Statistics 101C - Support Vector Machine

Shirong Xu

University of California, Los Angeles

shirong@stat.ucla.edu

December 1, 2024

Review of vector algebra

- We have two hyperplanes:
  - $L_1 : \boldsymbol{w}^T \boldsymbol{x} + b_0 = 0$
  - $L_2 : \boldsymbol{w}^T \boldsymbol{x} + b_1 = 0$

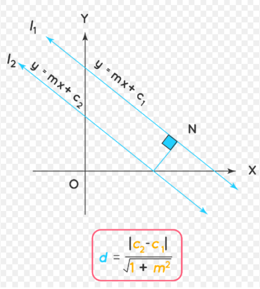
Review of vector algebra

- We have two hyperplanes:
  - $L_1 : \boldsymbol{w}^T\boldsymbol{x} + b_0 = 0$
  - $L_2 : \boldsymbol{w}^T\boldsymbol{x} + b_1 = 0$
- What is the distance between these two hyperplanes?

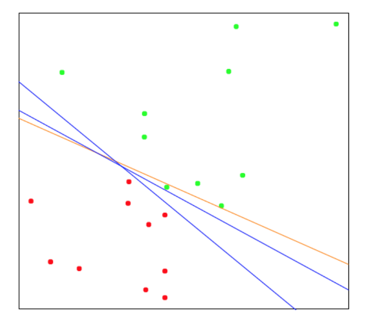
Review of vector algebra

- We have two hyperplanes:
  - $L_1 : \boldsymbol{w}^T\boldsymbol{x} + b_0 = 0$
  - $L_2 : \boldsymbol{w}^T\boldsymbol{x} + b_1 = 0$
- What is the distance between these two hyperplanes?

$$D(L_1, L_2) = \frac{|b_0 - b_1|}{\sqrt{\sum_{i=1}^{p} w_i^2}}$$



An illustrative plot: Binary Classification



- Points with different labels are **separable**.
- We can find **infinite** hyperplanes to separate two classes.
- **Question**: How can we choose the most appropriate one?

Preliminaries to Support Vector Machine

- Training data: $(\mathbf{x}_i, y_i)$; $i = 1, \ldots, n$ with $\mathbf{x}_i \in \mathcal{R}^p$ and $y_i \in \{-1, 1\}$

Preliminaries to Support Vector Machine

- Training data: $(\mathbf{x}_i, y_i)$; $i = 1, \ldots, n$ with $\mathbf{x}_i \in \mathcal{R}^p$ and $y_i \in \{-1, 1\}$
- Define a separating hyperplane

$$\{\mathbf{x} : f(\mathbf{x}) = w^T \mathbf{x} + b = 0\}$$

Preliminaries to Support Vector Machine

- Training data: $(\mathbf{x}_i, y_i)$; $i = 1, \dots, n$ with $\mathbf{x}_i \in \mathcal{R}^p$ and $y_i \in \{-1, 1\}$
- Define a separating hyperplane

$$\{\mathbf{x} : f(\mathbf{x}) = w^T \mathbf{x} + b = 0\}$$

- A classification decision function is: **Classifier**

$$G(\mathbf{x}) = \mathrm{sign}(f(\mathbf{x})) = \mathrm{sign}(w^T \mathbf{x} + b)$$

Preliminaries to Support Vector Machine

- Training data: $(\mathbf{x}_i, y_i)$; $i = 1, \ldots, n$ with $\mathbf{x}_i \in \mathcal{R}^p$ and $y_i \in \{-1, 1\}$
- Define a separating hyperplane

$$\{\mathbf{x} : f(\mathbf{x}) = w^T \mathbf{x} + b = 0\}$$

- A classification decision function is: **Classifier**

$$G(\mathbf{x}) = \mathrm{sign}(f(\mathbf{x})) = \mathrm{sign}(w^T \mathbf{x} + b)$$

- A misclassified point $(\mathbf{x}, y) \Leftrightarrow yf(\mathbf{x}) < 0$, and vice versa, where $yf(\mathbf{x})$ is called functional margin. For example, the functional margin of sample $(\boldsymbol{x}_i, y_i)$ is $y_i f(\boldsymbol{x}_i)$.

