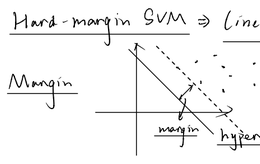
SVM: Supported Vector Machine

training set: $D = \{x_i, y_i\}_{i=1}^N$ where $y_i \in \{-1, +1\}$

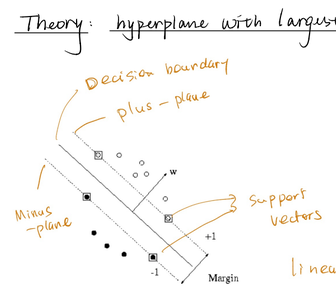
$\Rightarrow \hat{y} = \text{sign}(w^T x + b)$ (sign(): step function)

$x = (x_1 \cdots x_D)^T \quad w = (w_1 \cdots w_D)^T$

Hard-margin SVM $\Rightarrow$ linearly seperable data



Theory: hyperplane with largest margin generalize beste



linear SVM

since: $\hat{y} = \text{sign}(w^T x + b)$, Decision Boundary: $w^T x + b = 0$

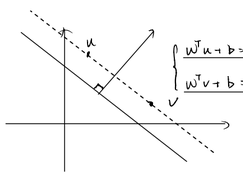
let: plus-plane: $w^T x + b = c$.
minus-plane: $w^T x + b = -c$

$\Rightarrow$ rewrite: $\frac{1}{c} w^T x + \frac{b}{c} = 1$, $\frac{1}{c} w^T x + \frac{b}{c} = -1$ $\Rightarrow$

rename: $\frac{w}{c} \to w$, $\frac{b}{c} \to b$.

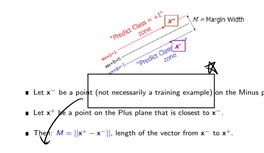
plus-plane: $w^T x + b = 1$
minus-plane: $w^T x + b = -1$

"Canonical Seperate hyperplane"



$w^T u + b = 1$, $w^T v + b = 1$ $\Rightarrow w^T (u - v) = 0$

$\Rightarrow$ w is perpendicular to the 3 planes



- Let x^- be a point (not necessarily a training example) on the Minus plane.
- Let x^+ be a point on the Plus plane that is closest to x^-.
- Then: $M = \|x^+ - x^-\|$, length of the vector from x^- to x^+.

$w^T \bar{x}^+ + b = 1$
$w^T \bar{x}^- + b = -1$
$\bar{x}^+ - \bar{x}^- = \lambda \bar{w}$, where λ is a constant

$w^T (\bar{x}^+ - \bar{x}^-) = 2 \Rightarrow w^T \cdot \lambda w = 2 \Rightarrow \boxed{\lambda = \frac{2}{\bar{w}^T \bar{w}}}$

$\Rightarrow |M|$ (length of margin) $= \|\lambda \bar{w}\| = \frac{2}{\sqrt{w^T w}} = \boxed{\frac{2}{\|w\|}}$

Optimal Margin Classifier

$\Rightarrow$ Max: $|M| = \frac{2}{|w|}$ with constraints: $w^T x_i + b \begin{cases} \geq 1 & \text{if } y_i = 1 \\ \leq -1 & \text{if } y_i = -1 \end{cases}$

$\min \frac{\|w\|^2}{2}$ subject to $y_i (w^T x_i + b) \geq 1$

"The Primal Optimization Problem"

Lagrangian multiplier

$\mathcal{L}(w, b, \alpha) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^N \alpha_i [(w^T x_i + b) y_i - 1]$

where: $w = (w_1 \cdots w_D)$ $b = (b_1 \cdots b_n)$ $\alpha = (\alpha_1 \cdots \alpha_n)$

primal primal problem:

$\min_{w,b} L_p(w, b) = \min_{w,b} \max_{\alpha} \mathcal{L}(w, b, \alpha)$

dual problem:

$\max_{\alpha} L_d(\alpha) = \max_{\alpha} \min_{w,b} \mathcal{L}(w, b, \alpha)$

$\max_{\alpha} \min_{w,b} \mathcal{L}(w, b, \alpha) \leq \max_{\alpha} \min_{w,b} \max_{\alpha} \mathcal{L}(w, b, \alpha)$

