Support Vector Machines and Kernel Methods

March 23, 2021

Slides Courtesy of Jiayu Zhou

SVM at a High Level

- Linear models are powerful and expressive
- Nonlinear transformations increase expressiveness but can result in overfitting and increased computation time

SVM at a High Level

- Linear models are powerful and expressive
- Nonlinear transformations increase expressiveness but can result in overfitting and increased computation time
- SVMs allow us to have expressive models without these drawbacks
- SVM uses a *(safety) cushion* to separate the data
  - Helps SVM be robust to noise $\rightarrow$ avoid overfitting

SVM at a High Level

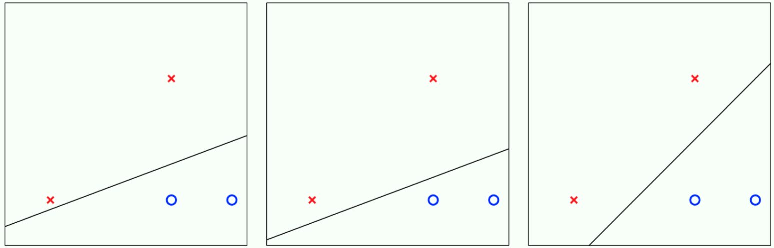
- Linear models are powerful and expressive
- Nonlinear transformations increase expressiveness but can result in overfitting and increased computation time
- SVMs allow us to have expressive models without these drawbacks
- SVM uses a *(safety) cushion* to separate the data
  - Helps SVM be robust to noise $\rightarrow$ avoid overfitting
- Kernels: computationally efficient ways to use high dimensional nonlinear transforms

SVM at a High Level

- Linear models are powerful and expressive
- Nonlinear transformations increase expressiveness but can result in overfitting and increased computation time
- SVMs allow us to have expressive models without these drawbacks
- SVM uses a *(safety) cushion* to separate the data
    - Helps SVM be robust to noise $\rightarrow$ avoid overfitting
- Kernels: computationally efficient ways to use high dimensional nonlinear transforms
- SVM: cushion $+$ kernel trick $=$ powerful nonlinear model with automatic regularization

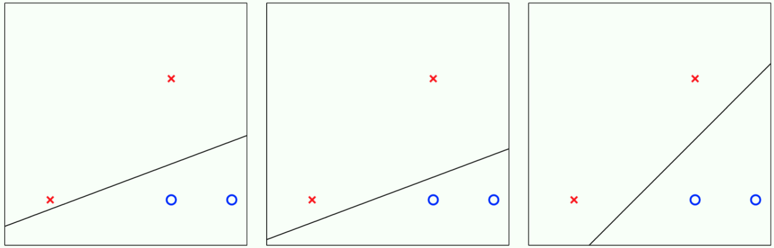
Which Separator Would You Pick?

Perceptron attempts to separate the data with a line (possible in this 2D example)



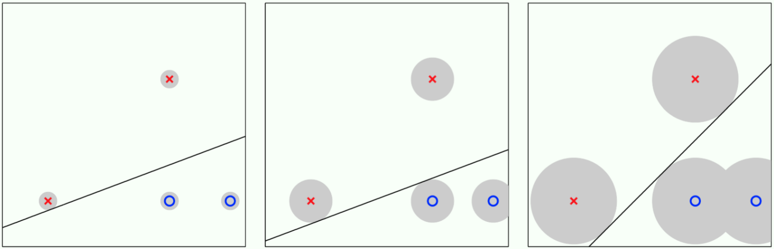
Which Separator Would You Pick?

Perceptron attempts to separate the data with a line
(possible in this 2D example)



If all lines have the same E_{out} and $E_{in} = 0$, which line would you pick?

Robustness to Noisy Data

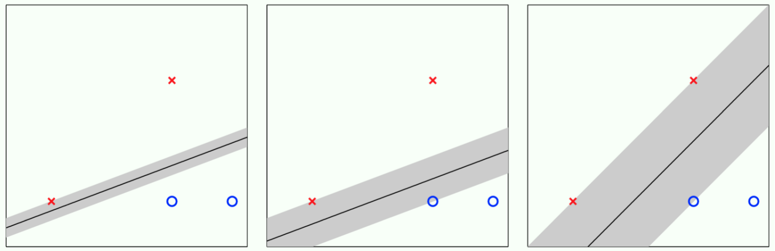


- Being robust to noise, or measurement error, is ideal
- Grey circles indicate a radius of possible noise
    - True data point falls anywhere within this ‘region of uncertainty’

Robustness to Noisy Data

- More noise a separator can tolerate = safer
- What if we look at noise tolerance from the view of the separator instead of the data points?

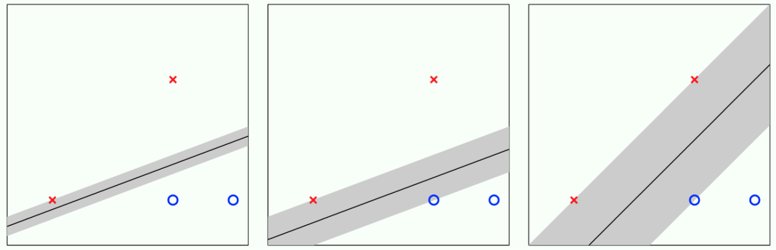
Thicker Cushion Means More Robustness



Place a *cushion* on each side of the separator

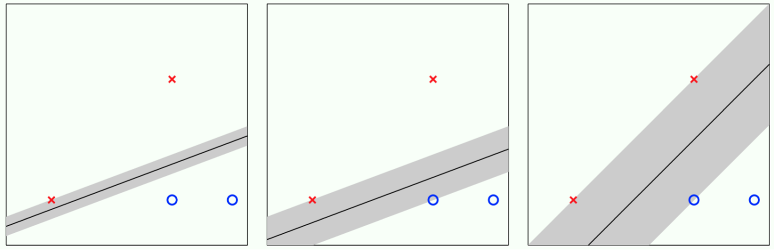
- *Fat* cushion or *width* of the hyperplane
- Separates the data if no point lies within the cushion

Thicker Cushion Means More Robustness



- To get the thickest cushion, we keep extending it on both sides until we hit a data point
- Thickness = amount of noise the separator can tolerate

Thicker Cushion Means More Robustness



- To get the thickest cushion, we keep extending it on both sides until we hit a data point
- Thickness = amount of noise the separator can tolerate
- Maximum thickness, or noise tolerance, possible for a separator is its **margin**

Three Crucial Questions

1. Can we efficiently find the widest separator?

$$\max_h \mathsf{Width}(h)$$
$$\Rightarrow \max_h \min_{x \in \mathcal{D}} \mathsf{dist}(x, h)$$

   - Given x and h, how to get the analytical form of $\mathsf{dist}(x, h)$?

2. Is a wider separator really better than a thin one?
3. What do we do if the data is not separable?

Three Crucial Questions

1. Can we efficiently find the widest separator?

$$\max_h \text{Width}(h)$$
$$\Rightarrow \max_h \min_{x \in \mathcal{D}} \text{dist}(x, h)$$

   - Given x and h, how to get the analytical form of $\text{dist}(x, h)$?

2. Is a wider separator really better than a thin one?
3. What do we do if the data is not separable?

NOTE: In 2D we typically refer to the line as the *separator*, in higher dimensions it's a *hyperplane*.

Pulling Out the Bias

Before

$$\mathbf{x} \in \{1\} \times \mathbb{R}^d; \mathbf{w} \in \mathbb{R}^{d+1}$$

$$\mathbf{x} = \begin{bmatrix} 1 \\ x_1 \\ \vdots \\ x_d \end{bmatrix}; \mathbf{w} = \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_d \end{bmatrix}$$

$$\mathsf{h}(\mathsf{x}) = sign(\mathbf{w}^T\mathbf{x})$$

Pulling Out the Bias

Before

$$\mathbf{x} \in \{1\} \times \mathbb{R}^d; \mathbf{w} \in \mathbb{R}^{d+1}$$

$$\mathbf{x} = \begin{bmatrix} 1 \\ x_1 \\ \vdots \\ x_d \end{bmatrix}; \mathbf{w} = \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_d \end{bmatrix}$$

$\mathsf{h}(\mathsf{x}) = sign(\mathbf{w}^T \mathbf{x})$

After

$$\mathbf{x} \in \mathbb{R}^d; b \in \mathbb{R}, \mathbf{w} \in \mathbb{R}^d$$

$$\mathbf{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_d \end{bmatrix}; \mathbf{w} = \begin{bmatrix} w_1 \\ \vdots \\ w_d \end{bmatrix}$$

bias b

$\mathsf{h}(\mathsf{x}) = sign(\mathbf{w}^T \mathbf{x} + b)$

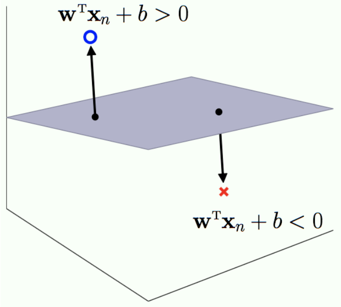
Separating the Data

A maximum-margin separating hyperplane has 2 defining characteristics:

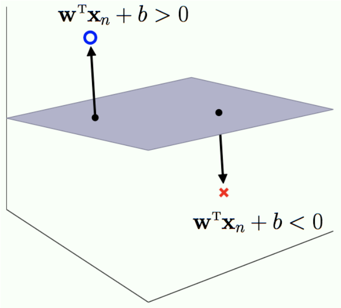
- It separates the data
- It has the thickest cushion among hyperplanes that separate the data

Separating the Data

Hyperplane $h = (b, \mathbf{w})$



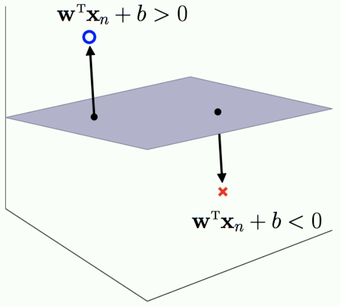
Separating the Data



Hyperplane $h = (b, \mathbf{w})$
h separates the data iff for $n = 1..N$:

$$y_n(\mathbf{w}^T\mathbf{x}_n + b) > 0$$

Separating the Data

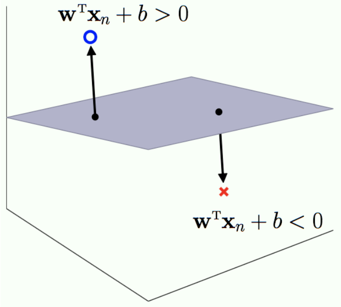


Hyperplane $h = (b, \mathbf{w})$
h separates the data iff for $n = 1..N$:

$$y_n(\mathbf{w}^T\mathbf{x}_n + b) > 0$$

Magnitude on its own isn't meaningful because we could rescale **w** and b.

