

Support Vector Machines

ML Instruction Team, Fall 2022

CE Department
Sharif University of Technology

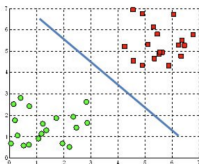
Ali Sharifi-Zarchi
Behrooz Azarkhalili
Alireza Gargoori Motlagh

Hyperplane

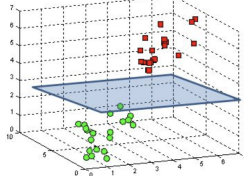
- **Hyperplane:** A hyperplane in p dimensions is an affine subspace of dimension $p-1$:

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_p X_p = 0$$

A hyperplane in \mathbb{R}^2 is a line



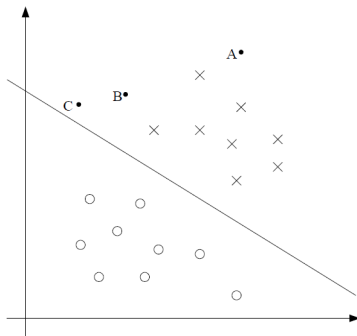
A hyperplane in \mathbb{R}^3 is a plane



- The vector $\beta = (\beta_1, \beta_2, \dots, \beta_p)$ is called the normal vector – it points in a direction orthogonal to the surface of the defined hyperplane.
- If $f(X) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_p X_p = \beta^T X + \beta_0$, then $f(X)$ divides the p -dimensional feature space into two half-spaces ($f(X) > 0$ for one side and $f(X) < 0$ for the other side).
- So if we code $Y^{(i)} \in \{\pm 1\}$, then $\forall i : Y^{(i)} f(X^{(i)}) > 0$

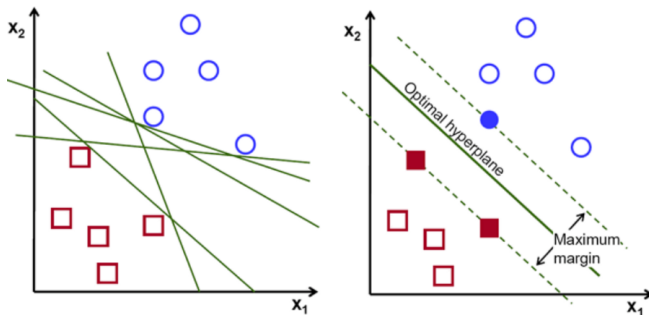
Intuition: Margins

■ Separating Hyperplane



- Our confidence about the prediction of classes of A, B and C relies on their distance from decision boundary.
- We try to find the optimal hyperplane that separates the classes in the feature space.

Maximal Margin Classifier



- **Maximal (Optimal) Separating Hyperplane:** The separating hyperplane with the largest margin between the classes.

$$\begin{aligned}
 & \max_{\beta_0, \beta, M} \quad M \\
 & \text{s.t.} \quad \sum_{j=1}^p \beta_j^2 = 1 \\
 & \quad y^{(i)}(\beta^T x^{(i)} + \beta_0) \geq M \quad \forall i \in \{1, 2, \dots, N\}
 \end{aligned} \tag{1}$$

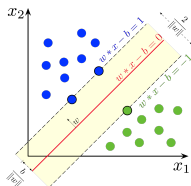
Maximal Margin Classifier: Quadratic Program

- Eq.(1) can be rephrased as a convex quadratic problem and be solved efficiently using QP solvers.
- (Euclidean) distance between two hyperplanes

$$\mathcal{H}_1 = \{x | \beta^T x + \beta_0 = 1\} \quad \mathcal{H}_2 = \{x | \beta^T x + \beta_0 = -1\}$$

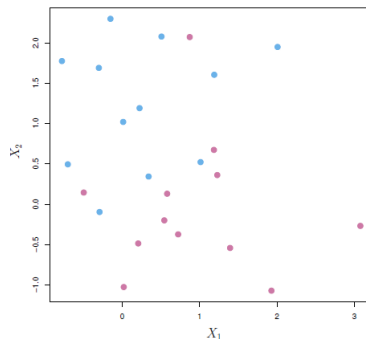
is $\text{dist}(\mathcal{H}_1, \mathcal{H}_2) = 2/\|\beta\|_2$

$$\begin{aligned} \min_{\beta, \beta_0} \quad & \frac{1}{2} \|\beta\|_2^2 \\ \text{s.t.} \quad & y^{(i)}(\beta^T x^{(i)} + \beta_0) \geq 1 \quad \forall i \in 1, 2, \dots, N \end{aligned} \quad (2)$$