Preliminaries to Support Vector Machine

- Training data: $(\mathbf{x}_i, y_i)$; $i = 1, \ldots, n$ with $\mathbf{x}_i \in \mathcal{R}^p$ and $y_i \in \{-1, 1\}$
- Define a separating hyperplane

$$\{\mathbf{x} : f(\mathbf{x}) = w^T \mathbf{x} + b = 0\}$$

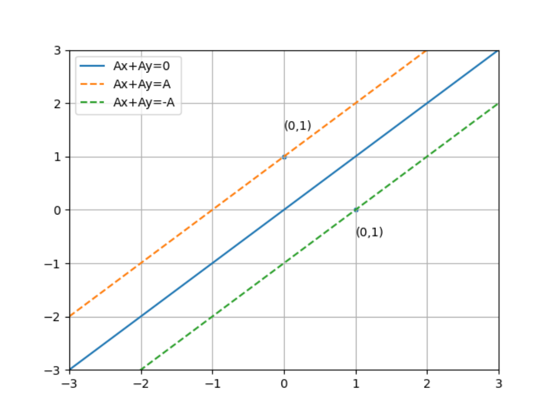
- A classification decision function is: **Classifier**

$$G(\mathbf{x}) = \mathrm{sign}(f(\mathbf{x})) = \mathrm{sign}(w^T \mathbf{x} + b)$$

- A misclassified point $(\mathbf{x}, y) \Leftrightarrow yf(\mathbf{x}) < 0$, and vice versa, where $yf(\mathbf{x})$ is called functional margin. For example, the functional margin of sample $(\boldsymbol{x}_i, y_i)$ is $y_i f(\boldsymbol{x}_i)$.
- Example: If the hyperplane is $f(x) = 3x + 1 = 0$. Then the functional margin of $(x, y) = (2, 1)$ is $1 * (3 * 2 + 1) = 7$.

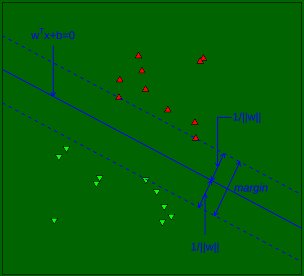
Example

- The following hyperplane separates two points (0,1) and (1,0): $Ax + Ay = 0, A > 0$.



- The distance from (0,1) to $Ax + Ay = 0$ is $1/\sqrt{2}$. If we set two lines as $Ax + Ay = 1$ and $Ax + Ay = -1$, then $A = 1$.

Illustrative plot: separable



A natural result is to find a hyperplane in the middle. Find the closest points on the following two hyperplanes:

- $\boldsymbol{w}^T\boldsymbol{x} + b = 1$ and $\boldsymbol{w}^T\boldsymbol{x} + b = -1$

Separable case

- The optimization problem can be rephrased as

$$\begin{aligned} \min_{w,b} \quad & \|w\| \text{ or } \frac{1}{2}\|w\|^2 \\ \text{subject to} \quad & y_i(w^T\mathbf{x}_i + b) \geq 1; \; i = 1, \ldots, n \end{aligned}$$

- The two dashed hyperplanes are set as $w^T \mathbf{x} + b = \pm 1$
- **Idea**: Finding the hyperplane maximizing the closet distances to two classes.