Separating the Data



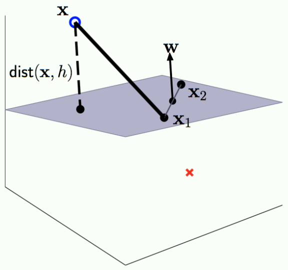
Hyperplane $h = (b, \mathbf{w})$
h separates the data iff for $n = 1..N$:

$$y_n(\mathbf{w}^T\mathbf{x}_n + b) > 0$$

Magnitude on its own isn't meaningful because we could rescale **w** and *b*.
By rescaling the weights and bias:

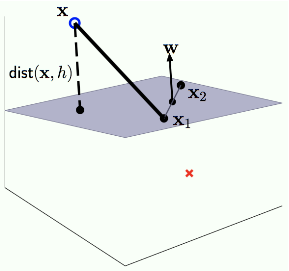
$$\min_{n=1,\dots,N} y_n(\mathbf{w}^T\mathbf{x}_n + b) = 1$$

Margin of a Hyperplane



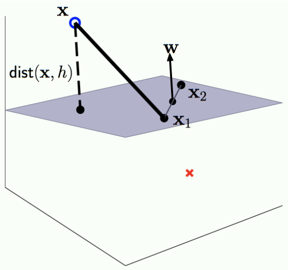
- To compute the margin of a separating hyperplane, we need to compute the distance from *h* to the *nearest* data point
- $dist(x, h)$: distance from arbitrary point **x** to separating hyperplane $h = (b, w)$

Margin of a Hyperplane



- To compute the margin of a separating hyperplane, we need to compute the distance from h to the *nearest* data point
- $dist(x, h)$: distance from arbitrary point **x** to separating hyperplane $h = (b, w)$
- $\mathbf{w}$ is normal to the hyperplane
  $\mathbf{w}^T(\mathbf{x}_2 - \mathbf{x}_1) = \mathbf{w}^T\mathbf{x}_2 - \mathbf{w}^T\mathbf{x}_1 = -b + b = 0$

Margin of a Hyperplane

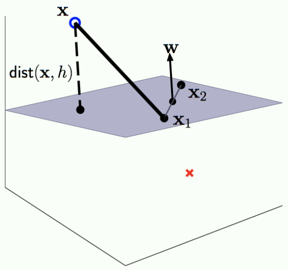


- Scalar projection:

$$\mathbf{a}^T\mathbf{b} = \|\mathbf{a}\|\|\mathbf{b}\|\cos(\mathbf{a}, \mathbf{b})$$

$$\Rightarrow \mathbf{a}^T\mathbf{b}/\|\mathbf{b}\| = \|\mathbf{a}\|\cos(\mathbf{a}, \mathbf{b})$$

Margin of a Hyperplane

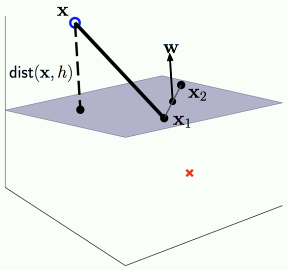


- Scalar projection:

$$\mathbf{a}^T\mathbf{b} = \|\mathbf{a}\|\|\mathbf{b}\|\cos(\mathbf{a}, \mathbf{b})$$
$$\Rightarrow \mathbf{a}^T\mathbf{b}/\|\mathbf{b}\| = \|\mathbf{a}\|\cos(\mathbf{a}, \mathbf{b})$$

- Let $\mathbf{x}_\perp$ be the orthogonal projection of $\mathbf{x}$ to h, then the distance to hyperplane is given by projection of $\mathbf{x} - \mathbf{x}_\perp$ to $\mathbf{w}$

Margin of a Hyperplane



- Scalar projection:

$$\mathbf{a}^T\mathbf{b} = \|\mathbf{a}\|\|\mathbf{b}\|\cos(\mathbf{a}, \mathbf{b})$$
$$\Rightarrow \mathbf{a}^T\mathbf{b}/\|\mathbf{b}\| = \|\mathbf{a}\|\cos(\mathbf{a}, \mathbf{b})$$

- Let $\mathbf{x}_\perp$ be the orthogonal projection of $\mathbf{x}$ to h, then the distance to hyperplane is given by projection of $\mathbf{x} - \mathbf{x}_\perp$ to $\mathbf{w}$

$$\begin{aligned}\text{dist}(\mathbf{x}, h) &= \frac{1}{\|\mathbf{w}\|} \cdot |\mathbf{w}^T\mathbf{x} - \mathbf{w}^T\mathbf{x}_\perp| \\ &= \frac{1}{\|\mathbf{w}\|} \cdot |\mathbf{w}^T\mathbf{x} + b|\end{aligned}$$

Margin of a Separating Hyperplane

- Given data points $\mathsf{x}_1..\mathsf{x}_N$ and a hyperplane $h = (b, \mathbf{w})$ that satisfies the separating condition

$$\text{dist}(\mathbf{x}, h) = \frac{1}{\|\mathbf{w}\|} \cdot |\mathbf{w}^T\mathbf{x} + b| = \frac{1}{\|\mathbf{w}\|} \cdot |y_n(\mathbf{w}^T\mathbf{x} + b)| = \frac{1}{\|\mathbf{w}\|} \cdot y_n(\mathbf{w}^T\mathbf{x} + b)$$

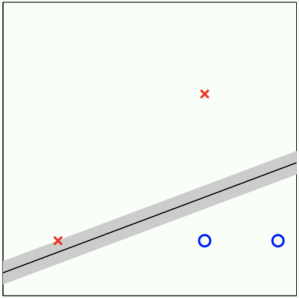
Margin of a Separating Hyperplane

- Given data points $\mathrm{x}_1..\mathrm{x}_N$ and a hyperplane $h = (b, \mathbf{w})$ that satisfies the separating condition

$$\text{dist}(\mathbf{x}, h) = \frac{1}{\|\mathbf{w}\|} \cdot |\mathbf{w}^T\mathbf{x} + b| = \frac{1}{\|\mathbf{w}\|} \cdot |y_n(\mathbf{w}^T\mathbf{x} + b)| = \frac{1}{\|\mathbf{w}\|} \cdot y_n(\mathbf{w}^T\mathbf{x} + b)$$

Width

= Distance to the closest point



$$\begin{aligned} \text{Width} &= \min_n \text{dist}(\mathbf{x}_n, h) \\ &= \frac{1}{\|\mathbf{w}\|} \min_n y_n(\mathbf{w}^T\mathbf{x} + b) \\ &= \frac{1}{\|\mathbf{w}\|} \end{aligned}$$

Maximizing the Margin

- Maximum-margin separating hyperplane *(b*, **w***)* satisfies the separating condition with minimum weight-norm
- Formal definition of margin:

$$\text{margin: } \gamma(h) = \frac{1}{\|\mathbf{w}\|}$$

Maximizing the Margin

- Maximum-margin separating hyperplane *(b*, **w***)* satisfies the separating condition with minimum weight-norm
- Formal definition of margin:

$$\text{margin: } \gamma(h) = \frac{1}{\|\mathbf{w}\|}$$

  - NOTE: Bias b does not appear in the margin.

Maximizing the Margin

- Maximum-margin separating hyperplane *(b*, $\mathbf{w}$*)* satisfies the separating condition with minimum weight-norm
- Formal definition of margin:

$$\text{margin: } \gamma(h) = \frac{1}{\|\mathbf{w}\|}$$

  - NOTE: Bias b does not appear in the margin.
- Objective for maximizing margin:
  Instead of minimizing the weight-norm, equivalently minimize:

$$\min_{b,\mathbf{w}} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w}$$

$$\text{subject to: } \quad \min_{n=1,\ldots,N} y_n(\mathbf{w}^T\mathbf{x}_n + b) = 1$$

Maximizing the Margin

- Maximum-margin separating hyperplane *(b*, **w***)* satisfies the separating condition with minimum weight-norm
- Formal definition of margin:

$$\text{margin: } \gamma(h) = \frac{1}{\|\mathbf{w}\|}$$

  - NOTE: Bias b does not appear in the margin.
- Objective for maximizing margin:
  Instead of minimizing the weight-norm, equivalently minimize:

$$\min_{b,\mathbf{w}} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w}$$

$$\text{subject to: } \min_{n=1,\ldots,N} y_n(\mathbf{w}^T\mathbf{x}_n + b) = 1$$

- An equivalent objective:

$$\text{subject to: } y_n(\mathbf{w}^T\mathbf{x}_n + b) \geq 1 \text{ for } n = 1, \ldots, N$$

Example - Toy Dataset

$$\min_{b,\mathbf{w}} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w}$$

subject to: $y_n(\mathbf{w}^T\mathbf{x}_n + b) \geq 1$ for $n = 1, \ldots, N$

Training Data:

$$X = \begin{bmatrix} 0 & 0 \\ 2 & 2 \\ 2 & 0 \\ 3 & 0 \end{bmatrix}, \mathbf{y} = \begin{bmatrix} -1 \\ -1 \\ +1 \\ +1 \end{bmatrix}$$

What is the margin?

