Support Vector Machine (SVM)

Jitkomut Songsiri

Department of Electrical Engineering
Faculty of Engineering
Chulalongkorn University

CUEE

March 15, 2023

Outline

1. Separating hyperplane
2. Hard and soft margin classifier
3. Computation of SVC and the dual
4. SVM: Nonlinearity and Kernels
5. Related formulations, extensions, and algorithm
6. Support Vector Regression (SVR)

Separating hyperplane

Linear function

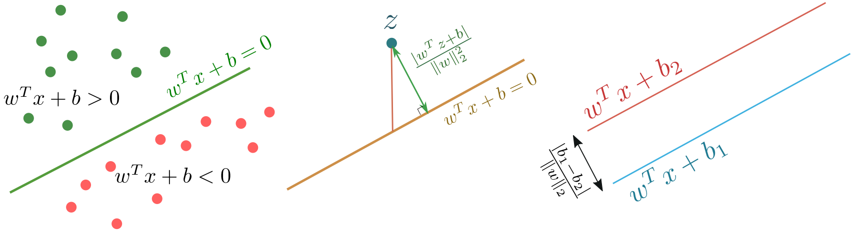
a linear function $f : \mathbf{R}^n \to \mathbf{R}$ is of the form

$$f(x) = w^T x = w_1 x_1 + w_2 x_2 + \cdots + w_n x_n$$

- $w = (w_1, w_2, \ldots, w_n)$ is a given parameter
- the contour of f is a hyperplane with the normal vector w
- $\nabla f(x) = w$ (constant, not depend on x)
- for $b \neq 0$, $f(x) = w^T x + b$ is called an affine function
- ✎ the ℓ_2-norm distance from a point z to the hyperplane $w^T x + b = 0$ is $|w^T z + b| / \|w\|_2$
- ✎ the distance between two parallel hyperplanes described by $w^T x + b_1$ and $w^T x + b_2$ is $|b_1 - b_2| / \|w\|_2$

Halfspaces

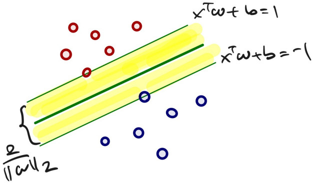
a hyperplane splits the space into two halfspaces



- for a given x, finding w, b so that $w^T x + b > 0$ can have many solutions because the linear inequality is homogeneous in w and b
- many ways to restrict some solutions:
  - find w, b so that $w^T x + b > M$ (just add a constant M)

Separating hyperplane

setting: given $\{(x_i, y_i)\}_{i=1}^N$ where $x_i \in \mathbf{R}^n$ are data with label $y_i \in \{1, -1\}$



modeling:

- the goal is to find a hyperplane $x^T w + b$ to classify data into two classes
- the distance between two hyperplanes $x^T w + b = \pm 1$ is $2/\|w\|_2$
- feasibility problem: for $i = 1, 2, \ldots, N$, data from each class satisfy

$$y_i = 1 : x_i^T w + b \geq 1, \text{ and } y_i = -1 : x_i^T w + b \leq -1 \quad \Rightarrow \quad y_i(x_i^T w + b) \geq 1$$

Hard and soft margin classifier

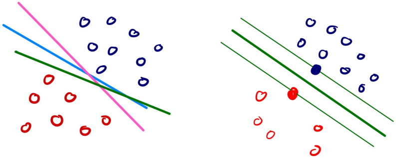
Hard-margin classifier

problem parameters: $x_i \in \mathbf{R}^n$ and $y_i \in \{-1, 1\}$ for $i = 1, \ldots, N$

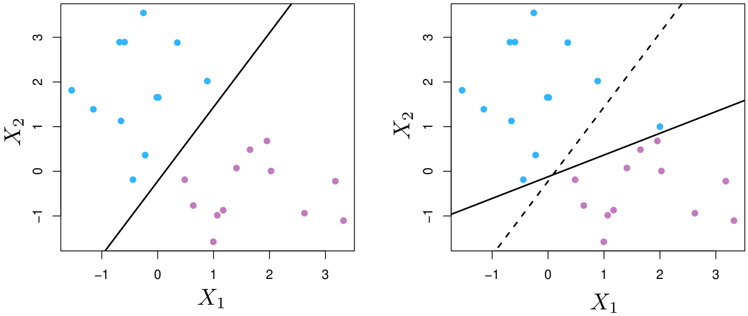
optimization variables: $w \in \mathbf{R}^n, b \in \mathbf{R}$

$$\text{minimize} \quad \|w\|_2^2 \quad \text{subject to} \quad y_i(x_i^T w + b) \geq 1, \; i = 1, 2, \ldots, N$$

- data are classified by separating hyperplane with maximized margin (right figure)
- if feasible, the data from two classes are separated perfectly
- the problem is a convex quadratic program (QP)
- the decision boundary pass through points from both classes– these points are called **support vectors**



