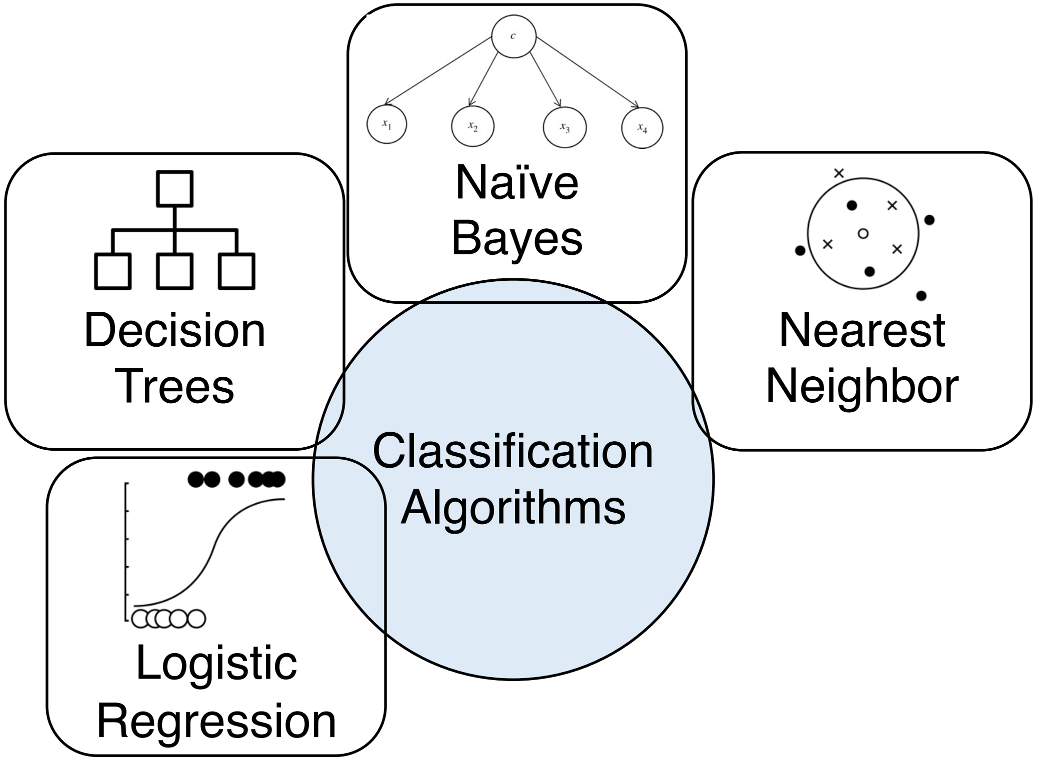
Machine Learning

Lecture 7: Support Vector Machine

Fall 2023

Instructor: Xiaodong Gu

The family of classification

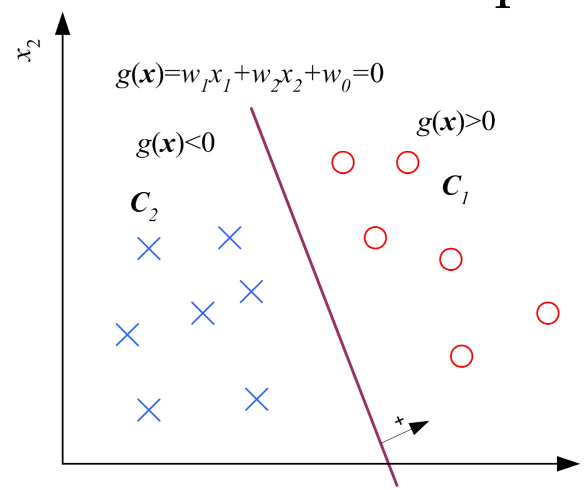


Recall: Linear Classifiers

- **Given**: a training set $\{(x^{(1)},y^{(1)}), \ldots, (x^{(N)}, y^{(N)})\}$ of N samples

  $x^{(\ell)} \in \mathbf{R}^d$: input, $y^{(\ell)} \in \{-1, 1\}$:target (label)

- Assume that the problem is linearly separable: there exists a linear surface to separate the two classes