Example - Toy Dataset

$$\min_{b,\mathbf{w}} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w}$$

subject to: $y_n(\mathbf{w}^T\mathbf{x}_n + b) \geq 1$ for $n = 1, \ldots, N$

$$X = \begin{bmatrix} 0 & 0 \\ 2 & 2 \\ 2 & 0 \\ 3 & 0 \end{bmatrix}, \mathbf{y} = \begin{bmatrix} -1 \\ -1 \\ +1 \\ +1 \end{bmatrix} \qquad \Rightarrow \begin{cases} (1) : -b \geq 1 \\ (2) : -(2w_1 + 2w_2 + b) \geq 1 \\ (3) : 2w_1 + b \geq 1 \\ (4) : 3w_1 + b \geq 1 \end{cases}$$

$$\begin{cases} (1) + (3) & \to w_1 \geq 1 \\ (2) + (3) & \to w_2 \leq -1 \end{cases} \Rightarrow \frac{1}{2}\mathbf{w}^T\mathbf{w} = \frac{1}{2}(w_1^2 + w_2^2) \geq 1$$

Example - Toy Dataset

$$\min_{b,\mathbf{w}} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w}$$

subject to: $y_n(\mathbf{w}^T\mathbf{x}_n + b) \geq 1$ for $n = 1, \ldots, N$

$$X = \begin{bmatrix} 0 & 0 \\ 2 & 2 \\ 2 & 0 \\ 3 & 0 \end{bmatrix}, \mathbf{y} = \begin{bmatrix} -1 \\ -1 \\ +1 \\ +1 \end{bmatrix} \qquad \Rightarrow \begin{cases} (1) : -b \geq 1 \\ (2) : -(2w_1 + 2w_2 + b) \geq 1 \\ (3) : 2w_1 + b \geq 1 \\ (4) : 3w_1 + b \geq 1 \end{cases}$$

$$\begin{cases} (1) + (3) & \rightarrow w_1 \geq 1 \\ (2) + (3) & \rightarrow w_2 \leq -1 \end{cases} \Rightarrow \frac{1}{2}\mathbf{w}^T\mathbf{w} = \frac{1}{2}(w_1^2 + w_2^2) \geq 1$$

Thus: $w_1 = 1, w_2 = -1, b = -1$

Example - Toy Dataset

- Given data $X = \begin{bmatrix} 0 & 0 \\ 2 & 2 \\ 2 & 0 \\ 3 & 0 \end{bmatrix}$
- Optimal solution

$$\mathbf{w}^* = \begin{bmatrix} w_1 = 1 \\ w_2 = -1 \end{bmatrix}, b^* = -1$$

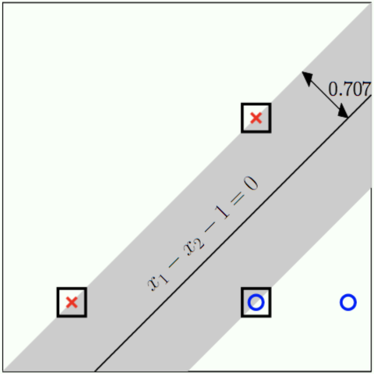
- Optimal hyperplane
  $g(\mathbf{x}) = \mathsf{sign}(x_1 - x_2 - 1)$

Example - Toy Dataset

- Given data $X = \begin{bmatrix} 0 & 0 \\ 2 & 2 \\ 2 & 0 \\ 3 & 0 \end{bmatrix}$
- Optimal solution

$$\mathbf{w}^* = \begin{bmatrix} w_1 = 1 \\ w_2 = -1 \end{bmatrix}, b^* = -1$$

- Optimal hyperplane
  $g(\mathbf{x}) = \text{sign}(x_1 - x_2 - 1)$
- margin:
  $\frac{1}{\|w\|} = \frac{1}{\sqrt{2}} \approx 0.707$

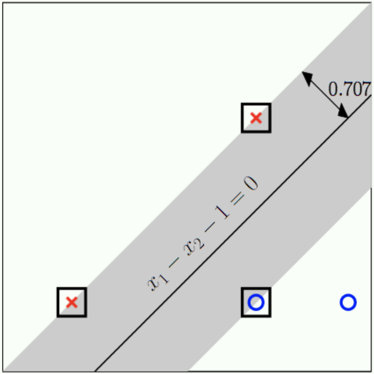


Example - Toy Dataset

- Given data $X = \begin{bmatrix} 0 & 0 \\ 2 & 2 \\ 2 & 0 \\ 3 & 0 \end{bmatrix}$
- Optimal solution

$$\mathbf{w}^* = \begin{bmatrix} w_1 = 1 \\ w_2 = -1 \end{bmatrix}, b^* = -1$$

- Optimal hyperplane
  $g(\mathbf{x}) = \text{sign}(x_1 - x_2 - 1)$
- margin:
  $\frac{1}{\|w\|} = \frac{1}{\sqrt{2}} \approx 0.707$



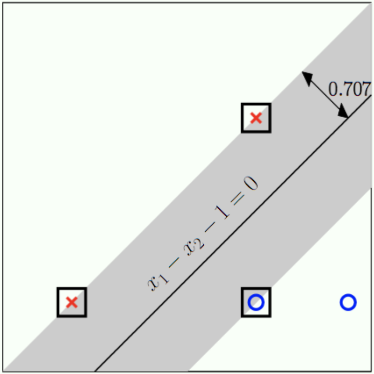
For data points (1), (2) and (3) $y_n({\mathbf{w}^*}^T \mathbf{x}^n + b^*) = 1$

Example - Toy Dataset

- Given data $X = \begin{bmatrix} 0 & 0 \\ 2 & 2 \\ 2 & 0 \\ 3 & 0 \end{bmatrix}$
- Optimal solution

$$\mathbf{w}^* = \begin{bmatrix} w_1 = 1 \\ w_2 = -1 \end{bmatrix}, b^* = -1$$

- Optimal hyperplane
  $g(\mathbf{x}) = \text{sign}(x_1 - x_2 - 1)$
- margin:
  $\frac{1}{\|w\|} = \frac{1}{\sqrt{2}} \approx 0.707$



For data points (1), (2) and (3) $y_n(\mathbf{w}^{*T}\mathbf{x}^n + b^*) = 1$
Support Vectors

Solver: Quadratic Programming (QP)

- For larger datasets, manually solving the optimization problem like the toy dataset is not feasible
- But it belongs to a family of optimization problems known as quadratic programming (QP)
  - Whenever minimizing a (convex) quadratic function subject to linear inequality constraints, can use QP
- Take given problem $\rightarrow$ convert to a standard form $\rightarrow$ input to QP solver

Standard Form of QP-problem

$$\min_{\mathbf{u} \in \mathbb{R}^q} \quad \frac{1}{2}\mathbf{u}^T Q \mathbf{u} + \mathbf{p}^T \mathbf{u}$$

subject to: $A\mathbf{u} \geq \mathbf{c}$

$$\mathbf{u}^* \leftarrow QP(Q, \mathbf{p}, A, \mathbf{c})$$

($Q = 0$ is linear programming)

Maximum Margin Hyperplane is QP

$$\min_{b,\mathbf{w}} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w}$$

subject to: $y_n(\mathbf{w}^T\mathbf{x}_n + b) \geq 1, \forall n$

$$\min_{\mathbf{u}\in\mathbb{R}^q} \quad \frac{1}{2}\mathbf{u}^T Q\mathbf{u} + \mathbf{p}^T\mathbf{u}$$

subject to: $A\mathbf{u} \geq \mathbf{c}$

$$\mathbf{u} = \begin{bmatrix} b \\ \mathbf{w} \end{bmatrix} \in \mathbb{R}^{d+1} \Rightarrow \frac{1}{2}\mathbf{w}^T\mathbf{w} = [b\mathbf{w}^T]\begin{bmatrix} 0 & \mathbf{0}_d^T \\ \mathbf{0}_d & I_d \end{bmatrix}\begin{bmatrix} b \\ \mathbf{w} \end{bmatrix} = \mathbf{u}^T\begin{bmatrix} 0 & \mathbf{0}_d^T \\ \mathbf{0}_d & I_d \end{bmatrix}\mathbf{u}$$

Maximum Margin Hyperplane is QP

$$\min_{b,\mathbf{w}} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w}$$

subject to: $y_n(\mathbf{w}^T\mathbf{x}_n + b) \geq 1, \forall n$

$$\min_{\mathbf{u}\in\mathbb{R}^q} \quad \frac{1}{2}\mathbf{u}^T Q\mathbf{u} + \mathbf{p}^T\mathbf{u}$$

subject to: $A\mathbf{u} \geq \mathbf{c}$

$$\mathbf{u} = \begin{bmatrix} b \\ \mathbf{w} \end{bmatrix} \in \mathbb{R}^{d+1} \Rightarrow \frac{1}{2}\mathbf{w}^T\mathbf{w} = [b\mathbf{w}^T]\begin{bmatrix} 0 & \mathbf{0}_d^T \\ \mathbf{0}_d & I_d \end{bmatrix}\begin{bmatrix} b \\ \mathbf{w} \end{bmatrix} = \mathbf{u}^T\begin{bmatrix} 0 & \mathbf{0}_d^T \\ \mathbf{0}_d & I_d \end{bmatrix}\mathbf{u}$$

$$Q = \begin{bmatrix} 0 & \mathbf{0}_d^T \\ \mathbf{0}_d & I_d \end{bmatrix}, \mathbf{p} = \mathbf{0}_{d+1}$$