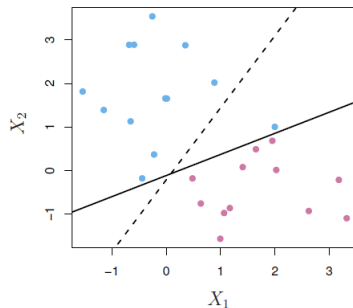
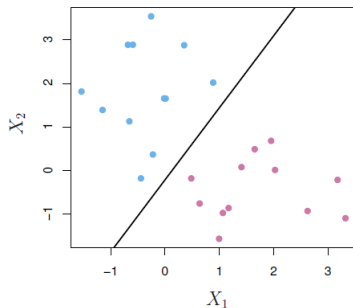
Non-linearly Separable Data

- In most cases however, the data are not linearly separable unless $N < p$.



Noisy Data

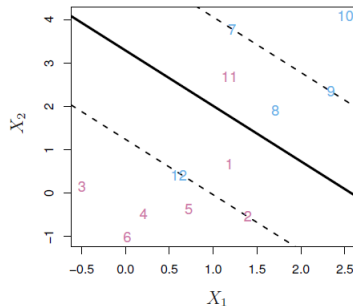
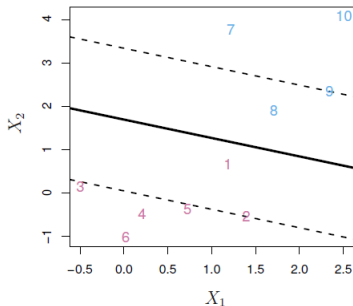
- Sometimes the data are linearly separable, but noisy. This can lead to a poor solution for the maximal margin classifier. Also, hard-margin classifier is sensitive to outliers.



- The *support vector classifier* (SVC) maximizes a *soft* margin.

Support Vector Classifier(Soft Margin Classifier)

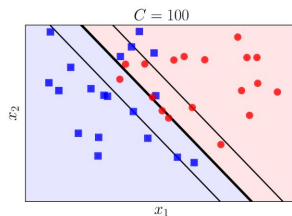
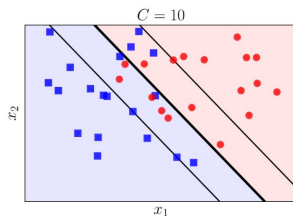
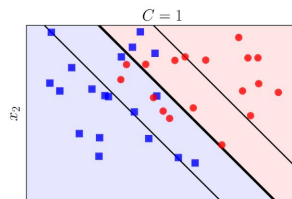
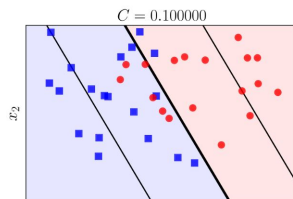
- Allowing some samples to violate the margin, with *slack variables* (ξ), in a controlled manner:



$$\begin{aligned}
 & \min_{\beta, \beta_0, \xi} \quad \frac{1}{2} \|\beta\|_2^2 + C \sum_{i=1}^N \xi_i \\
 & \text{s.t.} \quad y^{(i)}(\beta^T x^{(i)} + \beta_0) \geq 1 - \xi_i \\
 & \quad \quad \xi_i \geq 0 \quad \forall i \in \{1, 2, \dots, N\}
 \end{aligned} \tag{3}$$

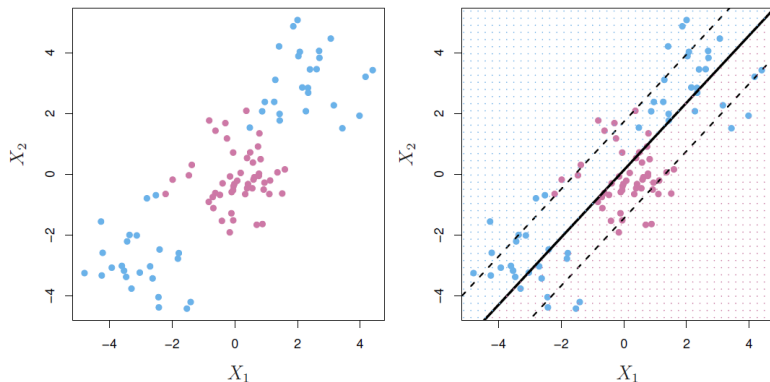
Effect of Regularization Parameter

- C is a regularization parameter that controls the bias-variance trade-off of the support vector classifier.



