Kernel Based Support Vector Machines

= kernel classification

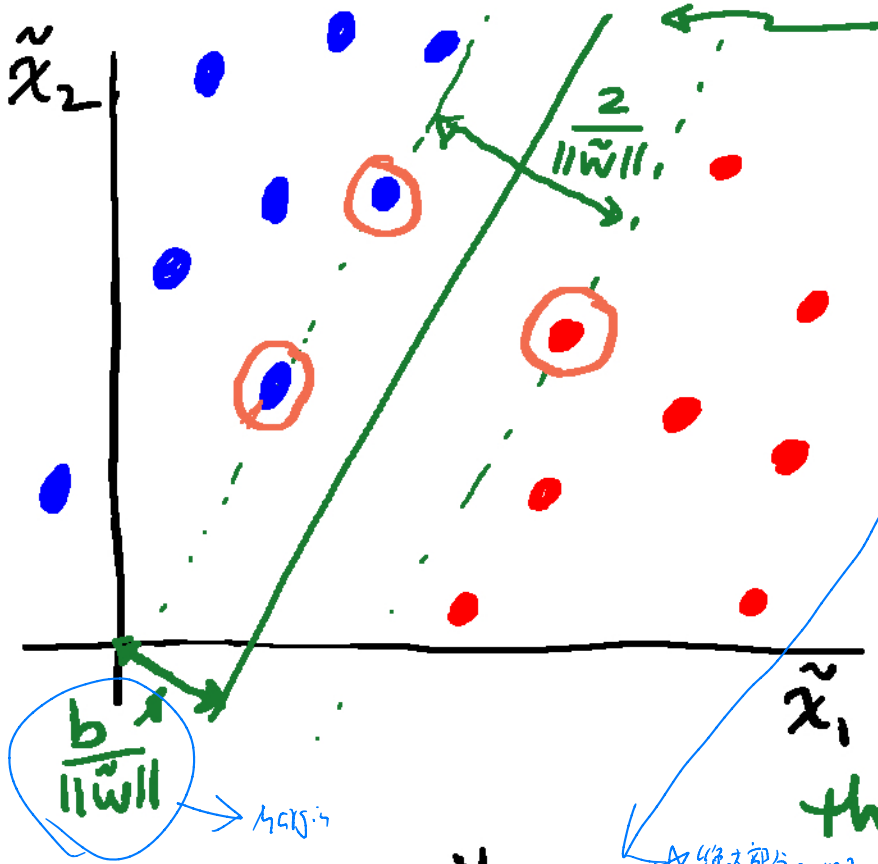
Objectives

- reformulate linear max margin classifier in terms of support vectors

- derive kernel version of hinge loss with ridge regression

- summarize features of support vector machines

Support vectors define max-margin classifier



$\tilde{x}^T\tilde{w} + b = 0$ → decision boundary

$x^T w = 0$

↓ b is included in w

$w = [\tilde{w}^T \; b]^T$ depends only on the support vectors

Recall kernel regression

weighted sum of kernels

$$d(x) = \phi^T(x) w = \sum_{j=1}^{N} \alpha_j K(x, x^j)$$

(phi: A high dimensional feature space derived from original x)

(The kernel of both x and the j^{th} training sample x^j)

Let $\phi(x) = x \Rightarrow K(x, x^j) = x^T x^j$

then $d(x) = \sum_{j=1}^{N} \alpha_j x^T x^j = x^T \underbrace{\sum_{j=1}^{N} \alpha_j x^j}_{w}$

☆ 绝大部分 α 都是 0，所以 w 实际由这些支持向量线性组合而来

So $w = \sum_{j=1}^{N} \alpha_j x^j$

→ 代表第 j 个样本在决定边界时的重要性（稀疏性）

All $\alpha_j = 0$ except Support vectors!

Use kernels for nonlinear decision boundaries

High-dimensional feature space: $\underline{x} \rightarrow \underline{\phi}(\underline{x})$

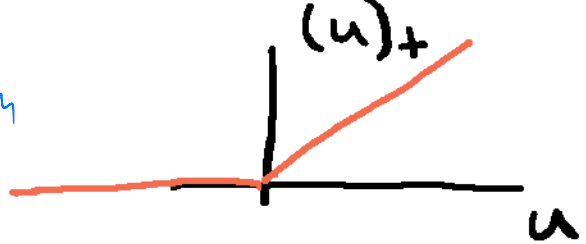
e.g., $\underline{\phi}(\underline{x}) = [\, x_1^2 \;\; x_2^2 \cdots x_2 x_4 \cdots x_{m-1} \;\; x_m \;\; 1 \,]$

$$\hat{d}(\underline{x}) = \text{sign}\left(\underline{\phi}^T(\underline{x})\,\underline{w}\right)$$