Maximum Margin Hyperplane is QP

$$\min_{b,\mathbf{w}} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w}$$

subject to: $y_n(\mathbf{w}^T\mathbf{x}_n + b) \geq 1, \forall n$

$$\min_{\mathbf{u}\in\mathbb{R}^q} \quad \frac{1}{2}\mathbf{u}^T Q\mathbf{u} + \mathbf{p}^T\mathbf{u}$$

subject to: $A\mathbf{u} \geq \mathbf{c}$

$$\mathbf{u} = \begin{bmatrix} b \\ \mathbf{w} \end{bmatrix} \in \mathbb{R}^{d+1} \Rightarrow \frac{1}{2}\mathbf{w}^T\mathbf{w} = [b\mathbf{w}^T]\begin{bmatrix} 0 & \mathbf{0}_d^T \\ \mathbf{0}_d & I_d \end{bmatrix}\begin{bmatrix} b \\ \mathbf{w} \end{bmatrix} = \mathbf{u}^T\begin{bmatrix} 0 & \mathbf{0}_d^T \\ \mathbf{0}_d & I_d \end{bmatrix}\mathbf{u}$$

$$Q = \begin{bmatrix} 0 & \mathbf{0}_d^T \\ \mathbf{0}_d & I_d \end{bmatrix}, \mathbf{p} = \mathbf{0}_{d+1}$$

$$y_n(\mathbf{w}^T\mathbf{x}_n + b) \geq 1 = [y_n, y_n\mathbf{x}_n^T]\mathbf{u} \geq 1 \Rightarrow \begin{bmatrix} y_1 & y_1\mathbf{x}_1^T \\ \vdots & \vdots \\ y_N & y_N\mathbf{x}_N^T \end{bmatrix}\mathbf{u} \geq \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}$$

Maximum Margin Hyperplane is QP

$$\min_{b,\mathbf{w}} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w}$$

subject to: $y_n(\mathbf{w}^T\mathbf{x}_n + b) \geq 1, \forall n$

$$\min_{\mathbf{u}\in\mathbb{R}^q} \quad \frac{1}{2}\mathbf{u}^T Q\mathbf{u} + \mathbf{p}^T\mathbf{u}$$

subject to: $A\mathbf{u} \geq \mathbf{c}$

$$\mathbf{u} = \begin{bmatrix} b \\ \mathbf{w} \end{bmatrix} \in \mathbb{R}^{d+1} \Rightarrow \frac{1}{2}\mathbf{w}^T\mathbf{w} = [b\mathbf{w}^T]\begin{bmatrix} 0 & \mathbf{0}_d^T \\ \mathbf{0}_d & I_d \end{bmatrix}\begin{bmatrix} b \\ \mathbf{w} \end{bmatrix} = \mathbf{u}^T\begin{bmatrix} 0 & \mathbf{0}_d^T \\ \mathbf{0}_d & I_d \end{bmatrix}\mathbf{u}$$

$$Q = \begin{bmatrix} 0 & \mathbf{0}_d^T \\ \mathbf{0}_d & I_d \end{bmatrix}, \mathbf{p} = \mathbf{0}_{d+1}$$

$$y_n(\mathbf{w}^T\mathbf{x}_n + b) \geq 1 = [y_n, y_n\mathbf{x}_n^T]\mathbf{u} \geq 1 \Rightarrow \begin{bmatrix} y_1 & y_1\mathbf{x}_1^T \\ \vdots & \vdots \\ y_N & y_N\mathbf{x}_N^T \end{bmatrix}\mathbf{u} \geq \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}$$

$$A = \begin{bmatrix} y_1 & y_1\mathbf{x}_1^T \\ \vdots & \vdots \\ y_N & y_N\mathbf{x}_N^T \end{bmatrix}, \boldsymbol{c} = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}$$

Back To Our Example

Exercise:

$$X = \begin{bmatrix} 0 & 0 \\ 2 & 2 \\ 2 & 0 \\ 3 & 0 \end{bmatrix}, \mathbf{y} = \begin{bmatrix} -1 \\ -1 \\ +1 \\ +1 \end{bmatrix} \qquad \begin{cases} (1) : -b \geq 1 \\ (2) : -(2w_1 + 2w_2 + b) \geq 1 \\ (3) : 2w_1 + b \geq 1 \\ (4) : 3w_1 + b \geq 1 \end{cases}$$

Show the corresponding $Q, \mathbf{p}, A, \mathbf{c}$.

Back To Our Example

Exercise:

$$X = \begin{bmatrix} 0 & 0 \\ 2 & 2 \\ 2 & 0 \\ 3 & 0 \end{bmatrix}, \mathbf{y} = \begin{bmatrix} -1 \\ -1 \\ +1 \\ +1 \end{bmatrix} \qquad \begin{cases} (1) : -b \geq 1 \\ (2) : -(2w_1 + 2w_2 + b) \geq 1 \\ (3) : 2w_1 + b \geq 1 \\ (4) : 3w_1 + b \geq 1 \end{cases}$$

Show the corresponding $Q, \mathbf{p}, A, \mathbf{c}$.

$$Q = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \mathbf{p} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, A = \begin{bmatrix} -1 & 0 & 0 \\ -1 & -2 & -2 \\ 1 & 2 & 0 \\ 1 & 3 & 0 \end{bmatrix}, \mathbf{c} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

Back To Our Example

Exercise:

$$X = \begin{bmatrix} 0 & 0 \\ 2 & 2 \\ 2 & 0 \\ 3 & 0 \end{bmatrix}, \mathbf{y} = \begin{bmatrix} -1 \\ -1 \\ +1 \\ +1 \end{bmatrix} \qquad \begin{cases} (1) : -b \geq 1 \\ (2) : -(2w_1 + 2w_2 + b) \geq 1 \\ (3) : 2w_1 + b \geq 1 \\ (4) : 3w_1 + b \geq 1 \end{cases}$$

Show the corresponding $Q, \mathbf{p}, A, \mathbf{c}$.

$$Q = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \mathbf{p} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, A = \begin{bmatrix} -1 & 0 & 0 \\ -1 & -2 & -2 \\ 1 & 2 & 0 \\ 1 & 3 & 0 \end{bmatrix}, \mathbf{c} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

Use your QP-solver to give

$$\boldsymbol{u}^* = [b^*, w_1^*, w_2^*]^T = [-1, 1, -1]$$

Linear Hard-Margin SVM with QP

Algorithm for finding optimal hyperplane:

1. Let $\boldsymbol{p} = \mathbf{0}_{d+1}$ ($(d+1)$-dimensional zero vector) and $\boldsymbol{c} = \mathbf{1}_N$ (N-dimensional vector of ones). Construct matrices Q and A, where

$$A = \begin{bmatrix} y_1 & -y_1 \mathbf{x}_1^T - \\ \vdots & \vdots \\ y_N & -y_N \mathbf{x}_N^T - \end{bmatrix}, Q = \begin{bmatrix} 0 & \mathbf{0}_d^T \\ \mathbf{0}_d & I_d \end{bmatrix}$$

2. Return $\begin{bmatrix} b^* \\ \mathbf{w}^* \end{bmatrix} = \boldsymbol{u}^* \leftarrow QP(Q, \boldsymbol{p}, A, \boldsymbol{c})$.
3. The final hypothesis is $g(\mathbf{x}) = \text{sign}({\mathbf{w}^*}^T \mathbf{x} + b^*)$.

Is a Wide Separator Better?

$$\min_{\mathbf{w}} \quad E_{in}(\mathbf{w})$$

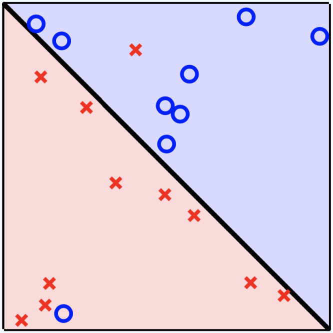
$$\text{subject to: } \mathbf{w}^T\mathbf{w} \leq C$$

	optimal hyperplane	regularization
minimize	$\mathbf{w}^T\mathbf{w}$	E_{in}
subject to	$E_{in} = 0$	$\mathbf{w}^T\mathbf{w} \leq C$

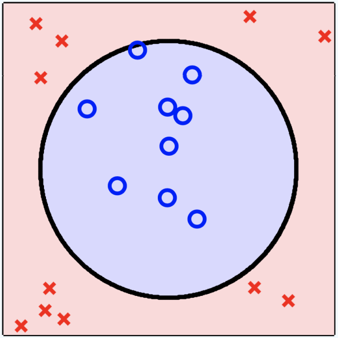
Is a Wide Separator Better?

- Larger-margin separator yields better performance
- Fat/wide hyperplanes generalize better than thin hyperplanes
- Out-of-sample error can be small, even if dimension d is large

How to Handle Non-separable Data?

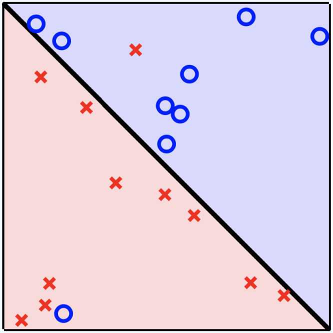


(a) Few noisy data.

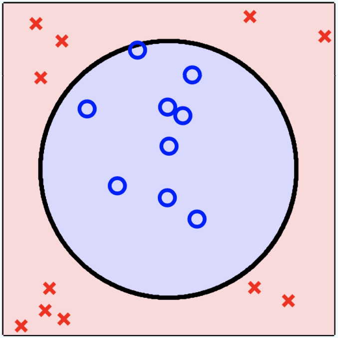


(b) Nonlinearly separable.

How to Handle Non-separable Data?



(a) Few noisy data.



(b) Nonlinearly separable.