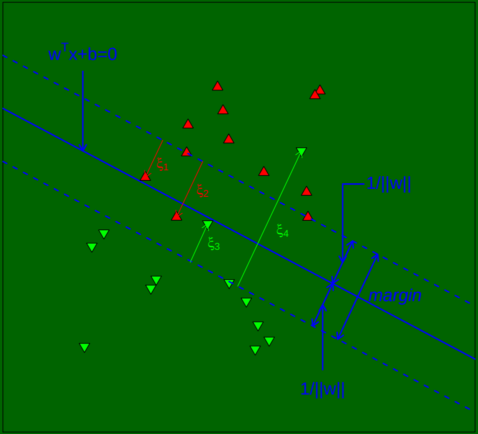
Separable case

- The optimization problem can be rephrased as

$$\begin{aligned} \min_{w,b} \quad & \|w\| \text{ or } \frac{1}{2}\|w\|^2 \\ \text{subject to} \quad & y_i(w^T \mathbf{x}_i + b) \geq 1;\ i = 1, \ldots, n \end{aligned}$$

- The two dashed hyperplanes are set as $w^T \mathbf{x} + b = \pm 1$
- **Idea**: Finding the hyperplane maximizing the closet distances to two classes.
- **Question**: What if the classes are non-separable?

Illustrative plot: nonseparable



SVM formulation

- When the data are not separable, the constraints need to be relaxed by introducing some slack variables ξ_i
- A standard SVM formulation:

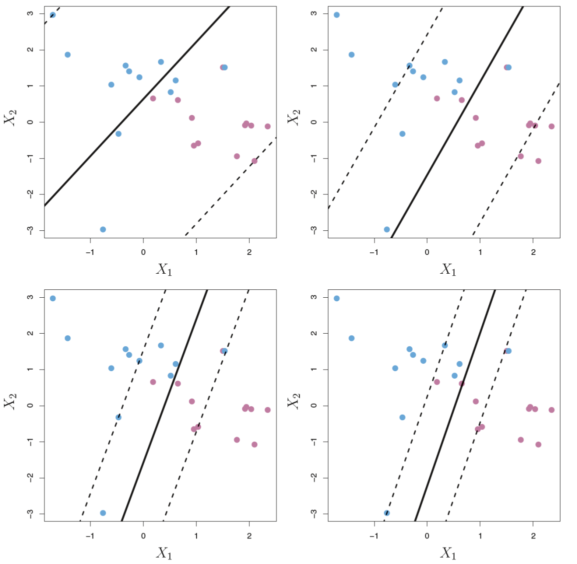
$$\begin{aligned} \min_{w,b} \quad & \frac{1}{2}\|w\|^2 \\ \text{subject to} \quad & y_i(w^T \mathbf{x}_i + b) \geq 1 - \xi_i;\ \ i = 1, \ldots, n, \\ & \xi_i \geq 0,\ \sum_{i=1}^{n} \xi_i \leq C \end{aligned}$$

- C controls the size of slack variables, or the severity of the misclassified observations

Some remarks

- The value of ξ_i tells if the i-th observation is on the wrong side of the separation hyperplane
- Different values of C lead to different hyperplanes
  - If $C = 0$, all ξ_i must be 0, and thus all observations have to be correctly classified
  - As C increases, it is more tolerant of misclassification, and so the margin will widen
- Optimal C can be determined by cross validation

Example with linear SVM



Derivation From surrogate loss: hinge loss

Let $D = \{\boldsymbol{x}_i, y_i\}_{i=1}^n$ be the dataset, and we use Hinge loss as surrogate loss

$$\min_{\boldsymbol{w},b} \frac{1}{n}\sum_{i=1}^{n} \max\left\{1-(\boldsymbol{w}^T\boldsymbol{x}_i+b)y_i, 0\right\} + \lambda\|\boldsymbol{w}\|_2^2 \tag{1}$$

Derivation From surrogate loss: hinge loss

Let $D = \{\boldsymbol{x}_i, y_i\}_{i=1}^n$ be the dataset, and we use Hinge loss as surrogate loss

$$\min_{\boldsymbol{w},b} \frac{1}{n}\sum_{i=1}^{n} \max\left\{1-(\boldsymbol{w}^T\boldsymbol{x}_i+b)y_i, 0\right\} + \lambda\|\boldsymbol{w}\|_2^2 \tag{1}$$