Sensitivity to individual observations



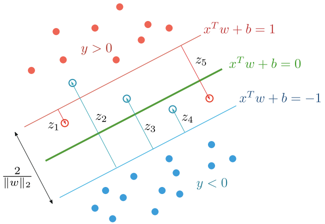
- *left:* hard-margin classifier with max margin
- *right:* by only adding a pair of data, the hyperplane dramatically changes; it may overfit the training data
- having the max-margin is no longer useful – we need something more robust to individual observations

Soft-margin support vector classifier (C-SVC)

problem parameters: $x_i \in \mathbf{R}^n$ and $y_i \in \mathbf{R}$ for $i = 1, \ldots, N, C > 0$

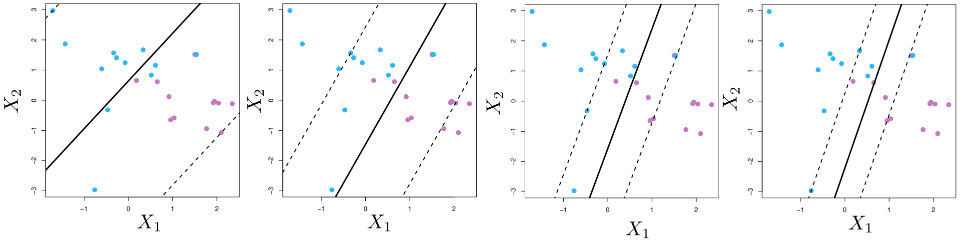
optimization variables: $w \in \mathbf{R}^n, b \in \mathbf{R}, z \in \mathbf{R}^N$

$$
\begin{array}{ll}
\text{minimize} & (1/2)\|w\|_2^2 + C\mathbf{1}^T z \\
\text{subject to} & y_i(x_i^T w + b) \geq 1 - z_i, \quad i = 1, 2, \ldots, N \\
& z \succeq 0
\end{array}
$$



- z_i is called a **slack variable**, allowing some of the hard constraints to be relaxed
- if $z_i > 0$ at optimum, the ith point is relaxed to be on the wrong side of its class
- the **regularization (penalty) parameter** C controls the trade-off between maximizing the margin and the total distance of points on the wrong side
- the problem is a convex quadratic program

Varying penalty parameter (C)



- *left.* C is the smallest (low penalty for observations being on the wrong side, so $\|w\|_2^2$ is small and the margin is large); C is larger from left to right
- when C is large, we get narrow margins that are rarely violated and the classifier is highly fit to the data (low bias, high variance)
- C is typically chosen via a cross-validation

Classification rule

after we have trained the classifier and obtain $\hat{w}, \hat{b}$, the class prediction based on a new input x is

$$\hat{y} = \hat{f}(x) = \mathbf{sign}(x^T \hat{w} + \hat{b}) = \begin{cases} 1, & x^T \hat{w} + \hat{b} \geq 0, \\ -1, & x^T \hat{w} + \hat{b} < 0 \end{cases}$$

it turns out that $\hat{w}$ and $\hat{b}$ are computed using only *some* of the training observations

this can be explained by the **optimality conditions** for the soft-margin SVC problem

Computation of SVC and the dual

Derivation of dual

let α and λ be **Lagrange multipliers** (w.r.t. 1st and 2nd inequalities on page 10)

$$L(w,b,z,\alpha,\lambda) = \frac{1}{2}\|w\|_2^2 - \sum_{i=1}^N \alpha_i y_i x_i^T w - b\sum_{i=1}^N \alpha_i y_i + (C\mathbf{1} - \alpha - \lambda)^T z + \mathbf{1}^T \alpha$$

note that L is quadratic in w: $\frac{1}{2}\|w\|_2^2 - d^T w$ and L is linear in b and z

- $\inf_w L$ occurs when $w = d = \sum_i \alpha_i y_i x_i$ and the infimum is

$$-(1/2)\|d\|_2^2 = -(1/2)d^T d = -(1/2)\sum_i \sum_j \alpha_i \alpha_j y_i y_j x_i^T x_j$$

- since L is linear in z, b, $\inf_z L$ and $\inf_b L$ exist (and are zero) only when

$$\sum_i \alpha_i y_i = 0, \quad C\mathbf{1} - \alpha - \lambda = 0$$

- dual function: $g(\alpha) = -(1/2)\sum_i \sum_j \alpha_i \alpha_j y_i y_j x_i^T x_j + \mathbf{1}^T \alpha$

