交通大数据

支持向量机

- 刘志远教授
- zhiyuanl@seu.edu.cn

支持向量机

❑ 学习目标

- 了解支持向量机的基本概念
- 掌握线性可分支持向量机模型
- 掌握软间隔支持向量机模型
- 理解非线性支持向量机中的核函数
- 介绍序列最小优化算法（阅读内容）

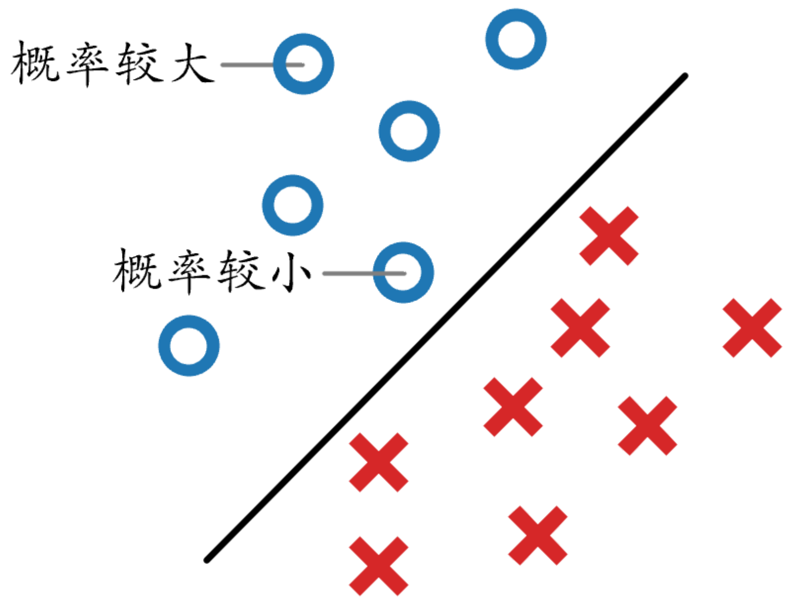
支持向量机

❑ 学习目标

- **了解支持向量机的基本概念**
- 掌握线性可分支持向量机模型
- 掌握软间隔支持向量机模型
- 理解非线性支持向量机中的核函数
- 介绍序列最小优化算法（阅读内容）

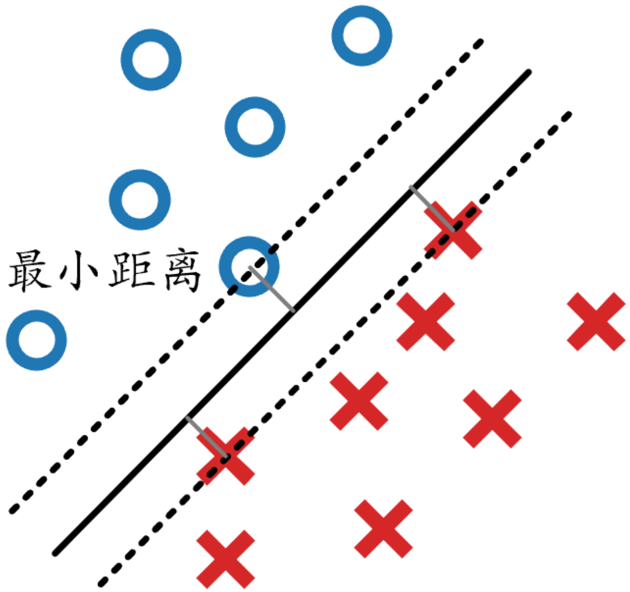
基本概念

- 如何将下面的数据分成两类?