We notice that $\max\left\{1-(\boldsymbol{w}^T\boldsymbol{x}_i+b)y_i, 0\right\}$ is positive, so we denote $\xi_i = \max\left\{1-(\boldsymbol{w}^T\boldsymbol{x}_i+b)y_i, 0\right\} \geq 0$. Then (1) can be **equivalently** written as

Derivation From surrogate loss: hinge loss

Let $D = \{\boldsymbol{x}_i, y_i\}_{i=1}^n$ be the dataset, and we use Hinge loss as surrogate loss

$$\min_{\boldsymbol{w},b} \frac{1}{n}\sum_{i=1}^{n} \max\left\{1-(\boldsymbol{w}^T\boldsymbol{x}_i+b)y_i, 0\right\} + \lambda\|\boldsymbol{w}\|_2^2 \tag{1}$$

We notice that $\max\left\{1-(\boldsymbol{w}^T\boldsymbol{x}_i+b)y_i, 0\right\}$ is positive, so we denote $\xi_i = \max\left\{1-(\boldsymbol{w}^T\boldsymbol{x}_i+b)y_i, 0\right\} \geq 0$. Then (1) can be **equivalently** written as

$$\begin{aligned} &\min_{\boldsymbol{w},b} \frac{1}{n}\sum_{i=1}^{n} \xi_i + \lambda\|\boldsymbol{w}\|_2^2 \\ &\text{subject to } 1-(\boldsymbol{w}^T\boldsymbol{x}_i+b)y_i \leq \xi_i, \\ &\xi_i \geq 0, i = 1, \ldots, n \end{aligned}$$

Derivation From surrogate loss: hinge loss

$$\min_{\boldsymbol{w},b} \frac{1}{n}\sum_{i=1}^{n}\xi_i + \lambda\|\boldsymbol{w}\|_2^2$$

$$\text{subject to } 1-(\boldsymbol{w}^T\boldsymbol{x}_i+b)y_i \leq \xi_i, \quad \xi_i \geq 0, i=1,\ldots,n$$

We know there exists a constant C_1 such that the above optimization problem is equivalent to

$$\min_{\boldsymbol{w},b} \quad \|\boldsymbol{w}\|_2^2$$

$$\text{subject to } (\boldsymbol{w}^T\boldsymbol{x}_i+b)y_i \geq 1-\xi_i,$$

$$\xi_i \geq 0, i=1,\ldots,n, \quad \frac{1}{n\lambda}\sum_{i=1}^{n}\xi_i \leq C_1$$

Derivation From surrogate loss: hinge loss

$$\min_{\boldsymbol{w},b} \frac{1}{n}\sum_{i=1}^{n}\xi_i + \lambda\|\boldsymbol{w}\|_2^2$$

$$\text{subject to } 1-(\boldsymbol{w}^T\boldsymbol{x}_i+b)y_i \le \xi_i, \quad \xi_i \ge 0, i=1,\ldots,n$$

We know there exists a constant C_1 such that the above optimization problem is equivalent to

$$\min_{\boldsymbol{w},b} \quad \|\boldsymbol{w}\|_2^2$$

$$\text{subject to } (\boldsymbol{w}^T\boldsymbol{x}_i+b)y_i \ge 1-\xi_i,$$

$$\xi_i \ge 0, i=1,\ldots,n, \quad \frac{1}{n\lambda}\sum_{i=1}^{n}\xi_i \le C_1$$

Conclusion: Linear SVM is using **hinge loss** as surrogate loss function with L-2 penalty term with linear function.

How can we solve linear SVM?