Hinge loss with ridge regression

$$\min_{\underline{w}} \sum_{i=1}^{N} \left(1 - d^i \underline{\phi}^T(\underline{x}^i)\,\underline{w}\right)_+ + \lambda \|\underline{w}\|_2^2$$

Cost function of SVM



☆ **Claim:** $\underline{w} = \sum_{j=1}^{N} \underline{\phi}(\underline{x}^j)\,\alpha_j$ → 可以替換成 solve for α (Representer Theorem)

(proof in notes)

Kernel "trick" replaces $\underline{\phi}^T(\underline{x}^i)\underline{\phi}(\underline{x}^i)$ with $K(\underline{x}^j, \underline{x}^i)$

Restate: $\underline{w} = \sum_{j=1}^{N} \alpha_j \underline{\phi}(\underline{x}^j)$

Hinge loss with ridge regression

$$\min_{\underline{\alpha}} \sum_{i=1}^{N} \left(1 - d^i \underline{\phi}^T(\underline{x}^i) \underbrace{\sum_{j=1}^{N} \alpha_j \underline{\phi}(\underline{x}^j)}_{\underline{w}}\right)_+ + \lambda \underbrace{\sum_{i=1}^{N} \alpha_i \underline{\phi}^T(\underline{x}^i)}_{\underline{w}^T} \underbrace{\sum_{j=1}^{N} \alpha_j \underline{\phi}(\underline{x}^j)}_{\underline{w}}$$

$$\min_{\underline{\alpha}} \sum_{i=1}^{N} \left(1 - d^i \sum_{j=1}^{N} \alpha_j \underbrace{\underline{\phi}^T(\underline{x}^i)\underline{\phi}(\underline{x}^j)}_{K(\underline{x}^i, \underline{x}^j)}\right)_+ + \lambda \sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j \underbrace{\underline{\phi}^T(\underline{x}^i)\underline{\phi}(\underline{x}^j)}_{K(\underline{x}^i, \underline{x}^j)}$$

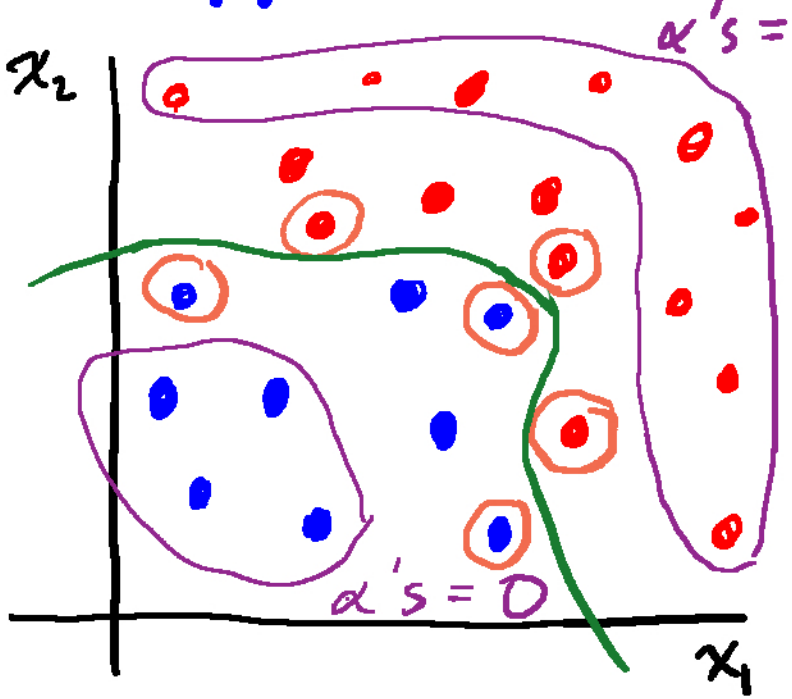
Kernel "trick"

SVM

$$\min_{\underline{\alpha}} \sum_{i=1}^{N} \left(1 - d^i \sum_{j=1}^{N} \alpha_j K(\underline{x}^i, \underline{x}^j)\right)_+ + \lambda \sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j K(\underline{x}^i, \underline{x}^j)$$

objective function

Support vector machines have sparse $\underline{\alpha}$



decision boundary

$$d(\underline{x}) = 0 = \underline{\phi}^T(\underline{x})\underline{w} = \sum_{j=1}^{N} \alpha_j K(\underline{x}, \underline{x}^j)$$

Boundary (hinge loss) depends only on the support vectors

$K(\underline{u},\underline{v})$ measures similarity/alignment of $\underline{u}, \underline{v}$

Ex: Gaussian kernels

$$K(\underline{u},\underline{v}) = \exp\left\{-\frac{\|\underline{u}-\underline{v}\|_2^2}{2\sigma^2}\right\}$$