Karush-Kuhn-Tucker conditions of soft-margin SVC

primal feasibility: $y_i(x_i^T w + b) \geq 1 - z_i, \quad i = 1, 2, \ldots, N,$

$z \succeq 0$

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

$0 \leq \alpha_i \leq C, \quad i = 1, 2, \ldots, N$

or equivalently, $\lambda \succeq 0, \quad \alpha = C\mathbf{1} - \lambda$

zero-gradient of L: $w = \sum_{i=1}^N \alpha_i y_i x_i$

complementary slackness: $\alpha_i[y_i(x_i^T w + b) - (1 - z_i)] = 0$

$\lambda_i z_i = 0, \quad i = 1, 2, \ldots, N$

Implications of SVC's KKT

dual feasibility and complementary slackness characterize three groups of points

$$\alpha_i = C - \lambda_i, \quad \lambda_i z_i = 0, \quad \alpha_i[y_i(x_i^T w + b) - (1 - z_i)] = 0$$

correct side of the margin

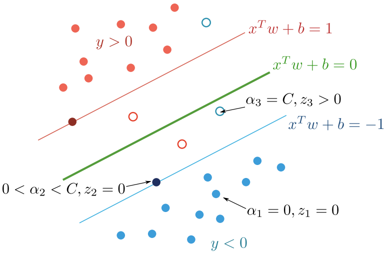
$$\alpha_i = 0, \quad \lambda_i = C, \quad z_i = 0, \quad y_i(x_i^T w + b) \geq 1$$

edge of the margin

$$0 < \alpha_i < C, \quad \lambda_i > 0, \quad z_i = 0, \quad y_i(x_i^T w + b) = 1$$

wrong side of the margin

$$\alpha_i = C, \quad \lambda_i = 0, \quad y_i(x_i^T w + b) = 1 - z_i, \quad z_i > 0$$



- the observations x_i for which $\alpha_i > 0$ are called **support vectors** because w is a linear combination of only those terms: $w = \sum_{i=1}^N \alpha_i y_i x_i$
- margin points: $y_i(x_i^T w + b) = 1 \Leftrightarrow b = -x_i^T w + y_i$ (averaging all solutions)

Support vectors

interesting properties of the soft-margin SVC problem on page 10

- observations that lie directly on the margin or on the wrong side of the margin for their class, are known as **support vectors**
- only the observations that are support vectors affect the support vector classifiers
- SVC's decision rule is based only on the support vectors (small subset of training observations), it is robust to the behavior of observations that are far away from the hyperplane
- this is distinct from LDA; LDA classification rule depends on the mean of *all* observations within each class, as well as the covariances of the class conditional distribution (which use *all* observations)

Dual of soft-margin support vector classifier

dual problem of soft-margin classifier on page 10 with variable $\alpha \in \mathbf{R}^N$

$$\begin{array}{ll} \text{maximize}_\alpha & \mathbf{1}^T\alpha - (1/2)\sum_{i=1}^N \sum_{j=1}^N \alpha_i\alpha_j y_i y_j x_i^T x_j \\ \text{subject to} & \sum_{i=1}^N \alpha_i y_i = 0, \quad 0 \leq \alpha_i \leq C, \;\; i = 1, 2, \ldots, N \end{array}$$

or a compact (vector) form

$$\begin{array}{ll} \text{minimize} & (1/2)\alpha^T G\alpha - \mathbf{1}^T\alpha \\ \text{subject to} & \alpha^T y = 0, \;\; 0 \preceq \alpha \preceq C\mathbf{1} \end{array}$$

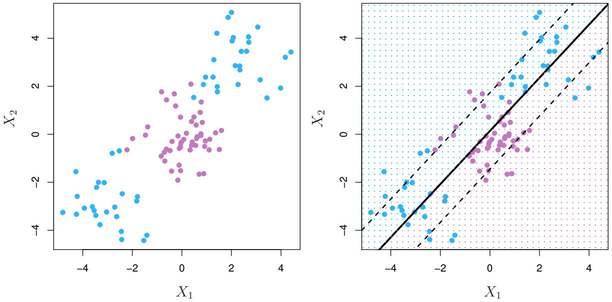
where $G \in \mathbf{R}^{N\times N}$, $G_{ij} = \langle y_i x_i, y_j x_j \rangle$ (called a **Gram** matrix); clearly, $G \succeq 0$

- it is a quadratic program with a linear equality and a box constraint
- this formulation is called C-SVC (C-support vector classification)

SVM: Nonlinearity and Kernels

Non-separable by linear boundary

sometimes we face with nonlinear class boundaries and SVC may perform poorly



instead of fitting SVC using $X_1, \ldots, X_n$, we could map input using nonlinear functions

$$X_1, X_2, \ldots, X_n, X_1^2, X_2^2, \ldots, X_n^2$$

or using nonlinear mappings $h_1(x), h_2(x), \ldots, h_m(x)$ in an enlarged space

How the classifier is computed

the computation involves only the **inner products** of observations: $\langle x, z \rangle = x^T z$

from the KKT conditions, we see that

1. $w = \sum_{i=1}^N \alpha_i y_i x_i$ and the sum can be taken only those terms that $\alpha_i \neq 0$
2. the linear support vector classifier can be represented as

$$f(x) = b + x^T w = b + x^T \sum_{i=1}^N \alpha_i y_i x_i = b + \sum_{i=1}^N \alpha_i y_i \langle x, x_i \rangle$$

   it seems to require $\langle x, x_i \rangle$ between *all* pairs but it actually involves far fewer terms