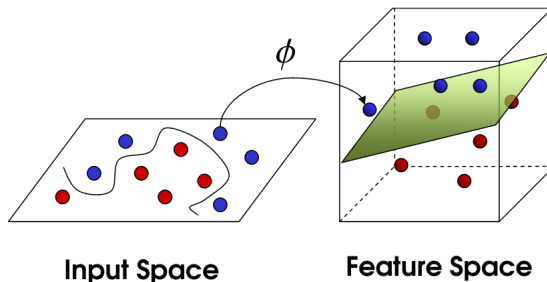
The Need for Non-Linear Boundary

- Linear boundary can fail in many cases, regardless of the value of C .



Feature Expansion

- Enlarge the space of features by including transformations; e.g. $X_1^2, X_1^3, X_1 X_2, X_1 X_2^2, \dots$. Hence increasing the dimension of the original p -dimensional input space.
- Then we can fit a support vector classifier in the enlarged space.
- This results in non-linear decision boundaries in the original feature space.



Dual Problem of SVC

■ Primal problem:

$$\begin{aligned} \min_{\beta, \beta_0, \xi} \quad & \frac{1}{2} \|\beta\|_2^2 + C \sum_{i=1}^N \xi_i \\ \text{s.t.} \quad & y^{(i)}(\beta^T x^{(i)} + \beta_0) \geq 1 - \xi_i \\ & \xi_i \geq 0 \quad \forall i \in \{1, 2, \dots, N\} \end{aligned}$$

■ Dual problem (using the lagrangian):

$$\begin{aligned} \max_{\alpha_i} \quad & \sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y^{(i)} y^{(j)} x^{(i)T} x^{(j)} \\ \text{s.t.} \quad & \sum_{i=1}^N \alpha_i y^{(i)} = 0 \\ & 0 \leq \alpha_i \leq C \quad \forall i \in \{1, 2, \dots, N\} \end{aligned} \tag{4}$$

KKT Conditions

- From Karush-Kuhn-Tucker (KKT) conditions and complementary slackness we have:

$$\begin{aligned}\alpha_i(1 - \xi_i - y^{(i)}(\beta^T x^{(i)} + \beta_0)) &= 0 \\ (C - \alpha_i)\xi_i &= 0\end{aligned}$$

- There can be 3 cases:

$$\left\{ \begin{array}{ll} \alpha_i = 0 \rightarrow \xi_i = 0 \Rightarrow 1 - y^{(i)}(\beta^T x^{(i)} + \beta_0) < 0: & \text{Non-Support} \\ 0 < \alpha_i < C \rightarrow \xi_i = 0 \Rightarrow 1 - y^{(i)}(\beta^T x^{(i)} + \beta_0) = 0: & \text{Support} \\ \alpha_i = C \rightarrow \xi_i > 0 \Rightarrow 1 - \xi_i - y^{(i)}(\beta^T x^{(i)} + \beta_0) = 0: & \text{Support} \end{array} \right.$$

Classification Function

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

- Also, when minimizing the conjugate function of primal problem, we get:

$$\beta = \sum_{i=1}^N \alpha_i y^{(i)} x^{(i)}$$

- This results in the following classification function for SVC:

$$f(x) = \beta_0 + \sum_{i=1}^N \alpha_i y^{(i)} x^T x^{(i)} = \beta_0 + \sum_{i=1}^N \alpha_i y^{(i)} \langle x, x^{(i)} \rangle \quad (5)$$

- It turns out that most of the $\hat{\alpha}_i$ can be zero:

$$f(x) = \beta_0 + \sum_{i \in \mathcal{S}} \hat{\alpha}_i y^{(i)} \langle x, x^{(i)} \rangle \quad (6)$$

where \mathcal{S} is the *support set* of indices i such that $\hat{\alpha}_i > 0$.

So, all we need is *inner products*!

Kernels and SVMs

- Consider the feature mapping $T : x \rightarrow \phi(x)$, we define the corresponding *kernel* to be

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

Now we can simply replace all inner products by $K(x, x')$, and our algorithm would now be learning using the features ϕ .

- Kernel matrix: $K_{i,j} = K(x^{(i)}, x^{(j)})$

A valid kernel function is the one that results in a **symmetric positive semi-definite** kernel matrix for any set of samples. (*Mercer theorem*: necessary and sufficient condition)

- Calculating kernel matrix may be very inexpensive, even though $\phi(x)$ itself may be very expensive to calculate(perhaps because it is an extremely high dimensional vector).
- The classification function solution has the form

$$f(x) = \beta_0 + \sum_{i \in \mathcal{S}} \hat{\alpha}_i y^{(i)} K(x, x^{(i)}) \quad (7)$$