Solve for $\underline{\alpha}$ using gradient descent

1. 预测函数 $d(\underline{x})$

两者的底层数学形式其实是**一样**的，都是核函数的加权和，只是用法不同。

- **通用公式：**

$$d(\underline{x}) = \sum_{i=1}^{N} \alpha_i K(\underline{x}, \underline{x}^i)$$

- **回归 (Regression)**：直接使用 $d(\underline{x})$ 的值作为预测结果（例如预测房价）。
- **分类 (SVM)**：使用 $\text{sign}(d(\underline{x}))$ 作为分类结果。
  - 如果 $d(\underline{x}) > 0 \rightarrow$ 预测为 **+1** 类。
  - 如果 $d(x) < 0 \rightarrow$ 预测为 **-1** 类。

2. 损失函数 (Cost Function)

这是两者本质的区别，决定了模型是"钻牛角尖"还是"只求稳妥"。

- **回归用的：平方误差 (Squared Error)**

$$(y_{true} - y_{pred})^2$$

  - *笔记*：它要求预测值必须无限接近真实值。
- **SVM 用的：合页损失 (Hinge Loss)**

$$(1 - y_{true} \cdot y_{pred})_+$$

  - *笔记*：$(z)_+$ 的意思是 $\max(0, z)$。
  - 它的意思是：只要你分类正确且信心足够大（即 $y \cdot d(x) > 1$），损失就是 0，我就不惩罚你了。这让 SVM 具有很强的**鲁棒性**。

3. 目标函数 (Objective Function)

这就是我们要让电脑去"最小化"的总公式。

- **核回归 (Kernel Ridge Regression)：**

$$\min_{\underline{w}} \sum_{i=1}^{N} (d^i - \underline{\phi}^T(\underline{x}^i)\underline{w})^2 + \lambda ||\underline{w}||^2$$

- **核分类 (Kernel SVM)：**

$$\min_{\underline{w}} \sum_{i=1}^{N} (1 - d^i \underline{\phi}^T(\underline{x}^i)\underline{w})_+ + \lambda ||\underline{w}||^2$$

4. 权重 $\underline{w}$ 与稀疏性

根据表示定理，$\underline{w}$ 总是数据的线性组合：$\underline{w} = \sum \alpha_j \phi(\underline{x}^j)$。

- **回归**：α_j 通常**都不为 0**。因为很难有样本的预测误差恰好完全等于 0，所以每个样本都会对模型产生一点拉力。
- **SVM**：α_j **大部分为 0**。只要样本被正确分类且在边界之外（安全区），它的 Hinge Loss 就是 0，对应的 α 也就是 0。只有那些在边界上的**支持向量**，α 才不为 0。

Kernel Based Support Vector Machines
Proof: Weights Lie in Space Spanned by $\boldsymbol{\phi}(\boldsymbol{x}^j)$

Background: Classifier training features and labels are $\boldsymbol{x}^i, d^i, i = 1, 2, \ldots N$. Classification in a high-dimensional feature space is performed using the mapping $\boldsymbol{\phi}(\boldsymbol{x})$ as $\hat{d}(\boldsymbol{x}) = \text{sign}\left\{\boldsymbol{\phi}^T(\boldsymbol{x})\boldsymbol{w}\right\}$.

Claim: The weights $\boldsymbol{w}$ that satisfy

$$\min_{\boldsymbol{w}} f(\boldsymbol{w}) = \min_{\boldsymbol{w}} \sum_{i=1}^{N} \left(1 - d^i \boldsymbol{\phi}^T(\boldsymbol{x}^i)\boldsymbol{w}\right)_+ + \lambda ||\boldsymbol{w}||_2^2 \tag{1}$$

are of the form

$$\boldsymbol{w} = \sum_{j=1}^{N} \boldsymbol{\phi}(\boldsymbol{x}^j)\alpha_j$$

Proof: The proof proceeds by adding a component to $\boldsymbol{w}$ that is orthogonal to the space spanned by the vectors $\{\boldsymbol{\phi}(\boldsymbol{x}^j), j = 1, 2, \ldots, N\}$ and then showing that component must be zero at the minimum of Eq. 1.

Suppose

$$\boldsymbol{w} = \sum_{j=1}^{N} \boldsymbol{\phi}(\boldsymbol{x}^j)\alpha_j + \boldsymbol{\phi}^\perp$$

where $\boldsymbol{\phi}^\perp$ is orthogonal to the space spanned by $\{\boldsymbol{\phi}(\boldsymbol{x}^j), j = 1, 2, \ldots, N\}$, that is, $\boldsymbol{\phi}^T(\boldsymbol{x}^j)\boldsymbol{\phi}^\perp = 0, j = 1, 2, \ldots, N\}$. Note that any vector $\boldsymbol{w}$ can be expressed as a sum of a component in the space spanned by the $\boldsymbol{\phi}(\boldsymbol{x}^j)$ and a component orthogonal to that same space.

