STOR566: Introduction to Deep Learning

Lecture 5: Kernel Methods

Yao Li
UNC Chapel Hill

Sept 5, 2024

Materials are from *Learning from data (Caltech)* and *Deep Learning (UCLA)*

Outline

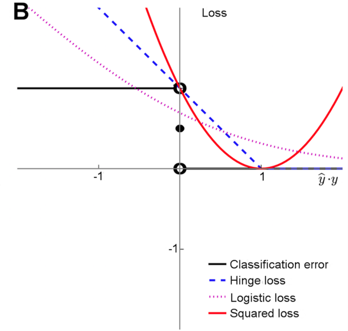
- Linear Support Vector Machines
- Nonlinear SVM, Kernel methods

Support Vector Machines

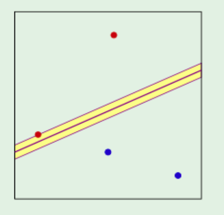
- Given training examples $(\boldsymbol{x}_1, y_1), \cdots, (\boldsymbol{x}_N, y_N)$
  Consider binary classification: $y_i \in \{+1, -1\}$
- Linear Support Vector Machine (SVM):

$$\arg\min_{\boldsymbol{w}} \frac{C}{N} \sum_{i=1}^{N} \max(1 - y_i \boldsymbol{w}^T \boldsymbol{x}_i, 0) + \frac{1}{2} \boldsymbol{w}^T \boldsymbol{w}$$

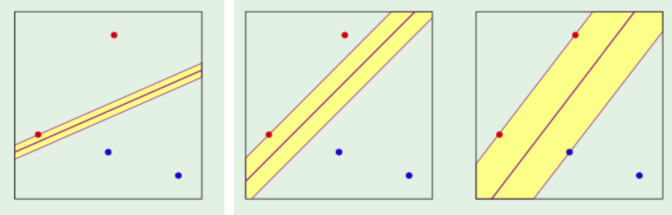
(hinge loss with L2 regularization)



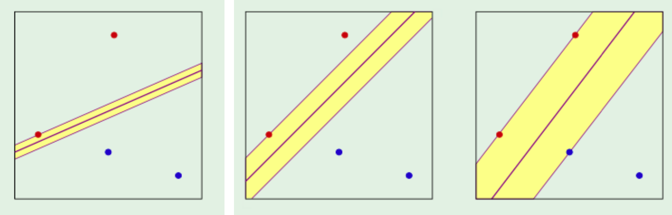
Linear Separation



Linear Separation

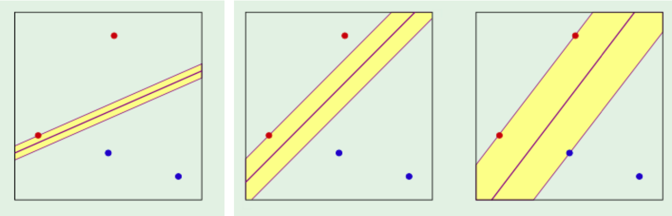


Linear Separation



- Which line is the best?

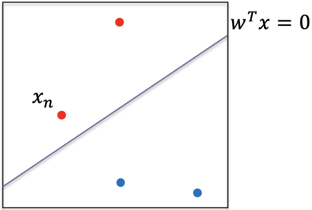
Linear Separation



- Which line is the best?
- Why big margin?
- Which $\boldsymbol{w}$ maximizes the margin?

