector machines for regression

If $\hat{\beta}, \hat{\beta}_0$ are the minimizers of H, the solution function can be shown to have the form

$$\hat{\beta} = \sum_{i=1}^{N} (\hat{\alpha}_i^* - \hat{\alpha}_i) x_i,$$

$$\hat{f}(x) = \sum_{i=1}^{N} (\hat{\alpha}_i^* - \hat{\alpha}_i) \langle x, x_i \rangle + \beta_0,$$

Support vector machines for regression

Where $\hat{\alpha}_i^*, \hat{\alpha}_i$ are positive and solve the quadratic programming problem

$$\min_{\hat{\alpha}_i^*, \hat{\alpha}_i} \varepsilon \sum_{i=1}^{N} (\hat{\alpha}_i^* + \hat{\alpha}_i) - \sum_{i=1}^{N} y_i (\hat{\alpha}_i^* - \hat{\alpha}_i) + \\ + \frac{1}{2} \sum_{i=1, j=1}^{N} (\hat{\alpha}_i^* - \hat{\alpha}_i)(\hat{\alpha}_j^* - \hat{\alpha}_j) \langle x_i, x_j \rangle$$

subject to the constraints

$$0 \leq \alpha_i,\ \alpha_i^* \leq 1/\lambda,$$

$$\sum_{i=1}^{N} (\alpha_i^* - \alpha_i) = 0$$

$$\alpha_i^* \alpha_i = 0.$$

Regression and kernels

Suppose we consider approximation of the regression function in terms of a set of basis functions $\{h_m(x)\}, m = 1, 2, \ldots, M$:

$$f(x) = \sum_{m=1}^{M} \beta_m h_m(x) + \beta_0.$$

To estimate β and β_0 we minimize

$$H(\beta, \beta_0) = \sum_{i=1}^{N} V\left(y_i - f(x_i)\right) + \frac{\lambda}{2} \sum_{m=1}^{M} \beta_m^2.$$

for some general error measure $V(r)$.

Regression and kernels

For any choice of $V(r)$, the solution $\hat{f}(x) = \sum_{m=1}^{M} \hat{\beta}_m h_m(x) + \hat{\beta}_0$ has the form

$$\hat{f}(x) = \sum_{i=1}^{N} \hat{\alpha}_i K(x, x_i)$$

with $K(x, y) = \sum_{m=1}^{M} h_m(x) h_m(y)$.

Regression and kernels

Let's work out the case $V(r) = r^2$. Let $\mathbf{H} \in \mathbb{R}^{N \times M}$ be basis matrix with imth element $h_m(x_i)$, and suppose that $M > N$ is large. We assume that $\beta_0 = 0$, or that the constant is absorbed in h. Estimate β by minimizing the penalized least squares criterion

$$\mathbf{H}(\beta) = (\mathbf{y} - \mathbf{H}\beta)^T(\mathbf{y} - \mathbf{H}\beta) + \lambda\|\beta\|^2.$$

The solution is

$$\hat{\mathbf{y}} = \mathbf{H}\hat{\beta}$$

with $\hat{\beta}$ determined by

$$-\mathbf{H}^T(\mathbf{y} - \mathbf{H}\beta) + \lambda\hat{\beta} = 0.$$

Regression and kernels

We need to evaluate the $M \times M$ matrix of inner products in the transformed space. However, we can premultiply by $\mathbf{H}$ to give

$$\mathbf{H}\hat{\beta} = (\mathbf{H}\mathbf{H}^T + \lambda\mathbf{I})^{-1}\mathbf{H}\mathbf{H}^T\mathbf{y}.$$

The $N \times N$ matrix $\mathbf{H}\mathbf{H}^T$ consists of inner products between pairs of observations i, j. The evaluation of an inner product kernel $\{\mathbf{H}\mathbf{H}^T\}_{i,j} = K(x_i, x_j)$.

Regression and kernels

The predicted values at an arbitrary x satisfy

$$\hat{f}(x) = h(x)^T \hat{\beta} = \sum_{i=1}^{N} \hat{\alpha}_i K(x, x_i),$$

where $\hat{\alpha} = (\mathbf{H}\mathbf{H}^T + \lambda \mathbf{I})^{-1}\mathbf{y}$. As in the support vector machine, we need not specify or evaluate the large set of functions $h_1(x), h_2(x), \ldots, h_M(x)$.

Thank you for your attention.