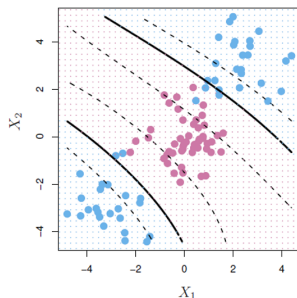
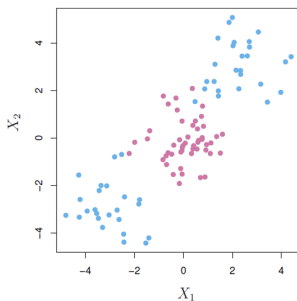
Polynomial Kernels

$$K(x, y) = (1 + \langle x, y \rangle)^d \quad (8)$$

- e.g. Feature transformation from 2D kernels:

$$K(x, y) = (1 + \langle x, y \rangle)^2 \Rightarrow \phi(x) = (1, \sqrt{2}x_1, \sqrt{2}x_2, x_1^2, x_2^2)$$

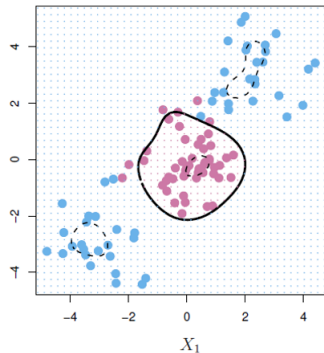
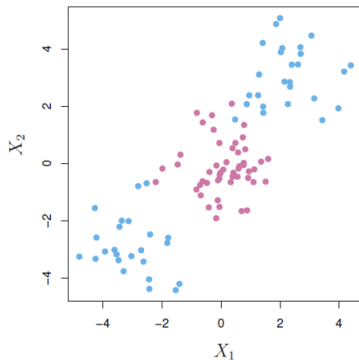
- e.g. Polynomial kernel of degree 3:



Radial Basis Kernels

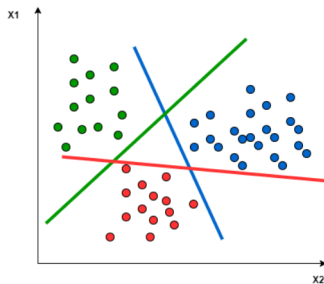
$$K(x, y) = \exp(-\gamma \|x - y\|_2^2) \quad (9)$$

- γ is the hyperparameter to be chosen to control the bias-variance trade-off.
- e.g. Radial basis functions (RBF) kernel:

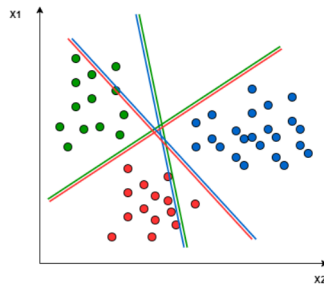


Multi-class SVM

- **One-vs-All:** Fit all K different 2-class SVM classifiers $\hat{f}_k(x)$, $\forall k \in \{1, 2, \dots, K\}$; each class versus the rest. Classify x^* to the class for which $\hat{f}_k(x)$ is the largest.
- **One-vs-One:** Fit all $\binom{K}{2}$ pairwise classifiers $\hat{f}_{kl}(x)$. Classify x^* to the class that wins the most pairwise competitions.

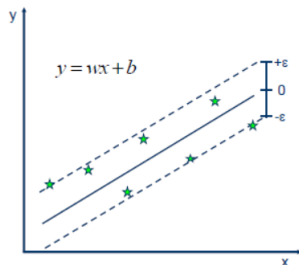


(a) One-vs-All



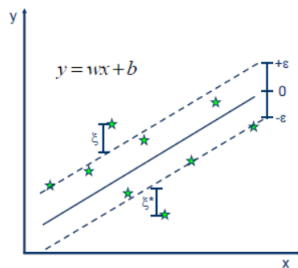
(b) One-vs-One

Support Vector Regression



$$\begin{aligned}
 &\min_{\beta, \beta_0} \quad \frac{1}{2} \|\beta\|_2^2 \\
 &\text{s.t.} \quad |y^{(i)} - (\beta^T x^{(i)} + \beta_0)|_2 \leq \epsilon \quad \forall i \in \{1, 2, \dots, N\}
 \end{aligned} \tag{10}$$

Soft Support Vector Regression



$$\begin{aligned}
 \min_{\beta, \beta_0} \quad & \frac{1}{2} \|\beta\|_2^2 + C \sum_{i=1}^N \xi_i \\
 \text{s.t.} \quad & |y^{(i)} - (\beta^T x^{(i)} + \beta_0)|_2 \leq \epsilon + \xi_i \\
 & \xi_i \geq 0 \quad \forall i \in \{1, 2, \dots, N\}
 \end{aligned} \tag{11}$$

Final Notes

Thank You!

Any Question?