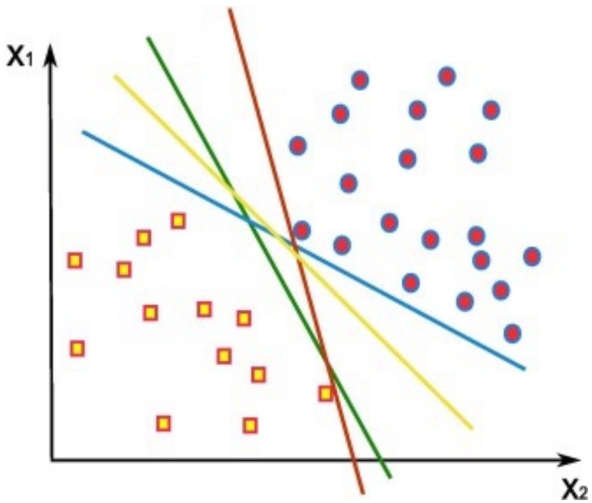
2-D: line;
3-D: plane;
…;
d-D: hyperplane ($\boldsymbol{wx}+w_0=0$)

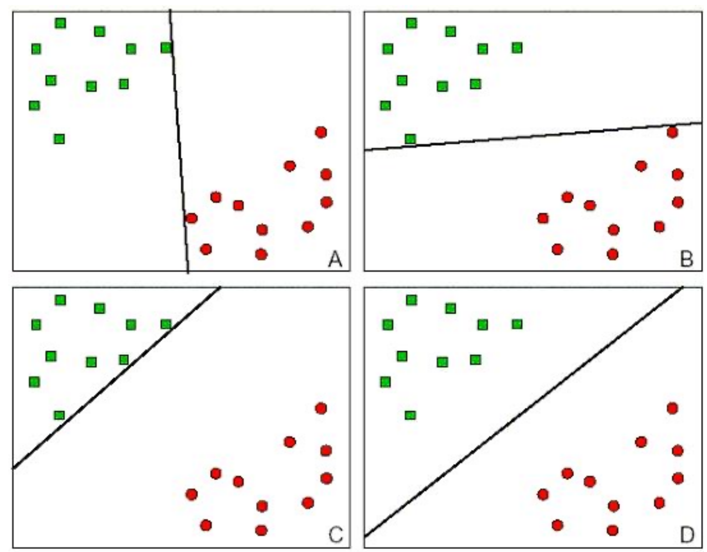
Goal:

Find $\boldsymbol{wx}+w_0=0$ that perfectly separates the two classes

The Problem

- There can be multiple separating hyperplanes.

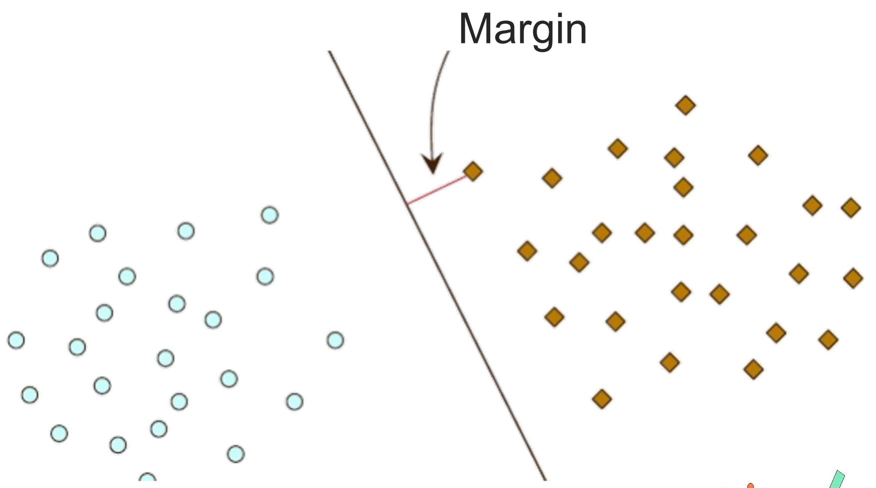




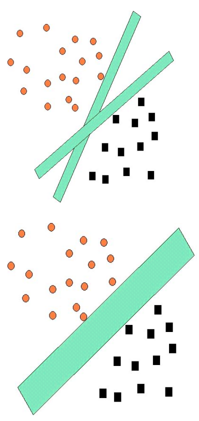
Which one is the best?

The Idea:

- Find a decision boundary that **maximizes the margin** between two classes.



Margin and generalization:
Statistical learning theories have shown that the boundary with the largest margin generalizes best (i.e., has the smallest generalization error).