Primal - Dual Optimization problem

- For a minimization problem (Primal Optimization Problem):

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & f(\boldsymbol{x}) \\ \text{subject to} \quad & h_i(\boldsymbol{x}) \leq 0, i = 1, \ldots, m \\ & l_j(\boldsymbol{x}) = 0, j = 1, \ldots, r \end{aligned}$$

- The Lagrangian is defined as

$$L(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{v}) = f(\boldsymbol{x}) + \sum_{i=1}^{m} u_i h_i(\boldsymbol{x}) + \sum_{j=1}^{r} v_j l_j(\boldsymbol{x}).$$

- $L(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{v})$ is a lower bound $f(\boldsymbol{x})$.

Primal - Dual Optimization problem

- The Lagrange dual function is (Fining $\boldsymbol{x}$ minimizing the Lagrangian)

$$g(\boldsymbol{u}, \boldsymbol{v}) = \min_{\boldsymbol{x}} L(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{v})$$

- **The dual problem** is defined as

$$\begin{aligned} \max_{\boldsymbol{u}, \boldsymbol{v}} \quad & g(\boldsymbol{u}, \boldsymbol{v}) \\ \text{subject to} \quad & \boldsymbol{u} \geq 0 \end{aligned}$$

- The Lagrange dual function can be viewd as a pointwise maximization of some affine functions so it is always concave. The dual problem is always convex even if the primal problem is not convex.

Primal - Dual Optimization problem

- For any primal problem and dual problem, the weak duality always holds

$$g^* \leq f^*$$

where g^* and f^* are optimal values for **dual** and **primal** problems, respectively.

- strong duality: $f^* = g^*$.
- The dual problem sometime can be easier to solve compared with the primal problem and the primal solution can be constructed from the dual solution.

KKT condition

Stationarity: $0 \in \partial f(x) + \sum_{i=1}^{m} u_i \partial h_i(x) + \sum_{j=1}^{r} v_j \partial \ell_j(x)$

Complementary: $u_i h_i(x) = 0$ for all i

Primal feasibility: $h_i(x) \leq 0$, $\ell_j(x) = 0$ for all i, j

Dual feasibility: $u_i \geq 0$ for all i

- If you have found a point that satisfies the KKT conditions, then it is an optimal solution. It will be the unique optimal solution when f is convex.

Karush-Kuhn-Tucker (KKT) conditions form the backbone of linear and nonlinear programming as they are

- Necessary and sufficient for optimality in linear programming.
- Necessary and sufficient for optimality in convex optimization, such as least square minimization in linear regression.
- Necessary for optimality in non-convex optimization problem, such as deep learning model training.

Example

Primal Problem

$$\min_{\boldsymbol{x}} \quad x^2$$
$$\text{subject to } x - 1 \leq 0$$

The Lagrangian is given as

$$L(x,u) = x^2 + u(x-1).$$

The Lagrangian dual is given as

$$g(u) = L(-u/2, u) = -u^2/4 - u$$

The maximum of $g(u)$ is equal to the minimum of $f(x)$: $f^* = g^* = 0$

Optimization in SVM

- First the SVM formulation is equivalent to

$$
\begin{aligned}
\min_{w,b} \quad & \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi_i \\
\text{subject to} \quad & y_i(w^T\mathbf{x}_i + b) \geq 1 - \xi_i;\ \xi_i \geq 0;\ i = 1, \ldots, n
\end{aligned}
$$

- Lagrange (primal) function is

$$
\begin{aligned}
L_P =& \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi_i \\
& - \sum_{i=1}^{n}\alpha_i(y_i(w^T\mathbf{x}_i + b) - 1 + \xi_i) - \sum_{i=1}^{n}\beta_i\xi_i,
\end{aligned}
$$

which we minimize w.r.t. w, b and ξ_i, subject to $\xi_i, \alpha_i, \beta_i \geq 0$