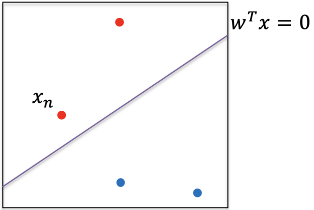
Margin

- $\boldsymbol{w}^T \boldsymbol{x} = 0$: the separation line or hyperplane
- $\boldsymbol{x}_n$: the nearest data point to the plane



Margin

- $\boldsymbol{w}^T\boldsymbol{x} = 0$: the separation line or hyperplane
- $\boldsymbol{x}_n$: the nearest data point to the plane



Preliminary:

- Normalize $\boldsymbol{w}$:

$$\|\boldsymbol{w}^T\boldsymbol{x}_n\| = 1$$

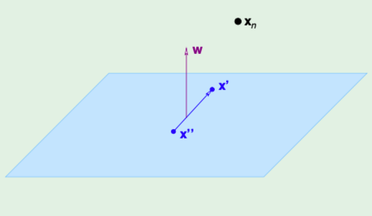
Distance

- The distance between $\mathbf{x}_n$ and the plane $\mathbf{w}^T\mathbf{x} = 0$.
- The vector $\mathbf{w}$ is orthogonal to the plane in the $\mathcal{X}$ space

  Take $\mathbf{x}^{'}$ and $\mathbf{x}^{''}$ on the plane

  $\mathbf{w}^T\mathbf{x}^{'} = 0$, $\mathbf{w}^T\mathbf{x}^{''} = 0$

  $\Longrightarrow \mathbf{w}^T(\mathbf{x}^{'} - \mathbf{x}^{''}) = 0$



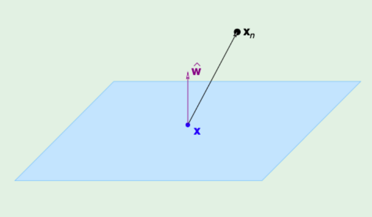
Distance

- The distance between $\boldsymbol{x}_n$ and the plane $\boldsymbol{w}^T\boldsymbol{x} = 0$:

  Take any point $\boldsymbol{x}$ on the plane

  Project $\boldsymbol{x}_n - \boldsymbol{x}$ on $\boldsymbol{w}$:

$$\text{distance} = \frac{1}{\|\boldsymbol{w}\|}|\boldsymbol{w}^T(\boldsymbol{x}_n - \boldsymbol{x})| = \frac{1}{\|\boldsymbol{w}\|}|\boldsymbol{w}^T\boldsymbol{x}_n - \boldsymbol{w}^T\boldsymbol{x}| = \frac{1}{\|\boldsymbol{w}\|}$$



Optimization Problem

- The optimization problem for SVM:

$$\max_{\boldsymbol{w}} \frac{1}{\|\boldsymbol{w}\|}$$
$$\text{s.t.} \quad \min_{i=1,..,N} |\boldsymbol{w}^T \boldsymbol{x}_i| = 1,$$

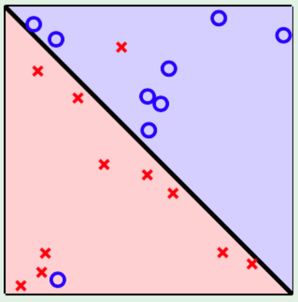
- Equivalent to:

$$\min_{\boldsymbol{w}} \frac{1}{2}\boldsymbol{w}^T \boldsymbol{w}$$
$$\text{s.t.} \quad y_i \boldsymbol{w}^T \boldsymbol{x}_i \geq 1, i = 1, \ldots, N,$$

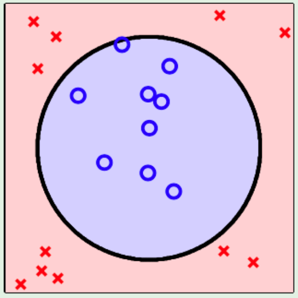
Notice: $|\boldsymbol{w}^T \boldsymbol{x}_i| = y_i \boldsymbol{w}^T \boldsymbol{x}_i$

Two Types of Non-separable

slightly:



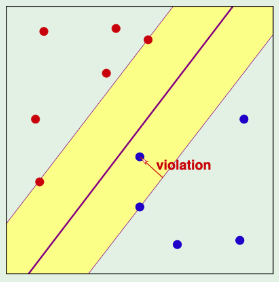
seriously:



Support Vector Machines (Soft)