$= \min_{w,b} \max_{\alpha} \mathcal{L}(w, b, \alpha)$

Dual Problem $\Rightarrow$ "A Lower Bound"

"$d^* \leq p^*$"

When traning example is linear seperable

$\exists\, w^*, b^*, \alpha^*$, $d^* = \mathcal{L}(w^*, b^*, \alpha^*) = p^*$

Satisfies KKT Conditions:

① $\frac{\partial \mathcal{L}(w^*, b^*, \alpha^*)}{\partial w_j} = 0$
② $\frac{\partial \mathcal{L}(w^*, b^*, \alpha^*)}{\partial b} = 0$
③ $\alpha_i [y_i (w^{*T} x_i + b) - 1] = 0$. Dual Complementary Condition
④ $y_i (w^{*T} x_i + b) \geq 1$
⑤ $\alpha_i \geq 0$

① ② $\Rightarrow$ Optimality
③ ④ ⑤ $\Rightarrow$ constraints

For given i:
① $y_i (w^T x_i + b) > 1 \Rightarrow \alpha_i = 0$.
② $\alpha_i > 0 \Rightarrow y_i (w^T x_i + b) = 1$
support vector (on plus/minus plane)

Solving Dual Problem

$\max_\alpha L_d(\alpha) = \max_{\alpha, \alpha_i \geq 0} \min_{w,b} \mathcal{L}(w, b, \alpha) = \frac{1}{2}\|w^*\|^2 - \left(\sum_{i=1}^N \alpha_i [y_i (w^T x_i + b_i) - 1]\right)$

$\begin{cases} \frac{\partial L}{\partial w_k} = w_k - \sum_{i=1}^N \alpha_i y_i x_{ik} = 0 \\ \frac{\partial L}{\partial b} = -\sum_{i=1}^N \alpha_i y_i = 0 \end{cases}$

$\Rightarrow \begin{cases} w_k = \sum_{i=1}^N \alpha_i y_i x_{ik} \\ \sum_{i=1}^N \alpha_i y_i = 0 \end{cases}$

$\frac{1}{2}\|w\|^2 = -\frac{1}{2} \sum_{i,j=1}^N \alpha_i \alpha_j y_i y_j x_i^T x_j$

$\sum_{i=1}^N \alpha_i [y_i (w^T x_i + b_i) - 1]$
$= -\sum_{i=1}^N \alpha_i - \sum_{i=1}^N \alpha_i y_i (w^T x_i + b_i) = 0$.
$= -\sum_{i=1}^N \alpha_i$

$\therefore \max_\alpha L(\alpha) = \sum_{i=1}^N \alpha - \frac{1}{2} \sum_{i,j=1}^N \alpha_i \alpha_j y_i y_j x_i^T x_j$

How to find b ?

N_s: # of support vector.

kkt: $y_i (w^T x_i + b) = 1$

$y_i \in \{-1, 1\}$ $\leftarrow$ $y_i y_i (w^T x_i + b) = y_i$

$w^T x_i + b = y_i$

$b = y_i - w^T x_i = \boxed{\frac{1}{N_s} \sum_{i=1}^{N_s} (y_i - \bar{w}^T \bar{x}_i)}$

Prediction

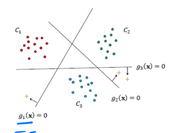
$\hat{y} = \text{sign}(w^T x + b) = \text{sign}\left(\sum_{i=1}^{N_s} \alpha_i y_i x_i^T x + b\right)$

Mutliple Classes

- One way to handle multiple (K) classes is to define K two-class problems, each separating one class from all other classes combined.
- Let $g_i(\mathbf{x})$ denote $\mathbf{w}_i^\top \mathbf{x} + b_i$ (Discriminant function)
- Classification rule:

$$\hat{y} = \arg\max_i g_i(\mathbf{x})$$

Note: Not using $\text{sign}(g_i(\mathbf{x}))$



$\bar{w}_i^T x + b_i$

Soft-Margin SVM: not linearly seperable

Relaxation: $y_i (w^T x_i + b) \geq 1 - \xi_i$ ($\xi_i \geq 0$) Slack Variable

Errors $\leq \sum_{i=1}^N \xi_i$

The larger the constant C, the more we want to minimize error, the more complex the decision boundary. Hyper-parameter