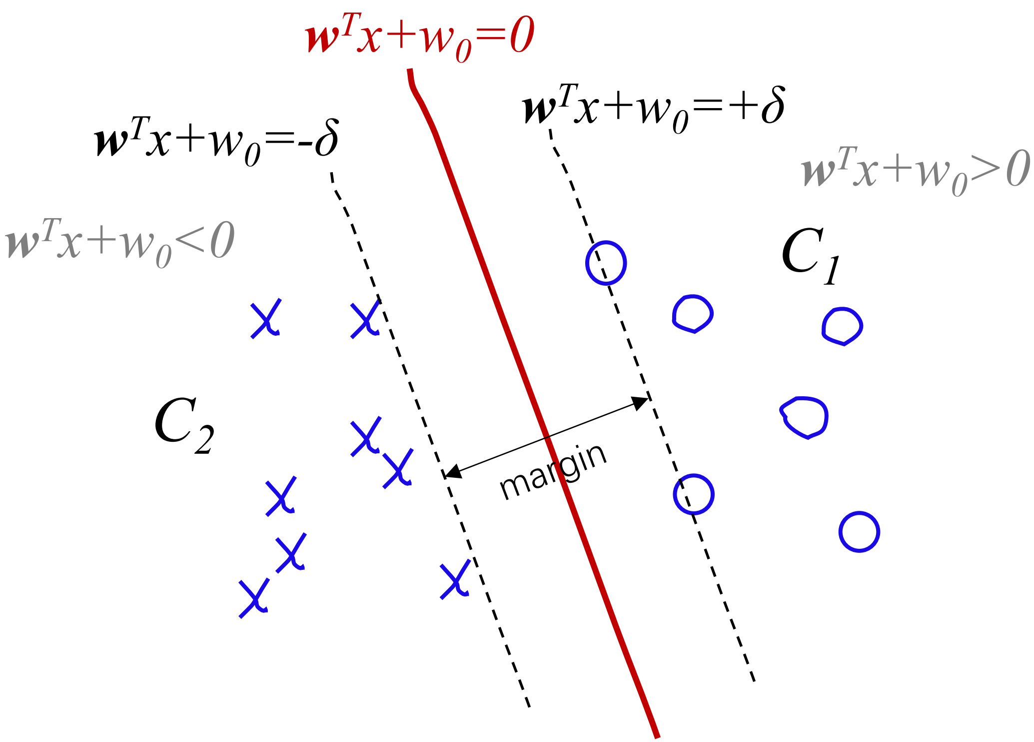
Today

Support Vector Machine

- Problem Formulation
- Dual Problem
- Loss and Optimization

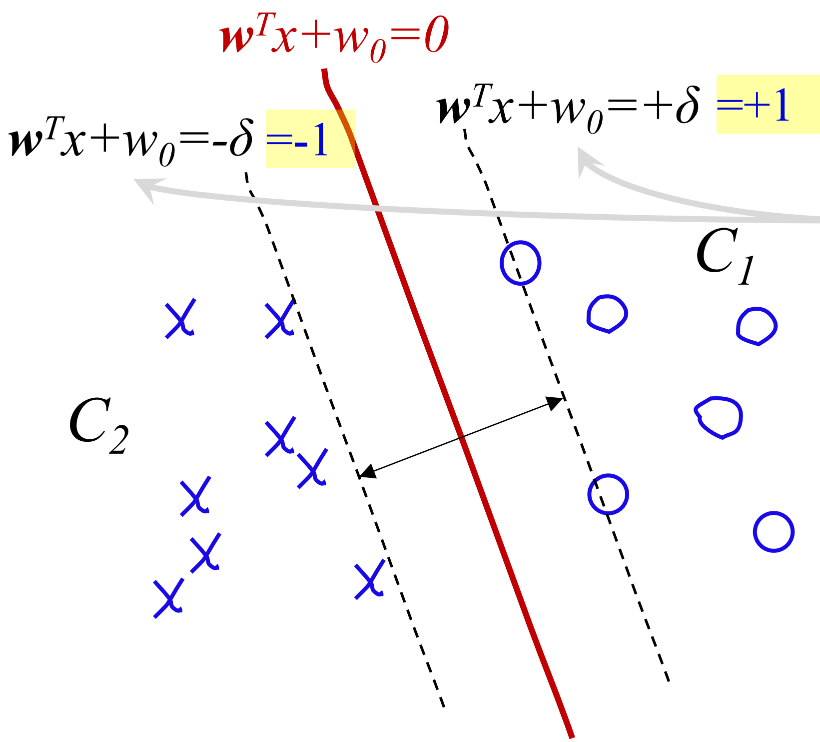
Problem Formulation

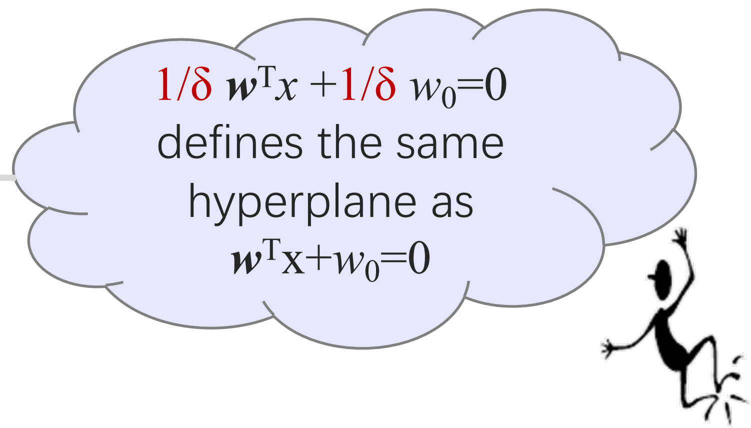
- A linear discriminant function models a **linear decision boundary** of two classes.



Problem Formulation

- A linear discriminant function models a **linear decision boundary** of two classes.





$$w^T x + w_0 \begin{cases} \geq +1 & x \in C_1 \\ \leq -1 & x \in C_2 \end{cases}$$

Maximizing the Margin

$$\boldsymbol{w}^{\mathrm{T}}x+w_0 = \begin{cases} 1 & \text{for the closest points on one side} \\ -1 & \text{for the closest points on the other} \end{cases}$$

- Let $x^{(1)}$ and $x^{(2)}$ be two closest points on each side of the hyperplane.