3. now we can introduce a nonlinearity by replacing the inner product with a generalization in a form of **Kernel functions**:

$$K(x, z) : \mathbf{R}^n \times \mathbf{R}^n \to \mathbf{R} \quad \text{that satisfies certain properties}$$

Support Vector Machine

SVM is an extension of SVC using input features

$$h(x) = (h_1(x), h_2(x), \ldots, h_p(x))$$

and produce the nonlinear function $f(x) = h(x)^T w + b$

- the dimension of the enlarged space is allowed to get very large
- following the dual of SVM (as before), the computation of SVM becomes easier using a **Kernel trick**

$$w = \sum_{i=1}^{N} \alpha_i y_i h(x_i), \quad f(x) = h(x)^T w + b = \sum_{i=1}^{N} \alpha_i y_i \langle h(x), h(x_i) \rangle + b$$

it involves $h(x)$ only through inner products

Primal and dual (nonlinear) SVM

the primal (nonlinear) SVM is to replace the linear function by a nonlinear h

$$\begin{array}{ll} \text{minimize}_{w,b} & (1/2)\|w\|_2^2 + C\mathbf{1}^T z \\ \text{subject to} & y_i(h(x_i)^T w + b) \geq 1 - z_i, \quad i = 1, 2, \ldots, N \\ & z \succeq 0 \end{array}$$

the dual SVM is similar to the dual SVC on page 18

but just replace the inner product with a **kernel function** $K(x, z) = \langle h(x), h(z) \rangle$

$$\begin{array}{ll} \text{maximize}_{\alpha} & \mathbf{1}^T \alpha - (1/2) \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y_i y_j K(x_i, x_j) \\ \text{subject to} & \sum_{i=1}^N \alpha_i y_i = 0, \quad 0 \leq \alpha_i \leq C, \quad i = 1, 2, \ldots, N \end{array}$$

important note: solving SVM on the dual and computing f does NOT require the nonlinear mapping $h(x)$ at all, but only knowledge of the kernel function

Kernel functions

the SVM has the form

$$f(x) = b + \sum_{i=1}^{N} \alpha_i K(x, x_i) \quad (\alpha_i \neq 0 \text{ for support vectors})$$

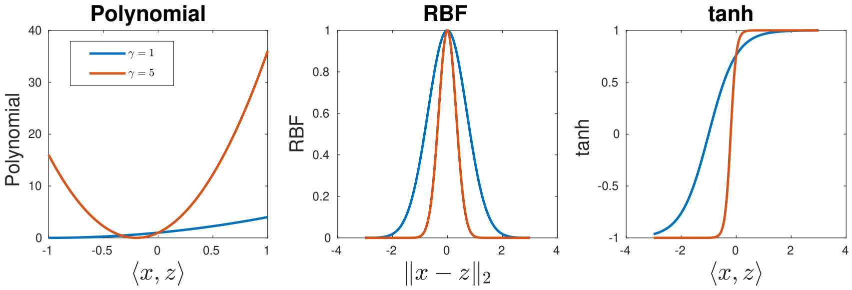
condition: a **kernel** function $K(x, z)$ is symmetric and positive semidefinite

$$K(x, z) = K(z, x), \quad K(x, x) \geq 0$$

1. **linear:** $\langle x, z \rangle = z^T x$: the similarity of a pair using Pearson (standard) correlation
2. **polynomial:** $(\gamma \langle x, z \rangle + r)^d$ where d is a positive integer and r is a coefficient
3. **radial basis function (RBF):** $e^{-\gamma \|x - z\|_2^2}$ where $\gamma > 0$
4. **hyperbolic tangent:** $\tanh(\gamma \langle x, z \rangle + r)$

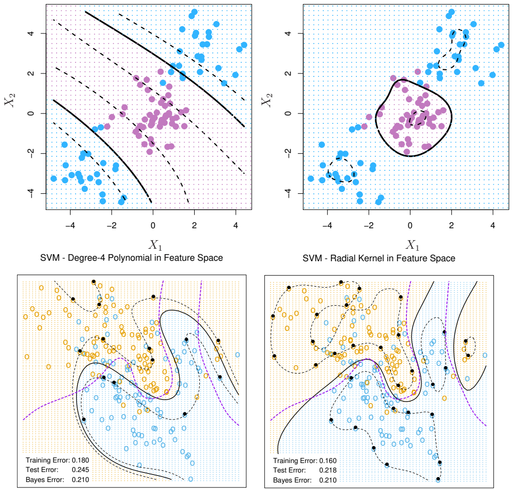
Parameters in kernel functions

set $r = 1, d = 2$ and adjust $\gamma = 1, 5$



- polynomial kernel amounts to fitting SVC in a high-dim space involving polynomials of degree d
- RBF: if x^* (test point) is far from x_i then $K(x^\star, x_i)$ is small; observations far from x^* play a small role in the predicted class label for x^* (RBF has a local behavior)