$\Rightarrow \min \frac{1}{2}\|w\|^2 + C \sum_i \xi_i$ subject to $y_i (w^T x_i + b) - 1 \geq 1 - \xi_i$

$\mathcal{L}(w, b, \xi, \alpha, r) = \frac{1}{2}\|w\|^2 + C \sum_{i=1}^N \xi_i - \sum_{i=1}^N \alpha_i (y_i (w^T x_i + b) - 1 + \xi_i) - \sum_{i=1}^N r_i \xi_i$ constraint: $\xi_i \geq 0$

- Original problem equivalent to (Primal Problem):

$$\min_{w,b,\xi} \max_{\alpha:\alpha_i \geq 0, r:r_i \geq 0} \mathcal{L}(\mathbf{w}, b, \xi, \alpha, \mathbf{r})$$

- Dual Problem:

$$\max_{\alpha:\alpha_i \geq 0, \mathbf{r}:r_i \geq 0} \min_{w,b,\xi} \mathcal{L}(\mathbf{w}, b, \xi, \alpha, \mathbf{r})$$

Solving the dual: $\min_{w,b,\xi} \mathcal{L}(\mathbf{w}, b, \xi, \alpha, \mathbf{r})$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{w}} = 0 \quad \Rightarrow \quad \mathbf{w} = \sum_{i=1}^N \alpha_i y_i \mathbf{x}_i$$
$$\frac{\partial \mathcal{L}}{\partial b} = 0 \quad \Rightarrow \quad \sum_{i=1}^N \alpha_i y_i = 0$$
$$\frac{\partial \mathcal{L}}{\partial \xi_i} = 0 \quad \Rightarrow \quad C - \alpha_i - r_i = 0$$

Dual Problem: $\Rightarrow \alpha \Rightarrow w, b$

$$\begin{aligned} \text{maximize} \quad & \sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i,j=1}^N \alpha_i \alpha_j y_i y_j \mathbf{x}_i^\top \mathbf{x}_j \\ \text{subject to} \quad & C \geq \alpha_i \geq 0, \quad i = 1, \ldots, N \\ & \sum_{i=1}^N \alpha_i y_i = 0 \end{aligned}$$

Impact of C:
- Small C: wider-margin, more misclassified points
- Large C: minimize misclassification errors, narrower margin

minimize $\frac{1}{2}\|\mathbf{w}\|^2 + C \sum_i \xi_i$

Relationship with Hinge Loss

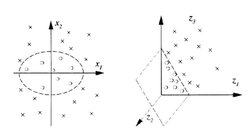
$\min \frac{1}{2}\|w\|^2 + C \sum_i \xi_i$ s.t. $y_i (w^T x_i + b) \geq 1 - \xi_i$, $\xi_i \geq 0$

$\begin{cases} \xi_i \geq 1 - y_i (w^T x_i + b) \\ \xi_i \geq 0 \end{cases} \Rightarrow \xi_i = \max\{0, 1 - y_i z_i\}$ where $z_i = w^T x_i + b$

Recall: Hinge loss: $L_{hin}(y, z) = \max\{0, 1 - yz\}$

$\Leftrightarrow \min \frac{1}{2}\|w\|^2 + C \sum_i \max\{0, 1 - y_i (w^T x_i + b)\}$

Kernel Extension



- Mapping $\varphi : \mathbb{R}^m \to \mathcal{H}$, $\mathbf{x} \mapsto \varphi(\mathbf{x})$
  - $\mathbb{R}^m$: input space (attributes)
  - $\mathcal{H}$: feature space (features)
- Transform the data with the mapping

$$(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N) \Rightarrow (\varphi(\mathbf{x}_1), y_1), \ldots, (\varphi(\mathbf{x}_N), y_N)$$

not linearly seperable $\xrightarrow[\uparrow \text{ dimension}]{\text{kernel}}$ linearly seperable

feature vector

Dual Problem on Features:

$$\begin{aligned} \text{maximize} \quad & \sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i,j=1}^N \alpha_i \alpha_j y_i y_j \varphi(\mathbf{x}_i)^\top \varphi(\mathbf{x}_j) \\ \text{subject to} \quad & C \geq \alpha_i \geq 0, \quad i = 1, \ldots, N \\ & \sum_{i=1}^N \alpha_i y_i = 0 \end{aligned}$$