- Note that

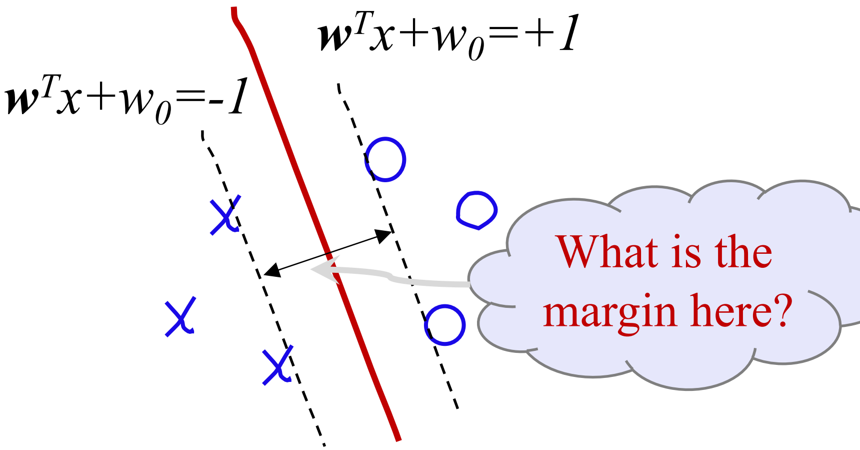
$$\boldsymbol{w}^{\mathrm{T}}\, x^{(1)} + w_0 = +1$$
$$\boldsymbol{w}^{\mathrm{T}}\, x^{(2)} + w_0 = -1$$

which imply

$$\boldsymbol{w}^{\mathrm{T}}(x^{(1)} - x^{(2)}) = 2.$$

Hence, the margin can be given by

$$\text{margin} = \frac{\boldsymbol{w}^{\mathrm{T}}(x^{(1)}-x^{(2)})}{\|\boldsymbol{w}\|} = \frac{2}{\|\boldsymbol{w}\|}$$

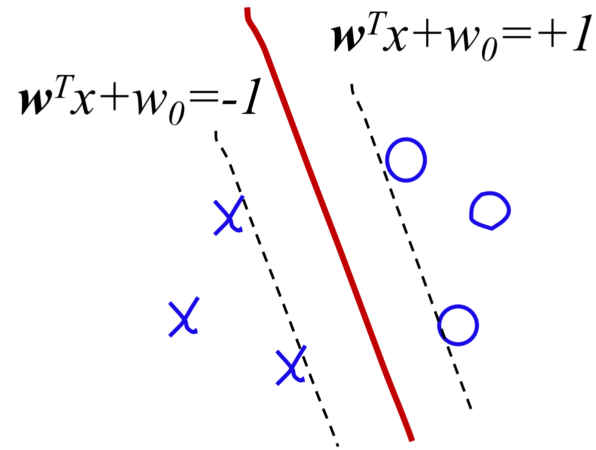


Maximizing the margin is equivalent to minimizing $\frac{1}{2}\|\boldsymbol{w}\|$

Inequality Constraints

- Given: $D = \{(x^{(1)}, y^{(1)}), \ldots, (x^{(N)}, y^{(N)})\}$, we want $\boldsymbol{w}$ and w_0 to satisfy

$$\boldsymbol{w}^{\mathrm{T}} x^{(\ell)} + w_0 \begin{cases} \geq +1 & \text{if } y^{(\ell)} = +1 \\ \leq -1 & \text{if } y^{(\ell)} = -1 \end{cases}$$