(a) Tolerate noisy data points: soft-margin SVM

(b) Inherent nonlinear boundary: nonlinear transformation with optimal hyperplane

Nonlinear Transformation

$$\mathbf{\Phi}_1(\mathbf{x}) = (x_1, x_2)$$
$$\mathbf{\Phi}_2(\mathbf{x}) = (x_1, x_2, x_1^2, x_1 x_2, x_2^2)$$
$$\mathbf{\Phi}_3(\mathbf{x}) = (x_1, x_2, x_1^2, x_1 x_2, x_2^2, x_1^3, x_1^2 x_2, x_1 x_2^2, x_2^3)$$

Nonlinear Transformation

- Using the nonlinear transform with the optimal hyperplane using the transform $\mathbf{\Phi}$: $\mathbb{R}^d \to \mathbb{R}^{\tilde{d}}$:

$$\mathbf{z}_n = \mathbf{\Phi}(\mathbf{x}_n)$$

Nonlinear Transformation

- Using the nonlinear transform with the optimal hyperplane using the transform $\mathbf{\Phi}$: $\mathbb{R}^d \to \mathbb{R}^{\tilde{d}}$:

$$\mathbf{z}_n = \mathbf{\Phi}(\mathbf{x}_n)$$

- Solve the hard-margin SVM in the $\mathcal{Z}$-space $(\tilde{\mathbf{w}}^*, \tilde{b}^*)$:

$$\min_{\tilde{b}, \tilde{\mathbf{w}}} \quad \frac{1}{2}\tilde{\mathbf{w}}^T\tilde{\mathbf{w}}$$

$$\text{subject to: } y_n(\tilde{\mathbf{w}}^T\mathbf{z}_n + \tilde{b}) \geq 1, \forall n$$

Nonlinear Transformation

- Using the nonlinear transform with the optimal hyperplane using the transform $\mathbf{\Phi}$: $\mathbb{R}^d \to \mathbb{R}^{\tilde{d}}$:

$$\mathbf{z}_n = \mathbf{\Phi}(\mathbf{x}_n)$$

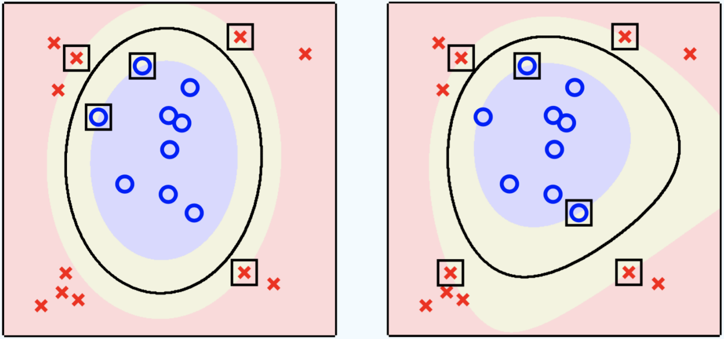
- Solve the hard-margin SVM in the $\mathcal{Z}$-space $(\tilde{\mathbf{w}}^*, \tilde{b}^*)$:

$$\min_{\tilde{b}, \tilde{\mathbf{w}}} \quad \frac{1}{2}\tilde{\mathbf{w}}^T\tilde{\mathbf{w}}$$
$$\text{subject to: } y_n(\tilde{\mathbf{w}}^T\mathbf{z}_n + \tilde{b}) \geq 1, \forall n$$

- Final hypothesis:

$$g(\mathbf{x}) = \mathsf{sign}(\tilde{\mathbf{w}}^{*T}\mathbf{\Phi}(\mathbf{x}) + \tilde{b}^*)$$

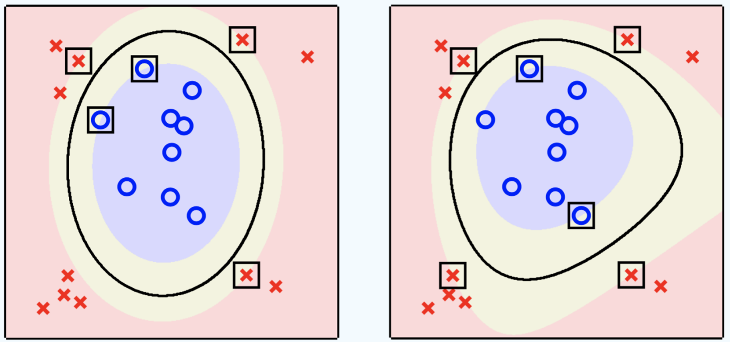
SVM and Nonlinear Transformation



(a) SVM with Φ_2 (b) SVM with Φ_3

The margin is shaded in yellow and the support vectors are boxed.

SVM and Nonlinear Transformation

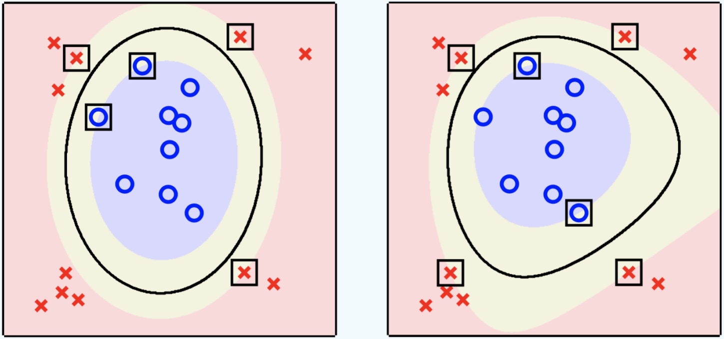


(a) SVM with Φ_2 (b) SVM with Φ_3

The margin is shaded in yellow and the support vectors are boxed.

- For $\mathbf{\Phi}_2$, $\tilde{d}_2 = 5$ and for $\mathbf{\Phi}_3$, $\tilde{d}_3 = 9$

SVM and Nonlinear Transformation



(a) SVM with Φ_2 (b) SVM with Φ_3

The margin is shaded in yellow and the support vectors are boxed.

- For $\mathbf{\Phi}_2$, $\tilde{d}_2 = 5$ and for $\mathbf{\Phi}_3$, $\tilde{d}_3 = 9$
- $\tilde{d}_3$ is nearly double $\tilde{d}_2$, yet the resulting SVM separator is not severely overfitting with $\mathbf{\Phi}_3$ (regularization?)

Support Vector Machine Summary

- Very powerful, easy to use linear model which comes with automatic regularization

Support Vector Machine Summary

- Very powerful, easy to use linear model which comes with automatic regularization
- To fully exploit SVM capabilities: **Kernel**
    - Use to implement nonlinear transforms
    - Potential robustness to overfitting even after transforming to a much higher dimension
    - How about infinite dimensional transforms?!
    - Kernel trick and Dual SVM next class!

SVM: Part 2

Support Vector Machines and Kernel Methods

March 25, 2021

SVM Dual: Formulation

- Primal and dual in optimization.

SVM Dual: Formulation

- Primal and dual in optimization.
- The dual view of SVM enables us to exploit the kernel trick.

SVM Dual: Formulation

- Primal and dual in optimization.
- The dual view of SVM enables us to exploit the kernel trick.
- In the primal SVM problem we solve $\mathbf{w} \in \mathbb{R}^d, b$, while in the dual problem we solve $\boldsymbol{\alpha} \in \mathbb{R}^N$

$$\max_{\boldsymbol{\alpha} \in \mathbb{R}^N} \sum_{n=1}^{N} \alpha_n - \frac{1}{2} \sum_{m=1}^{N} \sum_{n=1}^{N} \alpha_n \alpha_m y_n y_m \mathbf{x}_n^T \mathbf{x}_m$$

$$\text{subject to } \sum_{n=1}^{N} y_n \alpha_n = 0, \alpha_n \geq 0, \forall n$$

which is also a QP problem.

SVM Dual: Prediction

- We can obtain the primal solution:

$$\mathbf{w}^* = \sum_{n=1}^{N} y_n \alpha_n^* \mathbf{x}_n$$

where for support vectors $\alpha_n > 0$

SVM Dual: Prediction

- We can obtain the primal solution:

$$\mathbf{w}^* = \sum_{n=1}^{N} y_n \alpha_n^* \mathbf{x}_n$$

where for support vectors $\alpha_n > 0$

- The optimal hypothesis:

$$\begin{aligned} g(\mathbf{x}) &= \mathsf{sign}(\mathbf{w}^{*T}\mathbf{x} + b^*) \\ &= \mathsf{sign}\left(\sum_{n=1}^{N} y_n \alpha_n^* \mathbf{x}_n^T \mathbf{x} + b^*\right) \\ &= \mathsf{sign}\left(\sum_{\alpha_n^* > 0} y_n \alpha_n^* \mathbf{x}_n^T \mathbf{x} + b^*\right) \end{aligned}$$

Dual SVM: Summary

$$\max_{\boldsymbol{\alpha}\in\mathbb{R}^N} \sum_{n=1}^{N} \alpha_n - \frac{1}{2}\sum_{m=1}^{N}\sum_{n=1}^{N} \alpha_n \alpha_m y_n y_m \mathbf{x}_n^T \mathbf{x}_m$$

$$\text{subject to } \sum_{n=1}^{N} y_n \alpha_n = 0, \alpha_n \geq 0, \forall n$$

$$\mathbf{w}^* = \sum_{n=1}^{N} y_n \alpha_n^* \mathbf{x}_n$$

Common SVM Basis Functions

- $\mathbf{z}_k$ = polynomial terms of $\mathbf{x}_k$ of degree 1 to q
- $\mathbf{z}_k$ = radial basis function of $\mathbf{x}_k$

$$\mathbf{z}_k(j) = \phi_j(\mathbf{x}_k) = \exp(-|\mathbf{x}_k - \mathbf{c}_j|^2/\sigma^2)$$

- $\mathbf{z}_k$ = sigmoid functions of $\mathbf{x}_k$

Quadratic Basis Functions

$$\mathbf{\Phi}(\mathbf{x}) = \begin{bmatrix} 1 \\ \sqrt{2}x_1 \\ \vdots \\ \sqrt{2}x_d \\ x_1^2 \\ \vdots \\ x_d^2 \\ \sqrt{2}x_1x_2 \\ \vdots \\ \sqrt{2}x_1x_d \\ \sqrt{2}x_2x_3 \\ \vdots \\ \sqrt{2}x_{d-1}x_d \end{bmatrix}$$

- Including Constant Term, Linear Terms, Pure Quadratic Terms, Quadratic Cross-Terms
- The number of terms is approximately $d^2/2$.
- You may be wondering what those $\sqrt{2}$s are doing. You'll find out why they're there soon.