Dual form

- Setting the derivatives to zero,

$$w = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i, \; 0 = \sum_{i=1}^{n} \alpha_i y_i, \; C = \alpha_i + \beta_i,$$

as well as $\xi_i, \alpha_i, \beta_i \geq 0$

- Substituting them back to the primal function yields the Wolfe dual function,

$$\max_{\alpha} \qquad L_D = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{i'=1}^{n} \alpha_i \alpha_{i'} y_i y_{i'} \langle \mathbf{x}_i, \mathbf{x}_{i'} \rangle$$

$$\text{subject to} \qquad \sum_{i=1}^{n} \alpha_i y_i = 0; \; 0 \leq \alpha_i \leq C$$

KKT conditions

- The solutions to the primal and the dual forms are equivalent,

$$\hat{w} = \sum_{i=1}^{n} \hat{\alpha}_i y_i \mathbf{x}_i$$

- The solutions to the primal and the dual forms must satisfy the Karush-Kuhn-Tucker conditions:

$$\begin{aligned} \alpha_i(y_i(w^T \mathbf{x}_i + b) - 1 + \xi_i) &= 0, \\ \beta_i \xi_i &= 0, \\ y_i(w^T \mathbf{x}_i + b) &\geq 1 - \xi_i \end{aligned}$$

More on KKT conditions

- $\hat{\alpha}_i > 0$ only when $y_i(\hat{w}^T \mathbf{x}_i + \hat{b}) = 1 - \hat{\xi}_i$, or equivalently,

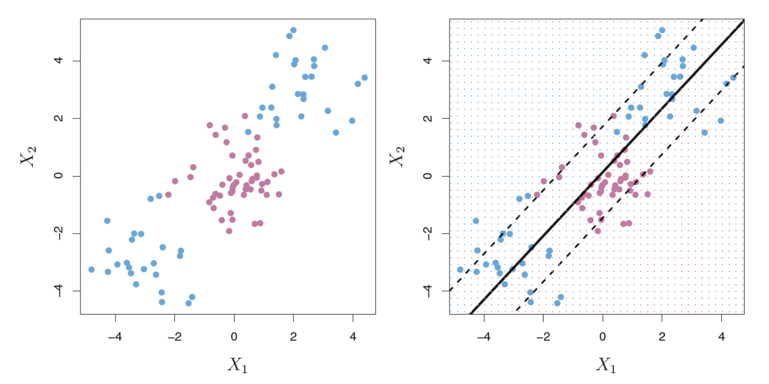
  $$y_i(\hat{w}^T \mathbf{x}_i + \hat{b}) \leq 1.$$

  These points are called support vectors
- Among support vectors, some have $\hat{\xi}_i > 0$ and $\hat{\alpha}_i = C$; and others lie on the margin $\hat{\xi}_i = 0$, and thus $0 < \hat{\alpha}_i \leq C$
- b cannot be obtained from the dual form, but can be solved by using any point with $0 < \hat{\alpha}_i < C$, where

  $$y_i(\hat{w}^T \mathbf{x}_i + b) = 1$$

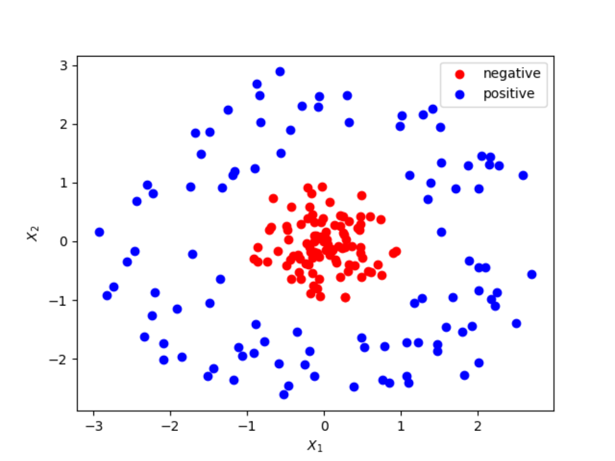
- If no point with $0 < \hat{\alpha}_i < C$, plugging $\hat{w}$ back into the primal form and solve for $\hat{b}$

A toy example



When the classes have nonlinear boundary, linear SVM can perform very poorly

Nonlinear Decision Boundary



If we fit a SVM on this example, the accuracy will be bad!