- Given training data $\boldsymbol{x}_1, \cdots, \boldsymbol{x}_N \in \mathbb{R}^d$ with labels $y_i \in \{+1, -1\}$.
- SVM problem with soft constraints:

$$
\begin{aligned}
\min_{\boldsymbol{w}, \xi} \ & \frac{1}{2} \boldsymbol{w}^T \boldsymbol{w} + \frac{C}{N} \sum_{i=1}^{N} \xi_i \\
\text{s.t.} \ & y_i \boldsymbol{w}^T \boldsymbol{x}_i \geq 1 - \xi_i, \\
& \xi_i \geq 0, i = 1, \ldots, N
\end{aligned}
$$



Support Vector Machines

- SVM problem:

$$\min_{\boldsymbol{w},\xi} \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} + \frac{C}{N}\sum_{i=1}^{N}\xi_i$$
$$\text{s.t. } y_i(\boldsymbol{w}^T\boldsymbol{x}_i) \geq 1 - \xi_i,$$
$$\xi_i \geq 0, i = 1, \ldots, N.$$

- Equivalent to

$$\min_{\boldsymbol{w}} \underbrace{\frac{1}{2}\boldsymbol{w}^T\boldsymbol{w}}_{\text{L2 regularization}} + \frac{C}{N}\sum_{i=1}^{N}\underbrace{\max(0, 1 - y_i\boldsymbol{w}^T\boldsymbol{x}_i)}_{\text{hinge loss}}$$

SVM: Unconstrained

- Unconstrained optimization:

$$\min_{\boldsymbol{w}} \quad \underbrace{\frac{1}{2}\boldsymbol{w}^T\boldsymbol{w}}_{\text{L2 regularization}} + \frac{C}{N}\sum_{i=1}^{N}\underbrace{\max(0, 1 - y_i\boldsymbol{w}^T\boldsymbol{x}_i)}_{\text{hinge loss}}$$

- Equivalent to:

$$\min_{\boldsymbol{w},\xi} \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} + \frac{C}{N}\sum_{i=1}^{N}\xi_i$$
$$\text{s.t. } \xi_i \geq \max(0, 1 - y_i\boldsymbol{w}^T\boldsymbol{x}_i), i = 1, \ldots, N.$$

SVM: Unconstrained

- Unconstrained optimization:

$$\min_{\boldsymbol{w}} \quad \underbrace{\frac{1}{2}\boldsymbol{w}^T\boldsymbol{w}}_{\text{L2 regularization}} \quad + \frac{C}{N}\sum_{i=1}^{N}\underbrace{\max(0, 1 - y_i\boldsymbol{w}^T\boldsymbol{x}_i)}_{\text{hinge loss}}$$

- Equivalent to:

$$\min_{\boldsymbol{w},\xi} \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} + \frac{C}{N}\sum_{i=1}^{N}\xi_i$$
$$\text{s.t. } \xi_i \geq \max(0, 1 - y_i\boldsymbol{w}^T\boldsymbol{x}_i), i = 1, \dots, N.$$

- Equivalent to:

$$\min_{\boldsymbol{w},\xi} \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} + \frac{C}{N}\sum_{i=1}^{N}\xi_i$$
$$\text{s.t. } \xi_i, \geq 1 - y_i\boldsymbol{w}^T\boldsymbol{x}_i, \xi_i \geq 0, i = 1, \dots, N.$$

Stochastic Subgradient Method for SVM

$$\min_{\boldsymbol{w}} \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} + \frac{C}{N}\sum_{i=1}^{N}\max(0, 1 - y_i\boldsymbol{w}^T\boldsymbol{x}_i)$$

- A subgradient of $\max(0, 1 - y_i\boldsymbol{w}^T\boldsymbol{x}_i)$:

$$\begin{cases} -y_i\boldsymbol{x}_i & \text{if } 1 - y_i\boldsymbol{w}^T\boldsymbol{x}_i > 0 \\ \mathbf{0} & \text{if } 1 - y_i\boldsymbol{w}^T\boldsymbol{x}_i < 0 \\ \mathbf{0} & \text{if } 1 - y_i\boldsymbol{w}^T\boldsymbol{x}_i = 0 \end{cases}$$