Dual SVM: Non-linear Transformation

$$\max_{\boldsymbol{\alpha}\in\mathbb{R}^N} \sum_{n=1}^{N} \alpha_n - \frac{1}{2}\sum_{m=1}^{N}\sum_{n=1}^{N} \alpha_n\alpha_m y_n y_m \boldsymbol{\Phi}(\mathbf{x}_n)^T \boldsymbol{\Phi}(\mathbf{x}_m)$$

$$\text{subject to } \sum_{n=1}^{N} y_n\alpha_n = 0, \alpha_n \geq 0, \forall n$$

$$\mathbf{w}^* = \sum_{n=1}^{N} y_n \alpha_n^* \Phi(\mathbf{x}_n)$$

- Need to prepare a matrix Q, $Q_{nm} = y_n y_m \boldsymbol{\Phi}(\mathbf{x}_n)^T \boldsymbol{\Phi}(\mathbf{x}_m)$
- Cost?

Dual SVM: Non-linear Transformation

$$\max_{\boldsymbol{\alpha}\in\mathbb{R}^N} \sum_{n=1}^{N} \alpha_n - \frac{1}{2}\sum_{m=1}^{N}\sum_{n=1}^{N} \alpha_n\alpha_m y_n y_m \boldsymbol{\Phi}(\mathbf{x}_n)^T \boldsymbol{\Phi}(\mathbf{x}_m)$$

$$\text{subject to } \sum_{n=1}^{N} y_n\alpha_n = 0, \alpha_n \geq 0, \forall n$$

$$\mathbf{w}^* = \sum_{n=1}^{N} y_n \alpha_n^* \Phi(\mathbf{x}_n)$$

- Need to prepare a matrix Q, $Q_{nm} = y_n y_m \boldsymbol{\Phi}(\mathbf{x}_n)^T \boldsymbol{\Phi}(\mathbf{x}_m)$
- Cost?
  - We must do $N^2/2$ dot products to get this matrix ready.
  - Each dot product requires $d^2/2$ additions and multiplications, The whole thing costs $N^2 d^2/4$.

Quadratic Dot Products

$$\mathbf{\Phi}(\mathbf{a})^T\mathbf{\Phi}(\mathbf{b}) = \begin{bmatrix} 1 \\ \sqrt{2}a_1 \\ \vdots \\ \sqrt{2}a_m \\ a_1^2 \\ \vdots \\ a_m^2 \\ \sqrt{2}a_1a_2 \\ \vdots \\ \sqrt{2}a_1a_d \\ \sqrt{2}a_2a_3 \\ \vdots \\ \sqrt{2}a_{d-1}a_d \end{bmatrix}^T \begin{bmatrix} 1 \\ \sqrt{2}b_1 \\ \vdots \\ \sqrt{2}b_d \\ b_1^2 \\ \vdots \\ b_d^2 \\ \sqrt{2}b_1b_2 \\ \vdots \\ \sqrt{2}b_1b_d \\ \sqrt{2}b_2b_3 \\ \vdots \\ \sqrt{2}b_{d-1}b_d \end{bmatrix}$$

- Constant Term 1
- Linear Terms

$$\sum_{i=1}^{d} 2a_ib_i$$

- Pure Quadratic Terms

$$\sum_{i=1}^{d} a_i^2b_i^2$$

- Quadratic Cross-Terms

$$\sum_{i=1}^{d}\sum_{j=i+1}^{d} 2a_ia_jb_ib_j$$

Quadratic Dot Product

- Does $\mathbf{\Phi}(\mathbf{a})^T\mathbf{\Phi}(\mathbf{b})$ look familiar?

$$\mathbf{\Phi}(\mathbf{a})^T\mathbf{\Phi}(\mathbf{b}) = 1 + 2\sum_{i=1}^{d} a_i b_i + \sum_{i=1}^{d} a_i^2 b_i^2 + \sum_{i=1}^{d}\sum_{j=i+1}^{d} 2a_i a_j b_i b_j$$

Quadratic Dot Product

- Does $\mathbf{\Phi}(\mathbf{a})^T\mathbf{\Phi}(\mathbf{b})$ look familiar?

$$\mathbf{\Phi}(\mathbf{a})^T\mathbf{\Phi}(\mathbf{b}) = 1 + 2\sum_{i=1}^{d} a_i b_i + \sum_{i=1}^{d} a_i^2 b_i^2 + \sum_{i=1}^{d}\sum_{j=i+1}^{d} 2a_i a_j b_i b_j$$

- Try this: $(\boldsymbol{a}^T\boldsymbol{b} + 1)^2$

Quadratic Dot Product

- Does $\mathbf{\Phi}(\mathbf{a})^T\mathbf{\Phi}(\mathbf{b})$ look familiar?

$$\mathbf{\Phi}(\mathbf{a})^T\mathbf{\Phi}(\mathbf{b}) = 1 + 2\sum_{i=1}^{d} a_i b_i + \sum_{i=1}^{d} a_i^2 b_i^2 + \sum_{i=1}^{d}\sum_{j=i+1}^{d} 2a_i a_j b_i b_j$$

- Try this: $(\boldsymbol{a}^T\boldsymbol{b}+1)^2$

$$\begin{aligned}(\boldsymbol{a}^T\boldsymbol{b}+1)^2 &= (\boldsymbol{a}^T\boldsymbol{b})^2 + 2\boldsymbol{a}^T\boldsymbol{b} + 1 \\ &= \left(\sum_{i=1}^{d} a_i b_i\right)^2 + 2\sum_{i=1}^{d} a_i b_i + 1 \\ &= \sum_{i=1}^{d}\sum_{j=1}^{d} a_i b_i a_j b_j + 2\sum_{i=1}^{d} a_i b_i + 1 \\ &= \sum_{i=1}^{d} a_i^2 b_i^2 + 2\sum_{i=1}^{d}\sum_{j=i+1}^{d} a_i a_j b_i b_j + 2\sum_{i=1}^{d} a_i b_i + 1\end{aligned}$$

Quadratic Dot Product

- Does $\mathbf{\Phi}(\mathbf{a})^T\mathbf{\Phi}(\mathbf{b})$ look familiar?

$$\mathbf{\Phi}(\mathbf{a})^T\mathbf{\Phi}(\mathbf{b}) = 1 + 2\sum_{i=1}^d a_i b_i + \sum_{i=1}^d a_i^2 b_i^2 + \sum_{i=1}^d \sum_{j=i+1}^d 2 a_i a_j b_i b_j$$

- Try this: $(\boldsymbol{a}^T\boldsymbol{b} + 1)^2$

$$\begin{aligned}
(\boldsymbol{a}^T\boldsymbol{b} + 1)^2 &= (\boldsymbol{a}^T\boldsymbol{b})^2 + 2\boldsymbol{a}^T\boldsymbol{b} + 1 \\
&= \left(\sum_{i=1}^d a_i b_i\right)^2 + 2\sum_{i=1}^d a_i b_i + 1 \\
&= \sum_{i=1}^d \sum_{j=1}^d a_i b_i a_j b_j + 2\sum_{i=1}^d a_i b_i + 1 \\
&= \sum_{i=1}^d a_i^2 b_i^2 + 2\sum_{i=1}^d \sum_{j=i+1}^d a_i a_j b_i b_j + 2\sum_{i=1}^d a_i b_i + 1
\end{aligned}$$

- They're the same! And this is only $O(d)$ to compute!

Dual SVM: Non-linear Transformation

$$\max_{\boldsymbol{\alpha} \in \mathbb{R}^N} \sum_{n=1}^{N} \alpha_n - \frac{1}{2} \sum_{m=1}^{N} \sum_{n=1}^{N} \alpha_n \alpha_m y_n y_m \boldsymbol{\Phi}(\mathbf{x}_n)^T \boldsymbol{\Phi}(\mathbf{x}_m)$$

$$\text{subject to } \sum_{n=1}^{N} y_n \alpha_n = 0, \alpha_n \geq 0, \forall n$$

$$\mathbf{w}^* = \sum_{n=1}^{N} y_n \alpha_n^* \Phi(\mathbf{x}_n)$$

- Need to prepare a matrix Q, $Q_{nm} = y_n y_m \boldsymbol{\Phi}(\mathbf{x}_n)^T \boldsymbol{\Phi}(\mathbf{x}_m)$
- Cost?
  - We must do $N^2/2$ dot products to get this matrix ready.
  - Each dot product requires d additions and multiplications.

Higher Order Polynomials

	$\mathbf{\Phi}(\mathbf{x})$	Cost	100dim
Quadratic	$d^2/2$ terms	$d^2N^2/4$	$2.5kN^2$
Cubic	$d^3/6$ terms	$d^3N^2/12$	$83kN^2$
Quartic	$d^4/24$ terms	$d^4N^2/48$	$1.96mN^2$

	$\Phi(\mathbf{a})^T\Phi(\mathbf{b})$	Cost	100dim
Quadratic	$(\mathbf{a}^T\mathbf{b}+1)^2$	$dN^2/2$	$50N^2$
Cubic	$(\mathbf{a}^T\mathbf{b}+1)^3$	$dN^2/2$	$50N^2$
Quartic	$(\mathbf{a}^T\mathbf{b}+1)^4$	$dN^2/2$	$50N^2$

Dual SVM with Quintic Basis Functions

$$\max_{\boldsymbol{\alpha} \in \mathbb{R}^N} \sum_{n=1}^{N} \alpha_n - \frac{1}{2} \sum_{m=1}^{N} \sum_{n=1}^{N} \alpha_n \alpha_m y_n y_m \underbrace{\boldsymbol{\Phi}(\mathbf{x}_n)^T \boldsymbol{\Phi}(\mathbf{x}_m)}_{(\mathbf{x}_n^T \mathbf{x}_m + 1)^5}$$

$$\text{subject to } \sum_{n=1}^{N} y_n \alpha_n = 0, \alpha_n \geq 0, \forall n$$