Nonlinear SVM with kernels

The key idea of extending linear SVM, and many other linear procedures, to nonlinear is to:

- Enlarge the predictor space using basis expansion functions $h_1(\mathbf{x}), \ldots, h_M(\mathbf{x})$
- Construct a linear separating hyperplane $f(x) = \mathbf{w}^T h(\mathbf{x}) + b$ in the enlarged space for better training performance
- The linear separating hyperplane in the enlarged space can be translated into a nonlinear separating hyperplane in the original space

- In the enlarged space, the dual form becomes

$$\max_{\alpha} \quad L_D = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{i'=1}^{n} \alpha_i \alpha_{i'} y_i y_{i'} \langle h(\mathbf{x}_i), h(\mathbf{x}_{i'}) \rangle$$

$$\text{subject to} \quad \sum_{i=1}^{n} \alpha_i y_i = 0;\ 0 \le \alpha_i \le C$$

- The solution is $\hat{\mathbf{w}} = \sum_{i=1}^{n} \hat{\alpha}_i y_i h(\mathbf{x}_i)$, and

$$\hat{f}(\mathbf{x}) = \hat{\mathbf{w}}^T h(\mathbf{x}) + \hat{b} = \sum_{i=1}^{n} \hat{\alpha}_i y_i \langle h(\mathbf{x}_i), h(\mathbf{x}) \rangle + \hat{b}$$

- The interesting part is the formulation relies on $h(\mathbf{x})$ only through their inner products

Kernels

- Define
$$K(\mathbf{x}, \mathbf{x}') = \langle h(\mathbf{x}), h(\mathbf{x}')\rangle,$$
and thus we need not specify $h(\cdot)$ at all, but only $K(\cdot, \cdot)$
- Popular choice of $K(\cdot, \cdot)$:
  - Linear: $K(\mathbf{x}, \mathbf{x}') = \mathbf{x}^T \mathbf{x}'$
  - Degree-d polynomial: $K(\mathbf{x}, \mathbf{x}') = (1 + \mathbf{x}^T \mathbf{x}')^d$
  - Radial (Gaussian): $K(\mathbf{x}, \mathbf{x}') = \exp(-\frac{\| \mathbf{x} - \mathbf{x}' \|^2}{\sigma^2})$

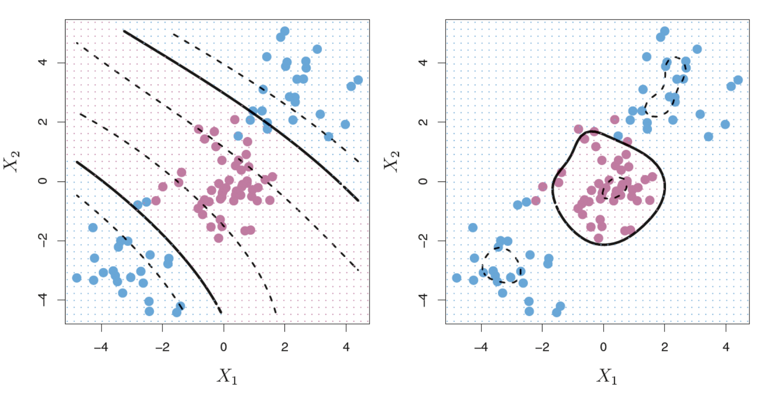
Why a Kernel is associated with a mapping

- Consider a polynomial-2 kernel

$$
\begin{aligned}
K(x, y) =& (1 + xy)^2 = (1 + 2xy + x^2y^2) \\
=& (1, \sqrt{2}x, x^2) \cdot (1, \sqrt{2}y, y^2) \\
=& \langle h(x), h(y) \rangle .
\end{aligned}
$$

- Using Kernel function $K(\cdot, \cdot)$ in SVM is equivalent to
- Finding a non-linear transformation $h(\boldsymbol{x})$ on $\boldsymbol{x}$
- Fit a linear SVM with respect to $h(\boldsymbol{x})$