Kernel Trick:

let $K(x_i, x_j) = \varphi(x_i)^T \varphi(x_j)$

Problem: $\max \sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i,j=1}^N \alpha_i \alpha_j y_i y_j K(\alpha_i, x_j)$
s.t. $C \geq \alpha_i \geq 0$. $i = 1 \cdots N$
$\sum_{i=1}^N \alpha_i y_i = 0$

The trick:
- No need to explicitly calculate φ and the inner product $\varphi(\mathbf{x}_i)^\top \varphi(\mathbf{x}_j)$
- Just need to specify the kernel function $K(\mathbf{x}_i, \mathbf{x}_j)$
- k is cheaper to calculate
- Allow one to use very high dimensional feature space.

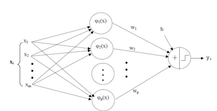
Common Kernels

- $k(\mathbf{x}, \mathbf{z}) \equiv (\mathbf{x}^\top \mathbf{z})^d$, polynomial kernel
  - Corresponding feature transformation when $d = 2$:

$$\phi(x_1, x_2, \ldots, x_n) = (x_1 x_1, x_1 x_2, x_1 x_3, \ldots, x_n x_n)$$

- Inhomogeneous polynomial: $k(\mathbf{x}, \mathbf{z}) = (\mathbf{x}^\top \mathbf{z} + 1)^d$
  - The corresponding feature transformation?
- Gaussian: $k(\mathbf{x}, \mathbf{z}) = \exp(-\|\mathbf{x} - \mathbf{z}\|^2 / 2\sigma^2)$
  - Radial basis function (RBF) network
  - Corresponds to an infinite-dimensional feature space
- Intuitively, $K(\mathbf{x}, \mathbf{z})$ represents our desired notion of similarity between data $\mathbf{x}$ and $\mathbf{z}$ and this is from our prior knowledge

Classification with SVM



- Choose nonlinear transformation $\varphi = (\phi_1, \phi_2, \ldots, \phi_p)$ (implicitly via $K(\mathbf{x}, \mathbf{z})$)
- Solve the dual optimization problem on features, get α
- Calculate $\mathbf{w}$ and b from α.
- Classify future examples as follows:

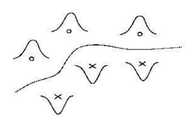
$$\text{sign}\left(\sum_{i=1}^{N_S} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b\right)$$

Gaussian Kernel

Recall that the decision rule uses

$$\text{sign}(\mathbf{w}^\top \phi(\mathbf{x}) + b) = \text{sign}\left(\sum_{i=1}^{N_S} \alpha_i y_i k(\mathbf{x}_i, \mathbf{x}) + b\right)$$

- Gaussian kernel: $k(\mathbf{x}_1, \mathbf{x}_2) = \exp(-\|\mathbf{x}_1 - \mathbf{x}_2\|^2 / 2\sigma^2)$
- Amounts to putting bumps of various sizes on the training set



- Small σ: Narrow Gaussian bell curve, Each support vector affects only nearby points, Decision boundary is complex and wiggly. Result: Low bias, high variance (overfitting risk)
- Large σ: Wide Gaussian bell curve, Each support vector affects a large neighborhood, Decision boundary is smooth and simple (near-linear). Result: High bias, low variance (underfitting risk)
- Small C: Soft margin (wide margin allowed), Many points can be misclassified or inside margin, Low penalty for training errors.
  Result: Simpler model, underfitting possible
- Large C: Hard margin (narrow margin enforced), Few misclassifications tolerated, High penalty for training errors. Result: Complex model, overfitting possible

When to use SVM instead of deep learning?

- Small datasets – SVM performs well with limited samples; deep learning needs large data.
- When training time must be short – SVM trains faster than deep networks for small/medium data.
- When interpretability matters – SVMs are simpler and easier to analyze than deep nets.
- When the decision boundary is complex but data is not huge – kernel tricks give SVM strong nonlinear power.
- Risk of overfitting is high – SVM's margin maximization naturally controls overfitting better on small data.