- Or, equivalently,

$$y^{(\ell)} \left(\boldsymbol{w}^{\mathrm{T}} x^{(\ell)} + w_0\right) \geq 1$$

SVM = Solving an Optimization Problem

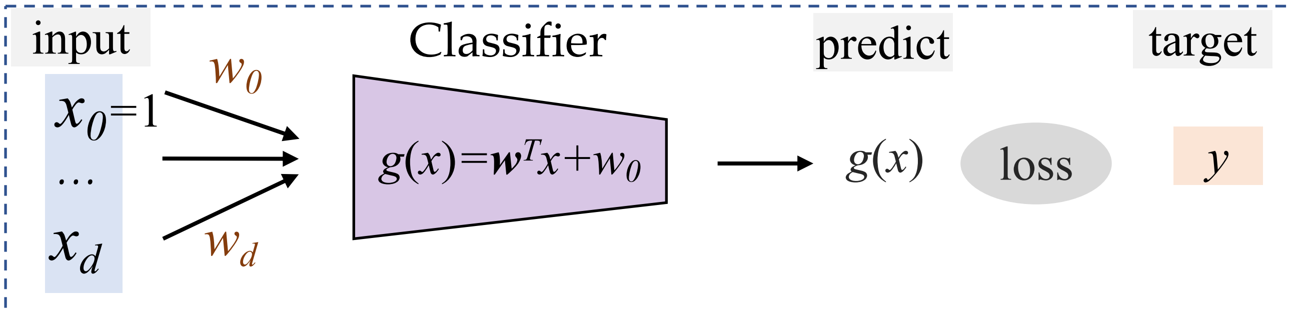
In summary, SVM aims to solve a constrained optimization problem:

$$\begin{aligned} &\text{minimize} && \frac{1}{2}||\boldsymbol{w}||^2 \\ &\text{subject to} && y^{(\ell)}(\boldsymbol{w}^{\mathrm{T}}x^{(\ell)}+w_0)\geq 1,\ \ell=1,\ldots,\ \mathrm{N} \end{aligned}$$

- This is a quadratic programming (QP) problem, which is one type of convex optimization problem.
- The complexity depends on the dimensionality d of inputs

Model Architecture

- The same as Perceptron.



- Train:
  - optimize the parameters $\boldsymbol{w}$ and w_0 using data
- Test:
  - calculate $g(x) = \boldsymbol{w}^{\mathrm{T}}x+w_0$ and choose C_1 if $g(x) > 1$ or choose C_2 if $g(x)<-1$.

Loss Function

$$\min \ \frac{1}{2}\|\boldsymbol{w}\|^2$$
$$\text{s.t.} \quad y^{(\ell)}(\boldsymbol{w}^{\mathrm{T}}x^{(\ell)}+w_0)\geq 1,\ \ell=1,\ldots,N$$

- For a given input x, the model outputs a score $g(x) = \boldsymbol{w}^{\mathrm{T}}x+w_0$. Let $y \in \{-1, +1\}$ be the label of the real class (y =+1: $x \in C_1$, y = -1: $x \in C_2$):
  - if $y(\boldsymbol{w}^{\mathrm{T}}x+w_0) <1$: we aim to maximize $y(\boldsymbol{w}^{\mathrm{T}}x+w_0)$ until reaching 1, cost is $1-y(\boldsymbol{w}^{\mathrm{T}}x+w_0)$
  - if $y(\boldsymbol{w}^{\mathrm{T}}x+w_0) >1$: outlier points, no need to optimize, cost is **0**

- Can writhe this succinctly as a **Hinge** loss:

$$\ell\,(\boldsymbol{w}, w_0 \,|\, x, y) = \max(0,\ 1\text{-}y(\boldsymbol{w}^{\mathrm{T}}x+w_0))$$

- Given: $D = \{(x^{(1)}, r^{(1)}), \ldots, (x^{(N)}, r^{(N)})\}$, the loss over the dataset is defined as:

$$L\,(\boldsymbol{w}, w_0 \,|\, D) = \frac{1}{N}\sum_{\ell=1}^{N} \max(0,\ 1-y^{(\ell)}(\boldsymbol{w}^{\mathrm{T}}x^{(\ell)}+w_0)) + \frac{\lambda}{2}\|\boldsymbol{w}\|^2$$

Optimization – Gradient Descend

$$\boldsymbol{w}_{\text{new}} = \boldsymbol{w}_{\text{old}} - \eta \frac{\partial L}{\partial w}$$