- Stochastic Subgradient descent for SVM:

  For $t = 1, 2, \ldots$

  Randomly pick an index i

  If $y_i\boldsymbol{w}^T\boldsymbol{x}_i < 1$, then

  $\boldsymbol{w} \leftarrow \boldsymbol{w} - \eta_t(\boldsymbol{w} - Cy_i\boldsymbol{x}_i)$

  Else (if $y_i\boldsymbol{w}^T\boldsymbol{x}_i \geq 1$):

  $\boldsymbol{w} \leftarrow \boldsymbol{w} - \eta_t\boldsymbol{w}$

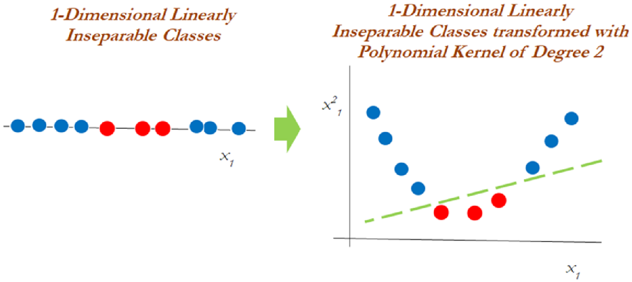
Kernel SVM

Non-linearly separable problems

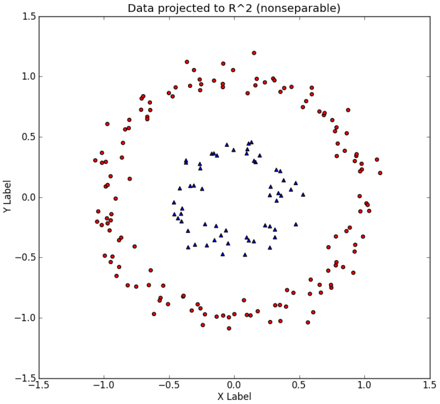
- What if the data is not linearly separable?
  Solution: map data $\boldsymbol{x}_i$ to higher dimensional(maybe infinite) feature space $\varphi(x_i)$, where they are linearly separable.
- Example: $\varphi(x) = (x, x^2)^T$

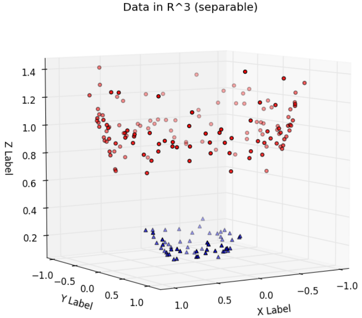
Non-linearly separable problems

- What if the data is not linearly separable?
  Solution: map data $\boldsymbol{x}_i$ to higher dimensional(maybe infinite) feature space $\varphi(x_i)$, where they are linearly separable.
- Example: $\varphi(x) = (x, x^2)^T$



Non-linearly separable problems





- $\varphi(\boldsymbol{x}) = \varphi\left(\begin{pmatrix} x_1 \\ x_2 \end{pmatrix}\right) = \begin{pmatrix} x_1 \\ x_2 \\ x_1^2 + x_2^2 \end{pmatrix}$

SVM with nonlinear mapping

- SVM with nonlinear mapping $\varphi(\cdot)$:

$$
\begin{aligned}
&\min_{\boldsymbol{w},\xi} \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} + \frac{C}{N}\sum_{i=1}^{n}\xi_i \\
&\text{s.t. } y_i(\boldsymbol{w}^T\varphi(\boldsymbol{x}_i)) \geq 1 - \xi_i, \quad \xi_i \geq 0, \; i = 1, \ldots, n,
\end{aligned}
$$

- Hard to solve if $\varphi(\cdot)$ maps to very high or infinite dimensional space