Classification:

$$g(\mathbf{x}) = \mathsf{sign}(\mathbf{w}^{*T} \boldsymbol{\Phi}(\mathbf{x}) + b^*) = \mathsf{sign}\left(\sum\nolimits_{\alpha_n^* > 0} y_n \alpha_n^* \boldsymbol{\Phi}(\mathbf{x}_n)^T \boldsymbol{\Phi}(\mathbf{x}) + b^* \right)$$

$$= \mathsf{sign}\left(\sum\nolimits_{\alpha_n^* > 0} y_n \alpha_n^* (\mathbf{x}_n^T \mathbf{x} + 1)^5 + b^* \right)$$

Dual SVM with General Kernel Functions

$$\max_{\boldsymbol{\alpha} \in \mathbb{R}^N} \sum_{n=1}^{N} \alpha_n - \frac{1}{2} \sum_{m=1}^{N} \sum_{n=1}^{N} \alpha_n \alpha_m y_n y_m K(\mathbf{x}_n, \mathbf{x}_m)$$

$$\text{subject to } \sum_{n=1}^{N} y_n \alpha_n = 0, \alpha_n \geq 0, \forall n$$

Classification:

$$g(\mathbf{x}) = \text{sign}(\mathbf{w}^{*T} \mathbf{\Phi}(\mathbf{x}) + b^*) = \text{sign}\left(\sum\nolimits_{\alpha_n^* > 0} y_n \alpha_n^* \mathbf{\Phi}(\mathbf{x}_n)^T \mathbf{\Phi}(\mathbf{x}) + b^* \right)$$

$$= \text{sign}\left(\sum\nolimits_{\alpha_n^* > 0} y_n \alpha_n^* K(\mathbf{x}_n, \mathbf{x}_m) + b^* \right)$$

Kernel Tricks

- Replacing dot product with a kernel function

Kernel Tricks

- Replacing dot product with a kernel function
- Not all functions are kernel functions!
  - Need to be decomposable $K(\mathbf{a}, \mathbf{b}) = \mathbf{\Phi}(\mathbf{a})^T \mathbf{\Phi}(\mathbf{b})$
  - Could $K(\mathbf{a}, \mathbf{b}) = (\mathbf{a} - \mathbf{b})^3$ be a kernel function?
  - Could $K(\mathbf{a}, \mathbf{b}) = (\mathbf{a} - \mathbf{b})^4 - (\mathbf{a} + \mathbf{b})^2$ be a kernel function?

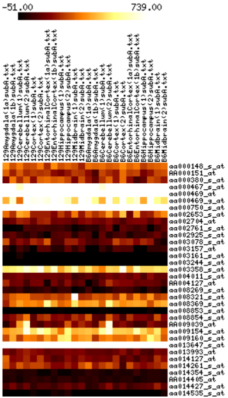
Kernel Tricks

- Replacing dot product with a kernel function
- Not all functions are kernel functions!
  - Need to be decomposable $K(\mathbf{a}, \mathbf{b}) = \mathbf{\Phi}(\mathbf{a})^T \mathbf{\Phi}(\mathbf{b})$
  - Could $K(\mathbf{a}, \mathbf{b}) = (\mathbf{a} - \mathbf{b})^3$ be a kernel function?
  - Could $K(\mathbf{a}, \mathbf{b}) = (\mathbf{a} - \mathbf{b})^4 - (\mathbf{a} + \mathbf{b})^2$ be a kernel function?
- Mercer's condition
  - To expand Kernel function $K(\mathbf{a}, \mathbf{b})$ into a dot product, i.e., $K(\mathbf{a}, \mathbf{b}) = \mathbf{\Phi}(\mathbf{a})^T \mathbf{\Phi}(\mathbf{b})$, $K(\mathbf{a}, \mathbf{b})$ has to be positive semi-definite function.

Kernel Tricks

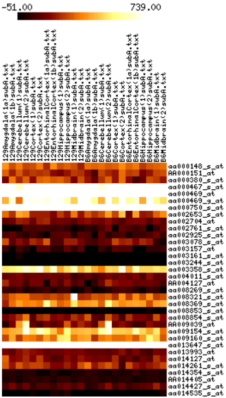
- Replacing dot product with a kernel function
- Not all functions are kernel functions!
  - Need to be decomposable $K(\mathbf{a}, \mathbf{b}) = \mathbf{\Phi}(\mathbf{a})^T \mathbf{\Phi}(\mathbf{b})$
  - Could $K(\mathbf{a}, \mathbf{b}) = (\mathbf{a} - \mathbf{b})^3$ be a kernel function?
  - Could $K(\mathbf{a}, \mathbf{b}) = (\mathbf{a} - \mathbf{b})^4 - (\mathbf{a} + \mathbf{b})^2$ be a kernel function?
- Mercer's condition
  - To expand Kernel function $K(\mathbf{a}, \mathbf{b})$ into a dot product, i.e., $K(\mathbf{a}, \mathbf{b}) = \mathbf{\Phi}(\mathbf{a})^T \mathbf{\Phi}(\mathbf{b})$, $K(\mathbf{a}, \mathbf{b})$ has to be positive semi-definite function.
  - kernel matrix K is always symmetric PSD for any given $\mathbf{x}_1, \ldots, \mathbf{x}_N$.

Kernel Design: Expression Kernel



- mRNA expression data
  - Each matrix entry is an mRNA expression measurement.
  - Each column is an experiment.
  - Each row corresponds to a gene.

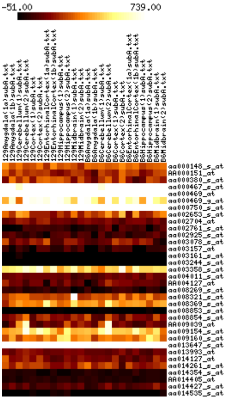
Kernel Design: Expression Kernel



- mRNA expression data
  - Each matrix entry is an mRNA expression measurement.
  - Each column is an experiment.
  - Each row corresponds to a gene.
- Similar or dissimilar

Similar

Dissimilar



- mRNA expression data
  - Each matrix entry is an mRNA expression measurement.
  - Each column is an experiment.
  - Each row corresponds to a gene.
- Similar or dissimilar

Similar

Dissimilar

- Kernel

$$K(x, y) = \frac{\sum_i x_i y_i}{\sqrt{\sum_i x_i x_i}\sqrt{\sum_i y_i y_i}}$$

Kernel Design: Sequence Kernel

- Work with non-vectorial data
- Scalar product on a pair of variable-length, discrete strings?

```
>ICYA_MANSE
GDIFYPGYCPDVKPVNDFDLSAFAGAWHEIAKLPLENENQGKCTIAEYKY
DGKKASVYNSFVSNGVKEYMEGDLEIAPDAKYTKQGKYVMTFKFGQRVVN
LVPWVLATDYKNYAINYMENSHPDKKAHSIHAWILSKSKVLEGNTKEVVD
NVLKTFSHLIDASKFISNDFSEAACQYSTTYSLTGPDRH

>LACB_BOVIN
MKCLLLALALTCGAQALIVTQTMKGLDIQKVAGTWYSLAMAASDISLLDA
QSAPLRVYVEELKPTPEGDLEILLQKWENGECAQKKIIAEKTKIPAVFKI
DALNENKVLVLDTDYKKYLLFCMENSAEPEQSLACQCLVRTPEVDDEALE
KFDKALKALPMHIRLSFNPTQLEEQCHI
```

Commonly Used SVM Kernel Functions

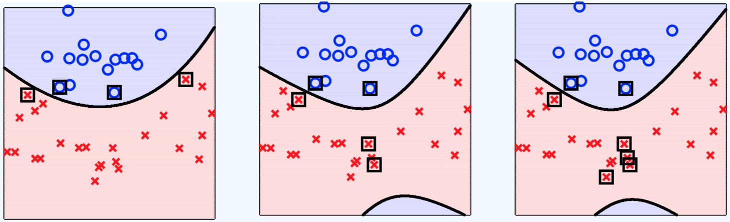
- $K(\mathbf{a}, \mathbf{b}) = (\alpha \cdot \mathbf{a}^T \mathbf{b} + \beta)^Q$ is an example of an SVM kernel function
- Beyond polynomials there are other very high dimensional basis functions that can be made practical by finding the right Kernel Function
    - Radial-basis style kernel (RBF)/Gaussian kernel function

$$K(\mathbf{a}, \mathbf{b}) = \exp\left(-\gamma \|\mathbf{a} - \mathbf{b}\|^2\right)$$

    - Sigmoid functions

2nd Order Polynomial Kernel

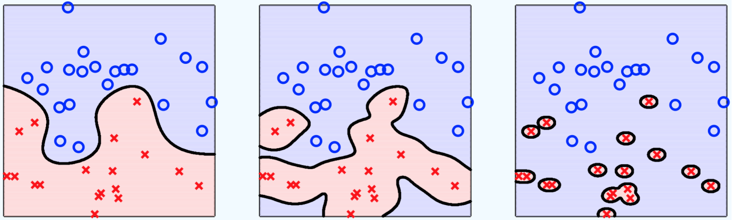
$$K(\mathbf{a},\mathbf{b}) = (\alpha \cdot \mathbf{a}^T\mathbf{b} + \beta)^2$$



$(1 + 0.001\mathbf{x}^{\mathrm{T}}\mathbf{x}')^2$ $\quad$ $1 + (\mathbf{x}^{\mathrm{T}}\mathbf{x}') + (\mathbf{x}^{\mathrm{T}}\mathbf{x}')^2$ $\quad$ $(1 + 1000\mathbf{x}^{\mathrm{T}}\mathbf{x}')^2$

Gaussian Kernels

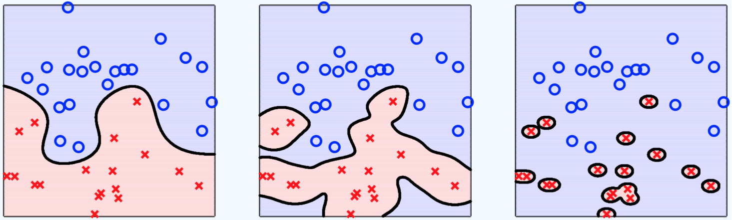
$$K(\mathbf{a}, \mathbf{b}) = \exp\left(-\gamma \|\mathbf{a} - \mathbf{b}\|^2\right)$$