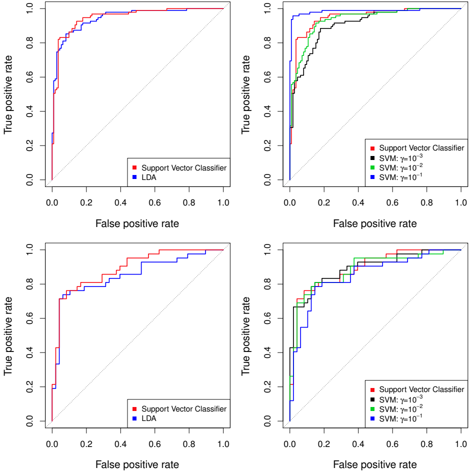
Polynomial and radial basis kernels



- *left.* polynomial *right.* RBF (either kernel is capable of capturing the nonlinear decision boundaries)
- *bottom.* ground truth is mixture Gaussians; RBF performs the best which is close to Bayes optimal

ROC curves tested on heart data

detect heart data using predictors such as age, sex, and cholesterol



- *top.* ROC is evaluated on training dataset
- *top left.* varying threshold $f(x) \leq t$ in LDA and SVC
- *top right.* vary γ of RBF in SVM; as γ increases, the fit is more nonlinear, the ROC improves
- *bottom.* ROC is evaluated on test set; SVMs with $\gamma = 10^{-2}, 10^{-3}$ perform comparably to SVC; SVC has a slight advantage over LDA

How to choose SVM parameters?

C is the penalty parameter common to all choices of kernel

- **high C:** focus on classifying all the training points correctly
- **low C:** less penalty on points on the wrong side; the decision surface is smoother

γ is the decay rate of RBF (in $e^{-\gamma\|x-z\|^2}$)

- γ can be regarded as the inverse of radius of influence a training point has

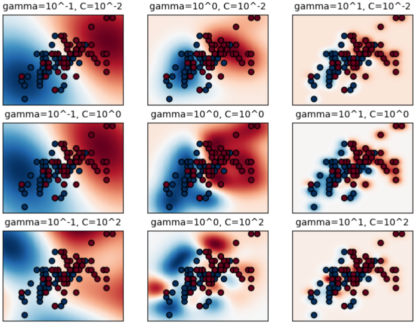
$$\text{same influence} = \text{same } K \quad \Rightarrow \quad \gamma_{\text{small}}\|x_i - x_j\|_2^2 = \gamma_{\text{large}}\|x_i - x_j\|_2^2$$

- **high γ:** only a close single training point can reach
- **low γ:** a far single training point can reach and affect the model

these two parameters affect SVM's performance (typically chosen via cross-validation)

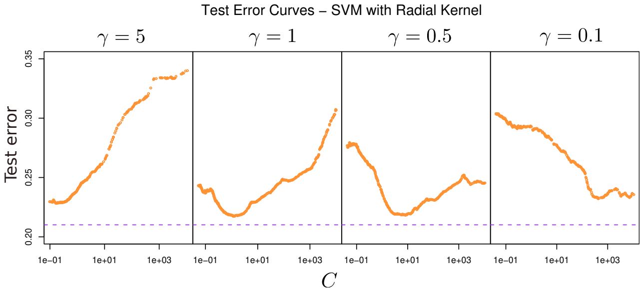
Effect of RBF parameter

decision function in a grid as C and γ of RBF vary



- if γ is too small, the model cannot capture the complexity of data
- intermediate γ gives smooth models that detect data pattern; can be made more complex by increasing C
- if γ is too large, the radius of influence area only includes the support vector itself

figure from `https://scikit-learn.org/stable/auto_examples/svm/plot_rbf_parameters.html` test error as a function of C, using different γ in RBF



- for each γ, choose C that corresponds to the minimum test (cross-validated) error
- when γ is large (narrow peaked kernel), a small C is chosen which is less penalty on misclassified points
- hence, a path algorithm to compute w for many values of C is required – see ESL section 12.3.5

Related formulations, extensions, and algorithm

Hinge primal SVM

- the original hard constraint relates to the **margin-perceptron cost**

$$y_i(x_i^T w + b) \geq 1 \quad \Longleftrightarrow \quad \max(0, 1 - y_i(x_i^T w + b)) = 0$$

- another equivalent problem of soft-margin SVC is to use the **hinge loss**

$$y_i(x_i^T w + b) \geq 1 - z_i, \quad \mathbf{1}^T z = \sum_{i=1}^{N} \max(0, 1 - y_i(x_i^T w + b))$$