Support Vector Machines

- Primal problem:

$$\min_{\boldsymbol{w},\boldsymbol{\xi}} \frac{1}{2}\|\boldsymbol{w}\|^2 + \frac{C}{N}\sum_i \xi_i$$

$$\text{s.t. } y_i \boldsymbol{w}^T \varphi(\boldsymbol{x}_i) - 1 + \xi_i \geq 0, \text{ and } \xi_i \geq 0 \quad \forall i = 1, \ldots, N$$

- Convex objective and linear constraints
- Equivalent to (Dual problem):

$$\max_{\boldsymbol{\alpha}\geq 0, \boldsymbol{\beta}\geq 0} \min_{\boldsymbol{w},\boldsymbol{\xi}} \frac{1}{2}\|\boldsymbol{w}\|^2 + \frac{C}{N}\sum_i \xi_i - \sum_i \alpha_i (y_i \boldsymbol{w}^T \varphi(\boldsymbol{x}_i) - 1 + \xi_i) - \sum_i \beta_i \xi_i$$

Support Vector Machines (dual)

- Reorganize the equation:

$$\max_{\boldsymbol{\alpha}\geq 0, \boldsymbol{\beta}\geq 0} \min_{\boldsymbol{w}, \boldsymbol{\xi}} \frac{1}{2}\|\boldsymbol{w}\|^2 - \sum_i \alpha_i y_i \boldsymbol{w}^T \varphi(\boldsymbol{x}_i) + \sum_i \xi_i (\frac{C}{N} - \alpha_i - \beta_i) + \sum_i \alpha_i$$

- By KKT, for any given $\boldsymbol{\alpha}, \boldsymbol{\beta}$, the minimizer will satisfy

$$\frac{\partial L}{\partial \boldsymbol{w}} = \boldsymbol{w} - \sum_i \alpha_i y_i \varphi(\boldsymbol{x}_i) = 0 \quad \Rightarrow \boldsymbol{w}^* = \sum_i y_i \alpha_i \varphi(\boldsymbol{x}_i)$$

$$\frac{\partial L}{\partial \boldsymbol{\xi}} = (\frac{C}{N} - \alpha_1 - \beta_1, \ldots, \frac{C}{N} - \alpha_N - \beta_N)^T = 0 \quad \Rightarrow \frac{C}{N} = \alpha_i + \beta_i, \forall i$$

- Substitue these two equations back we get

$$\max_{\boldsymbol{\alpha}\geq 0, \boldsymbol{\beta}\geq 0, \frac{C}{N} = \boldsymbol{\alpha}+\boldsymbol{\beta}} -\frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j \varphi(\boldsymbol{x}_i)^T \varphi(\boldsymbol{x}_j) + \sum_i \alpha_i$$

Support Vector Machines (dual)

- Therefore, we get the following dual problem

$$\max_{0 \leq \boldsymbol{\alpha} \leq \frac{C}{N}} \{-\frac{1}{2}\boldsymbol{\alpha}^T Q \boldsymbol{\alpha} + \mathbf{1}^T \boldsymbol{\alpha}\} := D(\boldsymbol{\alpha}),$$

  where Q is an N by N matrix with $Q_{ij} = y_i y_j \varphi(\boldsymbol{x}_i)^T \varphi(\boldsymbol{x}_j)$

- Based on the derivations,
  1. We can solve the dual problem instead of the primal problem.
  2. Let $\boldsymbol{\alpha}^*$ be the dual solution and $\boldsymbol{w}^*$ be the primal solution, we have

$$\boldsymbol{w}^* = \sum_i y_i \alpha_i^* \varphi(\boldsymbol{x}_i)$$

- To solve the dual, we only need to know $\varphi(\boldsymbol{x}_i)^T \varphi(\boldsymbol{x}_j)$.