$\exp(-1\|\mathbf{x} - \mathbf{x}'\|^2)$ $\exp(-10\|\mathbf{x} - \mathbf{x}'\|^2)$ $\exp(-100\|\mathbf{x} - \mathbf{x}'\|^2)$

Gaussian Kernels

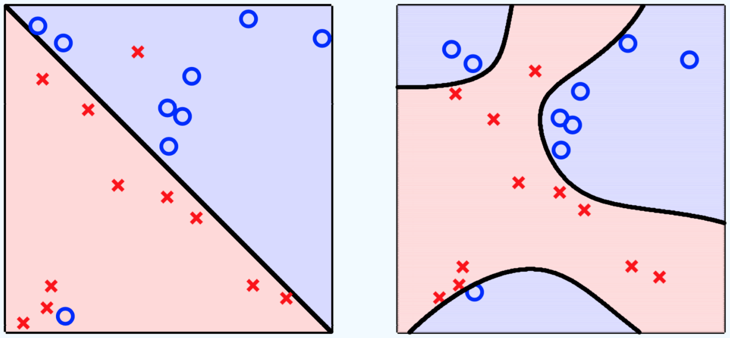
$$K(\mathbf{a}, \mathbf{b}) = \exp\left(-\gamma \|\mathbf{a} - \mathbf{b}\|^2\right)$$



$\exp(-1\|\mathbf{x} - \mathbf{x}'\|^2)$ $\exp(-10\|\mathbf{x} - \mathbf{x}'\|^2)$ $\exp(-100\|\mathbf{x} - \mathbf{x}'\|^2)$

When γ is large, we clearly see that even the protection of a large margin cannot suppress overfitting. However, for a reasonably small γ, the sophisticated boundary discovered by SVM with the Gaussian-RBF kernel looks quite good.

Gaussian Kernels



(a) linear classifier (b) Gaussian-RBF kernel

For (a) a noisy dataset the linear classifier appears to work quite well, and (b) using the Gaussian-RBF kernel with the hard-margin SVM leads to overfitting.

From Hard-margin to Soft-margin

- When there are outliers, hard margin SVM + Gaussian-RBF kernel result in an unnecessarily complicated decision boundary that overfits the training noise.

From Hard-margin to Soft-margin

- When there are outliers, hard margin SVM + Gaussian-RBF kernel result in an unnecessarily complicated decision boundary that overfits the training noise.
- Remedy: a soft formulation that allows small violation of the margins or even some classification errors.

From Hard-margin to Soft-margin

- When there are outliers, hard margin SVM + Gaussian-RBF kernel result in an unnecessarily complicated decision boundary that overfits the training noise.
- Remedy: a soft formulation that allows small violation of the margins or even some classification errors.
- Soft-margin: margin violation $\varepsilon_n \geq 0$ for each data point $(\mathbf{x}_n, y_n)$ and require that

$$y_n(\mathbf{w}^T \mathbf{x}_n + b) \geq 1 - \varepsilon_n$$

  - ε_n captures by how much $(\mathbf{x}_n, y_n)$ fails to be separated.

Soft-Margin SVM

We modify the hard-margin SVM to the soft-margin SVM by allowing margin violations but adding a penalty term to discourage large violations:

$$\min_{b,\mathbf{w},\boldsymbol{\varepsilon}} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{n=1}^{N}\varepsilon_n$$

$$\text{subject to: } y_n(\mathbf{w}^T\mathbf{x}_n + b) \geq 1 - \varepsilon_n \text{ for } n = 1, \ldots, N$$

$$\varepsilon_n \geq 0, \text{ for } n = 1, \ldots, N$$

The meaning of C?

Soft-Margin SVM

We modify the hard-margin SVM to the soft-margin SVM by allowing margin violations but adding a penalty term to discourage large violations:

$$\min_{b, \mathbf{w}, \boldsymbol{\varepsilon}} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{n=1}^{N}\varepsilon_n$$

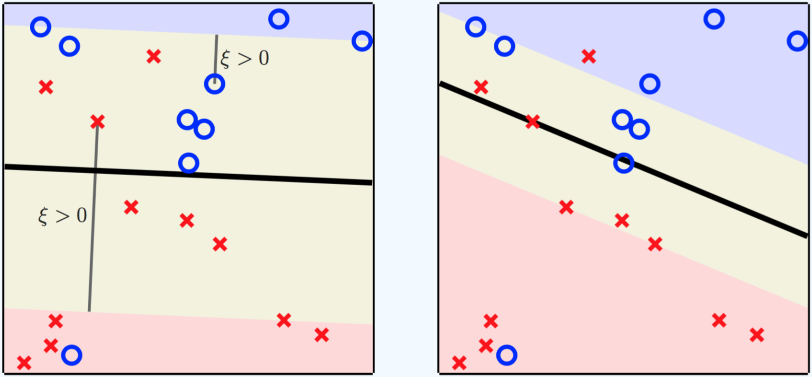
$$\text{subject to: } y_n(\mathbf{w}^T\mathbf{x}_n + b) \geq 1 - \varepsilon_n \text{ for } n = 1, \ldots, N$$

$$\varepsilon_n \geq 0, \text{ for } n = 1, \ldots, N$$

The meaning of C?

- When C is large, it means we care more about violating the margin, which gets us closer to the hard-margin SVM.
- When C is small, on the other hand, we care less about violating the margin.

Soft Margin Example

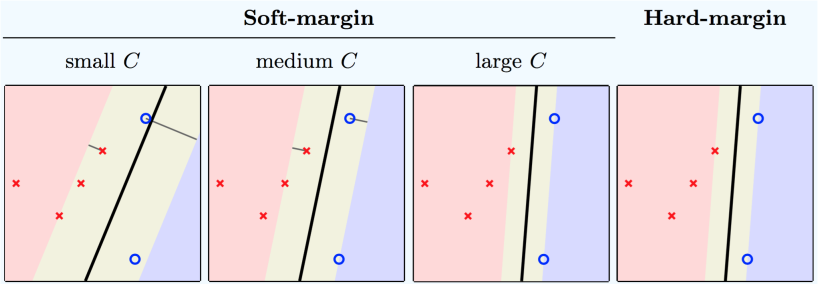


(a) $C = 1$ (b) $C = 500$

Soft Margin and Hard Margin

$$\min_{b,\mathbf{w},\boldsymbol{\varepsilon}} \quad \underbrace{\frac{1}{2}\mathbf{w}^T\mathbf{w}}_{\text{margin}} + \underbrace{C\sum_{n=1}^{N}\varepsilon_n}_{\text{error tolerance}}$$

subject to: $y_n(\mathbf{w}^T\mathbf{x}_n + b) \geq 1 - \varepsilon_n, \varepsilon_n \geq 0, \forall N$



The Hinge Loss

- The trade-off sounds very similar, right?

The Hinge Loss

- The trade-off sounds very similar, right?
- We have $\varepsilon_n \geq 0$, and that $y_n(\mathbf{w}^T\mathbf{x}_n + b) \geq 1 - \varepsilon_n \Rightarrow \varepsilon_n \geq 1 - y_n(\mathbf{w}^T\mathbf{x}_n + b)$

The Hinge Loss

- The trade-off sounds very similar, right?
- We have $\varepsilon_n \geq 0$, and that $y_n(\mathbf{w}^T\mathbf{x}_n + b) \geq 1 - \varepsilon_n \Rightarrow \varepsilon_n \geq 1 - y_n(\mathbf{w}^T\mathbf{x}_n + b)$
- The SVM loss (aka. Hinge Loss) function

$$E_{\mathsf{SVM}}(b, \mathbf{w}) = \frac{1}{N}\sum_{n=1}^{N} \max(1 - y_n(\mathbf{w}^T\mathbf{x}_n + b), 0)$$

The Hinge Loss

- The trade-off sounds very similar, right?
- We have $\varepsilon_n \geq 0$, and that $y_n(\mathbf{w}^T\mathbf{x}_n + b) \geq 1 - \varepsilon_n \Rightarrow \varepsilon_n \geq 1 - y_n(\mathbf{w}^T\mathbf{x}_n + b)$
- The SVM loss (aka. Hinge Loss) function

$$E_{\mathsf{SVM}}(b, \mathbf{w}) = \frac{1}{N}\sum_{n=1}^{N} \max(1 - y_n(\mathbf{w}^T\mathbf{x}_n + b), 0)$$

- The soft-margin SVM can be re-written as the following optimization problem:

$$\min_{b,\mathbf{w}} E_{\mathsf{SVM}}(b, \mathbf{w}) + \lambda\mathbf{w}^T\mathbf{w}$$

Dual Soft-Margin SVM

$$\max_{\boldsymbol{\alpha}\in\mathbb{R}^N} \sum_{n=1}^{N} \alpha_n - \frac{1}{2}\sum_{m=1}^{N}\sum_{n=1}^{N} \alpha_n\alpha_m y_n y_m \mathbf{x}_n^T \mathbf{x}_m$$

$$\text{subject to } \sum_{n=1}^{N} y_n\alpha_n = 0, 0 \le \alpha_n \le C, \forall n$$

$$\mathbf{w}^* = \sum_{n=1}^{N} y_n \alpha_n^* \mathbf{x}_n$$

Summary of Dual SVM

- Deliver a large-margin hyperplane, and in so doing it can control the effective model complexity.
- Deal with high- or infinite-dimensional transforms using the kernel trick.
- Express the final hypothesis $g(\mathbf{x})$ using only a few support vectors, their corresponding dual variables (Lagrange multipliers), and the kernel.
- Control the sensitivity to outliers and regularize the solution through setting C appropriately.

Support Vector Machine