Example with nonlinear SVM

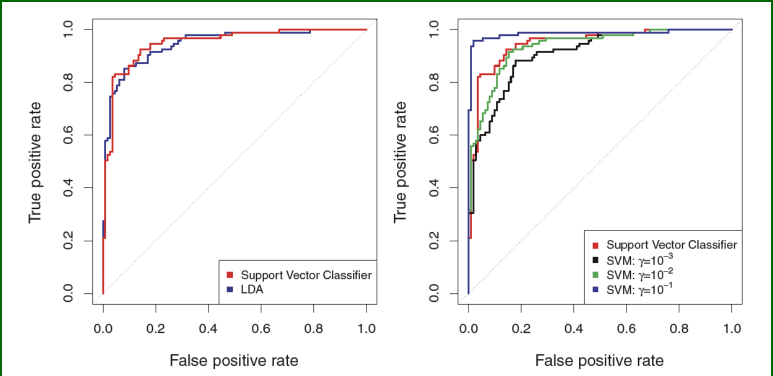


Left: SVM with a polynomial kernel of degree 3; Right: SVM with a radial kernel

Radial kernel

- It is also known as the Gaussian kernel
- When $\mathbf{x}_i$ is far away from $\mathbf{x}$, $K(\mathbf{x}_i, \mathbf{x})$ will be very tiny and thus $\mathbf{x}_i$ has little effect on $\hat{f}(\mathbf{x})$
- The radial kernel has very local behavior, and only nearby training observations have effects on the prediction of test observations
- The corresponding feature space of the radial kernel is implicit and infinite-dimensional

Heart example



Multiclass SVM

When the response $y \in \{1, \ldots, K\}$ with $K > 2$,

- One-versus-one approach
  - Construct C_K^2 binary SVM classifiers, each compares one pair of classes
  - Assign the test observation to the class to which it was most frequently assigned in these C_K^2 pairwise classification
- One-versus-rest approach
  - Construct K binary SVM classifiers, each compares one class to the rest $K - 1$ classes
  - Assign the test observation to the class for which the predicted function value is the largest
- Unified approach is also available, but with more sophisticated loss functions

SVM in a regularization form

- A generic regularization form

$$\min_{f \in \mathcal{F}} \sum_{i=1}^{n} L(y_i, f(\mathbf{x}_i)) + \lambda J(f)$$

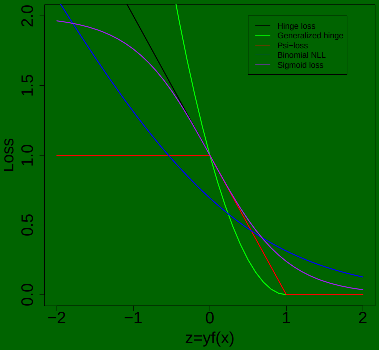
- The SVM formulation can be rephrased as

$$\min_{f \in \mathcal{F}} \sum_{i=1}^{n} (1 - y_i f(\mathbf{x}_i))_+ + \frac{\lambda}{2} \|f\|_{\mathcal{F}}^2,$$

where $\mathcal{F}$ is the candidate functional space, and $\| \cdot \|_{\mathcal{F}}$ is the norm associated with $\mathcal{F}$

- $L(z) = (1 - z)_+$ is the hinge loss, where $z = yf(x)$ is the functional margin

Large margin loss



- Hinge loss:
  $L(z) = (1 - z)_+ = \max(1 - z, 0)$
- Generalized hinge loss:
  $L(z) = (1 - z)_+^q$
- ψ-loss:
  $L(z) = \psi(z) = \min(1, (1 - z)_+)$
- Binomial NLL:
  $L(z) = \log(1 + e^{-z})$
- Sigmoid loss:
  $L(z) = 1 - \tanh(\lambda z)$