Kernel Trick

- Do not directly define $\varphi(\cdot)$
- Instead, define "kernel"

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \varphi(\boldsymbol{x}_i)^T \varphi(\boldsymbol{x}_j)$$

  This is all we need to know for Kernel SVM!
- Examples:
  - Gaussian kernel: $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = e^{-\gamma \|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}$
  - Polynomial kernel: $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = (\gamma \boldsymbol{x}_i^T \boldsymbol{x}_j + c)^d$

The Trick

- Can we compute $K(\boldsymbol{x}, \boldsymbol{x}^{'})$ without transforming $\boldsymbol{x}$ and $\boldsymbol{x}^{'}$?
- Example:
  Consider a transformation $\varphi(\boldsymbol{x}) = (1, x_1^2, x_2^2, \sqrt{2}x_1, \sqrt{2}x_2, \sqrt{2}x_1x_2)$
  The inner product between $\varphi(\boldsymbol{x})$ and $\varphi(\boldsymbol{x}^{'})$

$$K(\boldsymbol{x}, \boldsymbol{x}^{'}) = \varphi(\boldsymbol{x})^T \varphi(\boldsymbol{x}^{'}) = (1 + \boldsymbol{x}^T \boldsymbol{x}^{'})^2$$

- Computation cost: $(1 + \boldsymbol{x}^T \boldsymbol{x}^{'})^2$ vs. $\varphi(\boldsymbol{x})^T \varphi(\boldsymbol{x}^{'})$

The Trick

- Can we compute $K(\boldsymbol{x}, \boldsymbol{x}^{'})$ without transforming $\boldsymbol{x}$ and $\boldsymbol{x}^{'}$?
- Example:

  Consider a transformation $\varphi(\boldsymbol{x}) = (1, x_1^2, x_2^2, \sqrt{2}x_1, \sqrt{2}x_2, \sqrt{2}x_1x_2)$

  The inner product between $\varphi(\boldsymbol{x})$ and $\varphi(\boldsymbol{x}^{'})$

$$K(\boldsymbol{x}, \boldsymbol{x}^{'}) = \varphi(\boldsymbol{x})^T\varphi(\boldsymbol{x}^{'}) = (1 + \boldsymbol{x}^T\boldsymbol{x}^{'})^2$$

- Computation cost: $(1 + \boldsymbol{x}^T\boldsymbol{x}^{'})^2$ vs. $\varphi(\boldsymbol{x})^T\varphi(\boldsymbol{x}^{'})$
- Example: Simple one-dimensional Gaussian kernel maps to infinite-dimensional space

$$\begin{aligned} K(x_i, x_j) &= \exp(-\frac{1}{2}(x_i - x_j)^2) \\ &= \exp(-\frac{1}{2}x_i^2)\exp(-\frac{1}{2}x_j^2)\sum_{k=0}^{\infty}\frac{x_i^k x_j^k}{k!} \end{aligned}$$

Solution

- Training: compute $\boldsymbol{\alpha} = [\alpha_1, \cdots, \alpha_N]$ by solving the quadratic optimization problem:

$$\min_{0 \le \boldsymbol{\alpha} \le C} \frac{1}{2} \boldsymbol{\alpha}^T Q \boldsymbol{\alpha} - \mathbf{1}^T \boldsymbol{\alpha}$$

where $Q_{ij} = K(\boldsymbol{x}_i, \boldsymbol{x}_j)$

Solution

- Training: compute $\boldsymbol{\alpha} = [\alpha_1, \cdots, \alpha_N]$ by solving the quadratic optimization problem:

$$\min_{0 \le \boldsymbol{\alpha} \le C} \frac{1}{2} \boldsymbol{\alpha}^T Q \boldsymbol{\alpha} - \mathbf{1}^T \boldsymbol{\alpha}$$

  where $Q_{ij} = K(\boldsymbol{x}_i, \boldsymbol{x}_j)$

- Prediction: for a test data $\boldsymbol{x}$,

$$\begin{aligned} \boldsymbol{w}^T \varphi(\boldsymbol{x}) &= \sum_{i=1}^{N} y_i \alpha_i \varphi(\boldsymbol{x}_i)^T \varphi(\boldsymbol{x}) \\ &= \sum_{i=1}^{N} y_i \alpha_i K(\boldsymbol{x}_i, \boldsymbol{x}) \end{aligned}$$

Conclusions

- SVM, Kernel SVM

Questions?