What is $\frac{\partial L}{\partial w}$?

$$L(\boldsymbol{w}, w_0 \mid D) = \begin{cases} \frac{\lambda}{2} \|\boldsymbol{w}\|^2 & \textbf{if} \quad y^{(\ell)}(\boldsymbol{w}^{\mathrm{T}} x^{(\ell)} + w_0) \geq 1 \\ \frac{1}{N} \sum_{\ell=1}^{N} 1 - y^{(\ell)}(\boldsymbol{w}^{\mathrm{T}} x^{(\ell)} + w_0) + \frac{\lambda}{2} \|\boldsymbol{w}\|^2 & \text{otherwise} \end{cases}$$

For each $\boldsymbol{w}_{\mathrm{j}}$ ($j = 0, \ldots, d$):

$$\frac{\partial L}{\partial w_j} = \begin{cases} \lambda w_{\mathrm{j}} & \text{if } y^{(\ell)}(\boldsymbol{w}^{\mathrm{T}} x^{(\ell)} + w_0) \geq 1 \\ \lambda w_{\mathrm{j}} - y^{(\ell)} x_j^{(\ell)} & \text{o.w.} \end{cases}$$

$$\frac{\partial L}{\partial w_0} = \begin{cases} 0 & \text{if } y^{(\ell)}(\boldsymbol{w}^{\mathrm{T}} x^{(\ell)} + w_0) \geq 1 \\ -y^{(\ell)} & \text{o.w.} \end{cases}$$

The Algorithm

Gradient Descend for SVM

Input: $D = \{(x^{(\ell)}, y^{(\ell)})\}$ $(\ell=1:N)$
for $j = 0,\ldots,d$
  $\boldsymbol{w}_{\mathrm{j}} \leftarrow$ *rand* (-0.01, 0.01)
repeat
  for $\ell = 1,\ldots,N$
    $a \leftarrow 0$
    for $j = 0, \ldots, d$
      $a \leftarrow a + \boldsymbol{w}_{\mathrm{j}} x_{\mathrm{j}}^{(\ell)}$
    for $j = 0, \ldots, d$
      $\Delta\boldsymbol{w}_{\mathrm{j}} \leftarrow \lambda \boldsymbol{w}_{\mathrm{j}}$
    $\Delta\boldsymbol{w}_0 \leftarrow 0$
    if $y^{(\ell)} a < 1$:
      for $j = 0, \ldots, d$
        $\Delta\boldsymbol{w}_{\mathrm{j}} \leftarrow \Delta\boldsymbol{w}_{\mathrm{j}} - y^{(\ell)} x_j^{(\ell)}$
      $\Delta\boldsymbol{w}_0 \leftarrow \Delta\boldsymbol{w}_0 - y^{(\ell)}$
  for $j = 0,\ldots,d$
    $w_{\mathrm{j}} \leftarrow w_{\mathrm{j}} - \eta \Delta w_{\mathrm{j}}$
until convergence

Lagrangian

- The primal optimization problem:

$$\begin{array}{l} \text{minimize } \tfrac{1}{2}\|\boldsymbol{w}\|^2 \\ \text{subject to } y^{(\ell)}(\boldsymbol{w}\boldsymbol{x}^{(\ell)}+w_0) \geq 1, \forall \ell \end{array}$$

- **Lagrangian**: $\mathcal{L}= \tfrac{1}{2}\|\boldsymbol{w}\|^2 - \Sigma_\ell \alpha_\ell (y^{(\ell)}(\boldsymbol{w}\boldsymbol{x}^{(\ell)}+w_0) - 1)$

$$\nabla_{\boldsymbol{w},w_0}\mathcal{L}=0 \Rightarrow \begin{cases} \dfrac{\partial \mathcal{L}}{\partial \boldsymbol{w}} = 0 \Rightarrow \boldsymbol{w} = \sum_{\ell=1}^{N} \alpha_\ell y^{(\ell)} x^{(\ell)} \\ \dfrac{\partial \mathcal{L}}{\partial w_0} = 0 \Rightarrow \sum_{\ell=1}^{N} \alpha_\ell y^{(\ell)} = 0 \end{cases}$$

- Substitute back in the primal to get the dual

$$\begin{array}{ll} \textbf{maximize} & \mathcal{L}(\alpha)=\Sigma_\ell \alpha_\ell - \tfrac{1}{2}\sum_{\ell=1}^{N}\sum_{\ell'=1}^{N} \alpha_l \alpha_{l'} y^{(\ell)} y^{(\ell')} (x^{(\ell)})^{\mathrm{T}} x^{(\ell')} \\ \textbf{subject to} & \alpha_\ell \geq 0,\ \Sigma_{\ell=1}^{N} \alpha_\ell y^{(\ell)} = 0 \end{array}$$