The optimization problem Eq. 1 may be rewritten

$$\min_{\boldsymbol{w}} f(\boldsymbol{w}) = \min_{\boldsymbol{\alpha}, \boldsymbol{\phi}^\perp} \sum_{i=1}^{N} \left(1 - d^i \boldsymbol{\phi}^T(\boldsymbol{x}^i) \left(\sum_{j=1}^{N} \boldsymbol{\phi}(\boldsymbol{x}^j)\alpha_j + \boldsymbol{\phi}^\perp\right)\right)_+ + \lambda \left|\left|\sum_{j=1}^{N} \boldsymbol{\phi}(\boldsymbol{x}^j)\alpha_j + \boldsymbol{\phi}^\perp\right|\right|_2^2 \tag{2}$$

$$= \min_{\boldsymbol{\alpha}, \boldsymbol{\phi}^\perp} \sum_{i=1}^{N} \left(1 - d^i \left(\sum_{j=1}^{N} \boldsymbol{\phi}^T(\boldsymbol{x}^i)\boldsymbol{\phi}(\boldsymbol{x}^j)\alpha_j + \boldsymbol{\phi}^T(\boldsymbol{x}^i)\boldsymbol{\phi}^\perp\right)\right)_+ + \lambda \left(\sum_{i=1}^{N}\sum_{j=1}^{N} \boldsymbol{\phi}^T(\boldsymbol{x}^i)\boldsymbol{\phi}(\boldsymbol{x}^j)\alpha_i\alpha_j + 2\sum_{i=1}^{N} \alpha_i \boldsymbol{\phi}^T(\boldsymbol{x}^i)\boldsymbol{\phi}^\perp + \boldsymbol{\phi}^{\perp T}\boldsymbol{\phi}^\perp\right) \tag{3}$$

where in the second line we have used the identity

$$\left|\left|\sum_{j=1}^{N}\boldsymbol{\phi}(\boldsymbol{x}^j)\alpha_j+\boldsymbol{\phi}^{\perp}\right|\right|_2^2=\left(\sum_{i=1}^{N}\boldsymbol{\phi}(\boldsymbol{x}^i)\alpha_i+\boldsymbol{\phi}^{\perp}\right)^T\left(\sum_{j=1}^{N}\boldsymbol{\phi}(\boldsymbol{x}^j)\alpha_j+\boldsymbol{\phi}^{\perp}\right)$$

and multiplied out the terms in the product.

Now use the fact that $\boldsymbol{\phi}^T(\boldsymbol{x}^i)\boldsymbol{\phi}^{\perp}=0$ to reexpress the optimization problem as

$$\min_{\boldsymbol{w}} f(\boldsymbol{w})=\min_{\boldsymbol{\alpha},\boldsymbol{\phi}^{\perp}}\sum_{i=1}^{N}\left(1-d^i\sum_{j=1}^{N}\boldsymbol{\phi}^T(\boldsymbol{x}^i)\boldsymbol{\phi}(\boldsymbol{x}^j)\alpha_j\right)_+ +\lambda\sum_{i=1}^{N}\sum_{j=1}^{N}\boldsymbol{\phi}^T(\boldsymbol{x}^i)\boldsymbol{\phi}(\boldsymbol{x}^j)\alpha_i\alpha_j+\lambda{\boldsymbol{\phi}^{\perp}}^T\boldsymbol{\phi}^{\perp} \tag{4}$$

$$=\min_{\boldsymbol{\alpha},\boldsymbol{\phi}^{\perp}}\sum_{i=1}^{N}\left(1-d^i\sum_{j=1}^{N}\boldsymbol{\phi}^T(\boldsymbol{x}^i)\boldsymbol{\phi}(\boldsymbol{x}^j)\alpha_j\right)_+ +\lambda\left|\left|\sum_{j=1}^{N}\boldsymbol{\phi}(\boldsymbol{x}^j)\alpha_j\right|\right|_2^2+\lambda\left|\left|\boldsymbol{\phi}^{\perp}\right|\right|_2^2 \tag{5}$$

The only term containing $\boldsymbol{\phi}^{\perp}$ is the last one, $\lambda\left|\left|\boldsymbol{\phi}^{\perp}\right|\right|_2^2$, which is nonnegative since $\lambda>0$. Consequently, we conclude the minimum is attained when $\boldsymbol{\phi}^{\perp}=\mathbf{0}$ and thus

$$\boldsymbol{w}=\sum_{j=1}^{N}\boldsymbol{\phi}(\boldsymbol{x}^j)\alpha_j$$