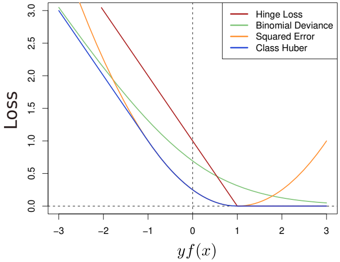
  and put the formulation as a single cost function (aka **hinge primal problem**)

$$\text{minimize} \quad \tfrac{\lambda}{2}\|w\|_2^2 + \textstyle\sum_{i=1}^{N} \max(0, 1 - y_i(x_i^T w + b))$$

  (role of λ is opposite to C in the soft-margin SVC)
- hinge primal SVC can be regarded as a **penalization method**

Loss + Penalty

the hinge primal SVC takes 'loss+penalty' form: minimize$_\beta$ $L(x,y;\beta) + \lambda P(\beta)$



loss	$L(y, f(x))$
binomial deviance	$\log[1 + e^{-yf(x)}]$
SVM hinge	$[1 - yf(x)]_+$
square	$[1 - yf(x)]^2$
Huberized	$\begin{cases} -4yf(x), & yf(x) < -1 \\ [1 - yf(x)]_+^2, & \text{otherwise} \end{cases}$

- $P(\beta)$ is a penalty function on β whose effect is controlled by λ
- **hinge loss** is closely related to **binomial deviance** (logistic regression loss) and huberized square hinge loss
- SVM loss has zero penalty to points well inside the margin and linear penalty to points on the wrong side

Another form of soft-margin SVC

given parameters $B \geq 0$ as a tolerance that the margin can be violated

$$\begin{array}{ll} \text{maximize} & M \\ \text{subject to} & \|w\|_2^2 = 1 \\ & y_i(x_i^T w + b) \geq M(1 - z_i), \quad i = 1, 2, \ldots, N \\ & z \succeq 0, \quad \mathbf{1}^T z \leq B \end{array}$$

with variables $w \in \mathbf{R}^n, b \in \mathbf{R}$ and $z \in \mathbf{R}^N$

- seek to make the width (M) of the margin as large as possible, while allowing some data to be on the wrong side
- z_i are slack variables that allow some data to be on the wrong side of the margin
- w is normalized to have a unit norm because the linear inequality is homogenous in w, b, M
- large B means more tolerant of margin violations, so the margin will widen

ν-SVC

problem parameters: $x_i \in \mathbf{R}^n$ and $y_i \in \{-1, 1\}$ for $i = 1, \ldots, N$, $\nu > 0$

optimization variables: $w \in \mathbf{R}^n, b \in \mathbf{R}, z \in \mathbf{R}^N$, $\rho \in \mathbf{R}$

the primal ν-SVC is

$$\begin{array}{ll} \text{minimize} & (1/2)\|w\|_2^2 - \nu\rho + \mathbf{1}^T z \\ \text{subject to} & y_i(x_i^T w + b) \geq \rho - z_i, \quad i = 1, 2, \ldots, N \\ & z \succeq 0, \quad \rho \geq 0 \end{array}$$

it can be shown that (see Chapter 9 in Schökopf page 206)

- when $z = 0$, the two classes are separated by the margin $2\rho/\|w\|_2$
- ν is an upper bound on the **fraction of margin errors:** no. of points for which $y_i(x_i^T w + b) < \rho$
- ν is a lower bound on the **fraction of support vectors**

Sparse SVC

from the soft-margin C-SVC, use $\|w\|_1$ in the objective instead

$$\text{minimize} \quad \lambda\|w\|_1 + \frac{1}{N}\sum_{i=1}^{N} \max(0, 1 - y_i(x_i^T w + b))$$

with optimization variables $w \in \mathbf{R}^n$ and $b \in \mathbf{R}$

- the ℓ_1-norm encourages sparsity of the optimal w
- for such a sparse w, the product $w^T x$ involves only a few entries in x (use less features)
- the optimization can be formulated as a linear program

SVMs: Multi-class classification

how to perform SVMs when there are $K > 2$ classes

1. **one-versus-one** classification
   - construct K-choose-2 SVMS; each of which compares a pair of classes
   - classify a test point using each of the K-choose-2 classifiers and count the number of times the test point is assigned to each class
   - assign the test point to the class that most frequently assigned in K-choose-2 classifications
2. **one-versus-all** classification
   - fit K SVMs; each time comparing one of the K classes to the remaining $K - 1$ classes
   - denote (w_k, b_k) for $k = 1, 2, \ldots, K$ the parameters of the kth SVM
   - assign a test point z to the class for which the $b_k + w_k^T z$ is largest (high level of confidence that z belongs to kth class)