The Dual Problem

Dual optimization problem:

$$
\begin{aligned}
&\text{maximize} \quad \sum_{\ell=1}^{N} \alpha_\ell - \frac{1}{2} \sum_{\ell=1}^{N} \sum_{\ell'=1}^{N} \alpha_\ell \alpha_{\ell'} y^{(\ell)} y^{(\ell')} (x^{(\ell)})^{\mathrm{T}} x^{(\ell')} \\
&\text{subject to} \quad \sum_{\ell=1}^{N} \alpha_\ell y^{(\ell)} = 0 \\
&\qquad\qquad\quad\;\; \alpha_\ell \geq 0,\ \ell = 1 \dots \mathrm{N}
\end{aligned}
$$

- This is also a QP problem, but its complexity depends on the sample size N (rather than the input dimensionality d)

Primal and Dual

Primal

$$\min \ \frac{1}{2}||\boldsymbol{w}||^2$$

$$\text{s.t.} \quad y^{(\ell)}(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}^{(\ell)}+w_0)\geq 1,\ \forall \ell$$

The complexity depends on the dimensionality d of inputs

Dual

$$\max \ \Sigma_\ell \alpha_\ell - \tfrac{1}{2}\Sigma_\ell \Sigma_{\ell'} \alpha_\ell \alpha_{\ell'} y^{(\ell)} y^{(\ell')} (x^{(\ell)})^T x^{(\ell')}$$

$$\text{s.t.} \quad \Sigma_\ell \alpha_\ell y^{(\ell)}=0$$

$$\alpha_\ell \geq 0,\ \ \ell=1\ldots N$$

The complexity depends on the sample size N

- **It turns out to be more convenient to solve the dual problem** than solving the **primal problem** ($N < d$)
- We can firstly solve **Dual** to obtain $\{\alpha_\ell\}$, and then obtain the W in **Primal**

Training (Dual)

- Given: $D = \{(x^{(1)}, y^{(1)}), \ldots, (x^{(N)}, y^{(N)})\}$
- minimize the **loss function** $L(\alpha)$ using any general purpose optimization toolkits (e.g., Matlab)
- But SVM is usually optimized using the **SMO** (sequential minimal optimization)

> - **Goal:**
>
> $$\min_{\alpha} L(\boldsymbol{\alpha})$$
>
> - **Iteration:**
>
> Update two variables each time until convergence {
> 1. Select an α_1 that violates the KKT condition
> 2. Pick a second multiplier α_2 and optimize $L(\alpha)$ w.r.t. α_1 and α_2
>
> }

https://en.wikipedia.org/wiki/Sequential_minimal_optimization

Support Vectors

Suppose the optimal $\{\alpha_\ell\}$ have been obtained

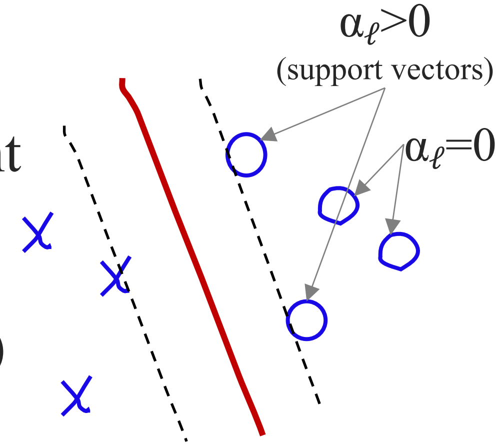
- Patterns for which $y^{(\ell)}(\boldsymbol{w}x^{(\ell)}+w_0) > 1$

  $\alpha_\ell=0$ (inactive constraints) $\Rightarrow x^{(\ell)}$ irrelevant

  recall complimentary slackness: $\lambda_i^* g_i(x^*)=0$

- Patterns that have $\alpha_\ell>0$ (active constraints)

  $y^{(\ell)}(\boldsymbol{w}x^{(\ell)}+w_0) = 1 \Rightarrow x^{(\ell)}$ lies on margin



- Most of the dual variables vanish with $\alpha_\ell=0$. They are points lying beyond the margin with **no effect** on the hyperplane.
- Solution is only determined by the examples on the margin (support vectors), i.e., $x^{(\ell)}$ with $\alpha_\ell>0$, hence the name **support vector machine (SVM).**

Computation of Primal Variables

- Having obtained the optimal $\boldsymbol{\alpha}$, we can obtain $\boldsymbol{w}$:

$$\boxed{\mathbf{w} = \sum_{\ell=1}^{N} \alpha_\ell y^{(\ell)} \mathbf{x}^{(\ell)} = \sum_{\mathbf{x}^{(\ell)} \in \mathcal{SV}} \alpha_\ell y^{(\ell)} \mathbf{x}^{(\ell)}}$$

where $\mathcal{SV}$ denotes the set of support vectors.