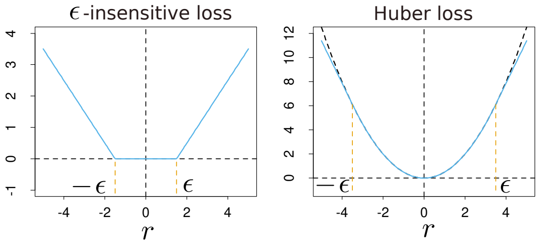
Available algorithms

- quadratic programming solvers (active-set, interior-point) on the dual
- sequential minimal optimization (SMO) on the dual
  - **MATLAB:** `fitcsvm`
  - **Python** `sklearn.svm.SVC` using `libsvm` library, which supports nonlinear classifiers)
- coordinate descent on the dual (large-scale linear SVM, used in `liblinear`)

Support Vector Regression (SVR)

ϵ-insensitive loss

ϵ-insensitive loss does not penalize errors below some $\epsilon \geq 0$



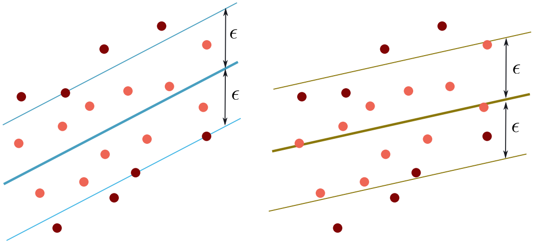
$$V_\epsilon(r) = \begin{cases} 0, & |r| < \epsilon, \\ |r| - \epsilon, & \text{otherwise} \end{cases}$$

$$V_{\text{huber}}(r) = \begin{cases} r^2/2, & |r| \leq M, \\ M|r| - M^2/2, & |r| > M, \end{cases}$$

- **Huber loss** penalizes error with linear rate when residual greater than M
- V_ϵ also has linear tails but it flattens the contributions of small residuals
- analogy to SVC: points on the correct side, and far away from it, are ignored in the optimization
- another equivalent form: $V_\epsilon(r) = \max(0, |r| - \epsilon) = (|r| - \epsilon)_+$ or just notation $|r|_\epsilon$

Flatness vs Margin

let $f(x) = w^T x + b$ be the regression model to estimate y

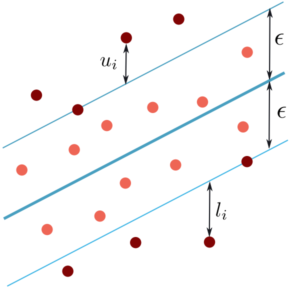


$$\underset{w,b}{\text{minimize}} \quad \sum_{i=1}^{N} V_\epsilon(y_i - f(x_i)) + \frac{\gamma}{2}\|w\|_2^2$$

- we aim to estimate y by a linear function where small residual less than ϵ is not penalized and trade off with the model complexity (measured by ℓ_2-norm)
- small $\|w\|_2^2$ corresponds to a flat linear function, but the margin is large
- the region for which $|w^T x + b| \leq \epsilon$ is an **ϵ-slab** (but sometimes called a tube)

Primal ϵ-SVR

the optimization (QP) on page 41 is equivalent to



$$\begin{array}{ll} \text{minimize} & (1/2)\|w\|_2^2 + C\sum_{i=1}^N (u_i + l_i) \\ \text{subject to} & y_i - (x_i^T w + b) \leq \epsilon + u_i, \quad i = 1, \ldots, N, \\ & x_i^T w + b - y_i \leq \epsilon + l_i, \quad i = 1, \ldots, N \\ & u \succeq 0, \quad l \succeq 0 \end{array}$$

with variables $w \in \mathbf{R}^n, b \in \mathbf{R}, u \in \mathbf{R}^N, l \in \mathbf{R}^N$

- the **primal ϵ-SVR** is similar to the concept of **soft-margin SVC**
- slack variables allow the ith residual error to exceed ϵ up to the value of u_i and l_i
- a given $C > 0$ controls the amount of slack variables (its effect is opposite to γ on page 41) – when C is large, the linear function is more flat

Derivation of the dual

the primal SVR in vector form (X contains x_i^T as rows)

$$\begin{array}{ll} \text{minimize} & (1/2)\|w\|_2^2 + C\mathbf{1}^T(u+l) \\ \text{subject to} & y - (Xw + b\mathbf{1}) - \epsilon\mathbf{1} - u \preceq 0 \\ & Xw + b\mathbf{1} - y - \epsilon\mathbf{1} - l \preceq 0 \\ & u \succeq 0, \quad l \succeq 0 \end{array}$$

let L be the Lagrangian and the Lagrange multipliers are

- $\alpha^*, \alpha \in \mathbf{R}^N$ correspond to the slab inequalities
- $\lambda^*, \lambda \in \mathbf{R}^N$ correspond to $u \succeq 0$ and $l \succeq 0$, respectively

$$\begin{aligned} L(w, b, \alpha^*, \alpha, \lambda^*, \lambda) = \frac{1}{2}\|w\|_2^2 + C\mathbf{1}^T(u+l) + \alpha^{*T}[y - Xw - b\mathbf{1} - \epsilon\mathbf{1} - u] \\ + \alpha^T[Xw + b\mathbf{1} - y - \epsilon\mathbf{1} - l] - \lambda^{*T}u - \lambda^T l \end{aligned}$$

Dual of SVR

take the infimum of L over (w, b, u, l) and use $\lambda^*, \lambda \succeq 0$ we have the conditions:

$$w = X^T(\alpha^* - \alpha), \quad \mathbf{1}^T(\alpha^* - \alpha) = 0, \quad C\mathbf{1} - \alpha^* \succeq 0, \quad C\mathbf{1} - \alpha \succeq 0$$

(from $\lambda^* = C\mathbf{1} - \alpha^*$ and $\lambda = C\mathbf{1} - \alpha$)

substitute these back to L and we have the dual function

$$g(\alpha^*, \alpha) = -(1/2)(\alpha^* - \alpha)^T XX^T(\alpha^* - \alpha) - \epsilon\mathbf{1}^T(\alpha^* + \alpha) + y^T(\alpha^* - \alpha)$$

the dual problem of SVR

$$\begin{array}{ll} \text{minimize} & (1/2)(\alpha^* - \alpha)^T XX^T(\alpha^* - \alpha) + \epsilon\mathbf{1}^T(\alpha^* + \alpha) - y^T(\alpha^* - \alpha) \\ \text{subject to} & \mathbf{1}^T(\alpha^* - \alpha) = 0, \\ & 0 \preceq \alpha^*, \alpha \preceq C\mathbf{1} \end{array}$$

with variables $\alpha^*, \alpha \in \mathbf{R}^N$

Karush-Kuhn-Tucker conditions

the estimated linear model of SVR is

$$w = \sum_{i=1}^{N} (\alpha_i^* - \alpha_i) x_i, \quad f(x) = w^T x + b = \sum_{i=1}^{N} (\alpha_i^* - \alpha_i) \langle x_i, x_i \rangle + b$$

the complementary slackness conditions are

$$\begin{aligned} \alpha_i^*(y_i - x_i^T w - b - \epsilon - u_i) &= 0, \quad u_i(C - \alpha_i^*) = 0 \\ \alpha_i(x_i^T w + b - y_i - \epsilon - l_i) &= 0, \quad l_i(C - \alpha_i) = 0 \end{aligned}$$

important conclusions:

- if $u_i > 0$, then $\alpha_i^* = C$; only data (x_i, y_i) with $\alpha_i^* = C$ can lie outside the slab
- if $|y_i - (x_i^T w + b)| \leq \epsilon$ then $\alpha_i^*, \alpha_i = 0$ ✎ we need only **support vectors** to compute w – those with nonzero coefficients
- if $0 < \alpha_i^* < C$ then $u_i = 0$ OR if $0 < \alpha_i < C$ then $l_i = 0$; we can compute b

$$b = y_i - x_i^T w - \epsilon, \quad \textbf{OR} \quad b = y_i - x_i^T w + \epsilon$$

Nonlinear SVR

obtain by replacing the dot product with a nonlinear kernel function

$$f(x) = \sum_{i=1}^{N} (\alpha_i^* - \alpha_i) K(x_i, x) + b$$

which is solved from the dual for nonlinear SVR when XX^T is replaced by $K(x_i, x_j)$

$$XX^T = \begin{bmatrix} x_1^T x_1 & \cdots & x_1^T x_N \\ \vdots & \ddots & \vdots \\ x_N^T x_1 & \cdots & x_N^T x_N \end{bmatrix} \Rightarrow G = \begin{bmatrix} K(x_1, x_1) & K(x_1, x_2) & \cdots & K(x_1, x_N) \\ K(x_2, x_1) & K(x_2, x_2) & \cdots & K(x_2, x_N) \\ \vdots & \vdots & \ddots & \vdots \\ K(x_N, x_1) & K(x_N, x_2) & \cdots & K(x_N, x_N) \end{bmatrix}$$

choice of kernel functions: polynomial, radial basis kernels

Softwares for SVR

- **MATLAB:** `fitrsvm`
- **Python** `sklearn.svm.SVR` using `libsvm` library)

References

some figures and examples are taken from the first two references (ISLR, ESL)

1. Chapter 7 and 9 in B. Schökopf and A. J. Smola, *Learning with Kernels: Support vector machines, regularization, optimization, and Beyond*, The MIT Press, 2002
2. Chapter 12 in T. Hastie, R. Tibshirani and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, Springer, Second edition, 2009
3. Chapter 9 in G. James, D. Witten, T. Hastie, and R. Tibshirani, *An Introduction to Statistical Learning: with Application in R*, Springer, 2021
4. R.Fan, K.Chang, C.Hsieh, X.Wang and C.Lin, *LIBBLINEAR: A Library for large linear classification*, JMLR, 2008, `https://www.csie.ntu.edu.tw/~cjlin/papers/liblinear.pdf`