- The support vectors must lie on the margin, so they should satisfy $y^{(\ell)}(\mathbf{w}^T \mathbf{x}^{(\ell)} + w_0) = 1$ or $w_0 = y^{(\ell)} - \mathbf{w}^T \mathbf{x}^{(\ell)}$
- For numerical stability, all support vectors are used to compute w_0:

$$\boxed{w_0 = \frac{1}{|\mathcal{SV}|} \sum_{\mathbf{x}^{(\ell)} \in \mathcal{SV}} \left(y^{(\ell)} - \mathbf{w}^T \mathbf{x}^{(\ell)} \right)}$$

Prediction

- **Discriminant Function:**

$$
\begin{aligned}
g(\mathbf{x}) &= \mathbf{w}^T\mathbf{x} + w_0 \\
&= \left(\sum_{\mathbf{x}^{(\ell)} \in \mathcal{SV}} \alpha_\ell y^{(\ell)} \mathbf{x}^{(\ell)}\right)^T \mathbf{x} + \frac{1}{|\mathcal{SV}|} \sum_{\mathbf{x}^{(\ell)} \in \mathcal{SV}} \left(y^{(\ell)} - \mathbf{w}^T\mathbf{x}^{(\ell)}\right)
\end{aligned}
$$

- **Decision Rule:**

$$
\text{Choose} \begin{cases} C_1 & \text{if } g(x) > +1 \\ C_2 & \text{if } g(x) < -1 \end{cases}
$$

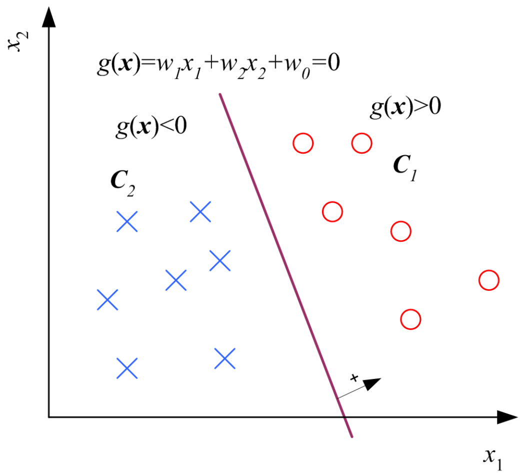
Programming Time

Tutorial: SVM from scratch with explanation (Python)

https://towardsdatascience.com/implementing-svm-from-scratch-784e4ad0bc6a

https://www.kaggle.com/code/misbahbilgili/svm-from-scratch-with-explanation/notebook

What’s Next

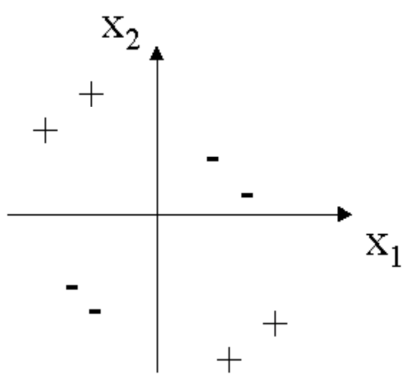


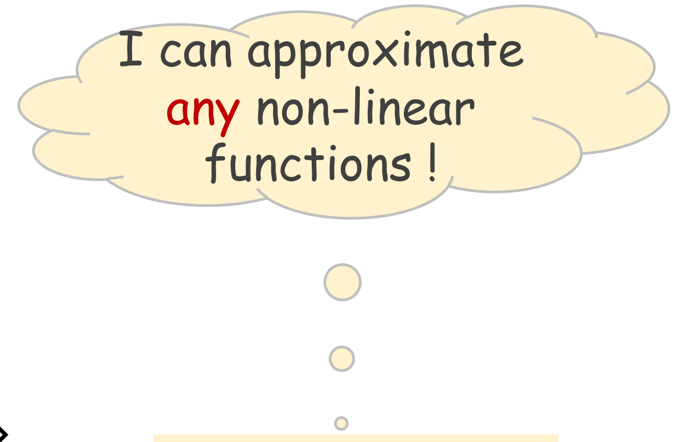
Perceprton

Logistic Regression

SVM

The curse of nonlinearity





Neural Networks