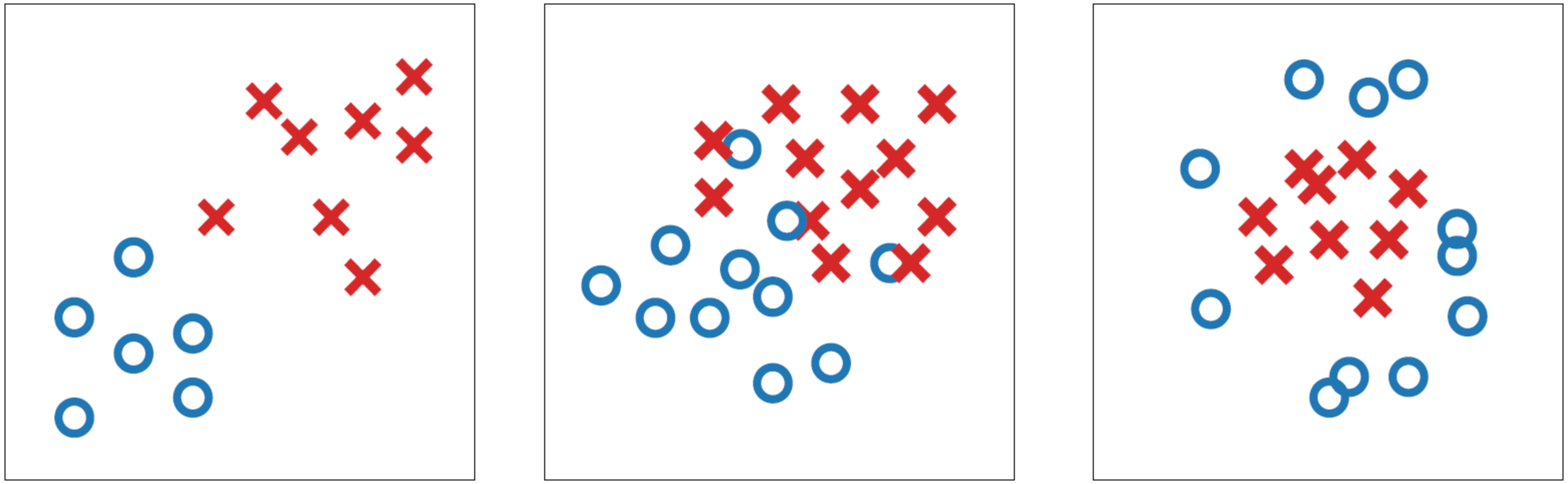
基本概念

- 如何将下面的数据分成两类?
- 是否有更符合直觉的方法?



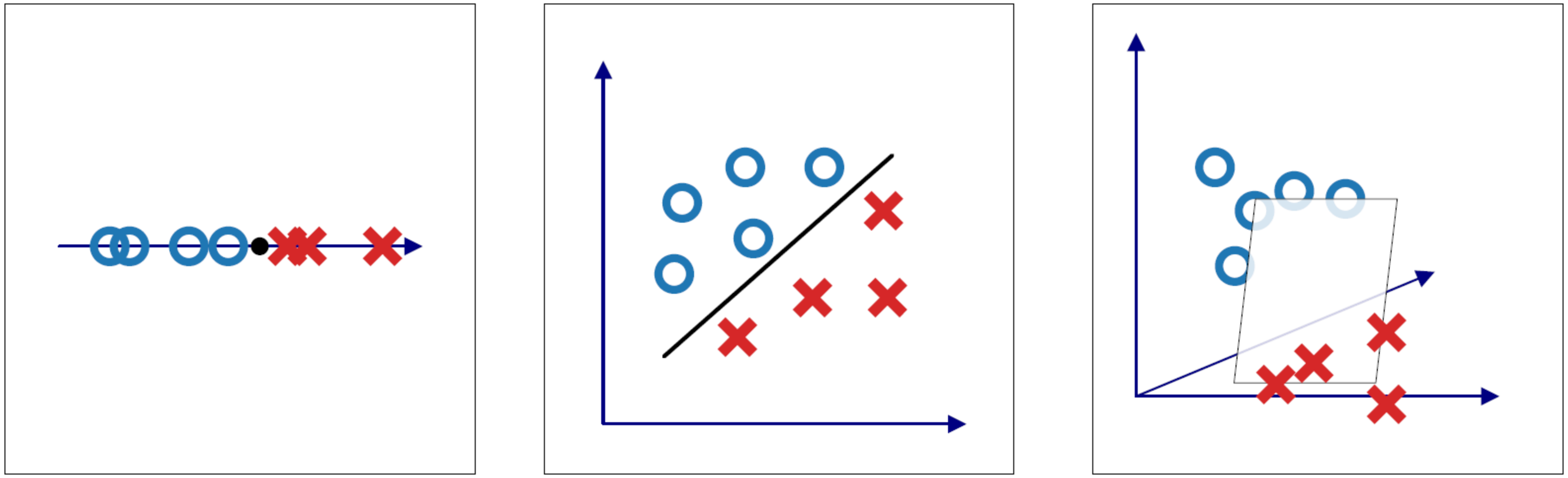
基本概念

- 下列数据能否用一条直线划分开来?
- **线性可分（linearly separable），硬间隔、软间隔**
- **线性不可分**



基本概念

- **不同维度**空间下的线性可分
- 对于n维数据，可以使用n - 1维的超平面进行分隔

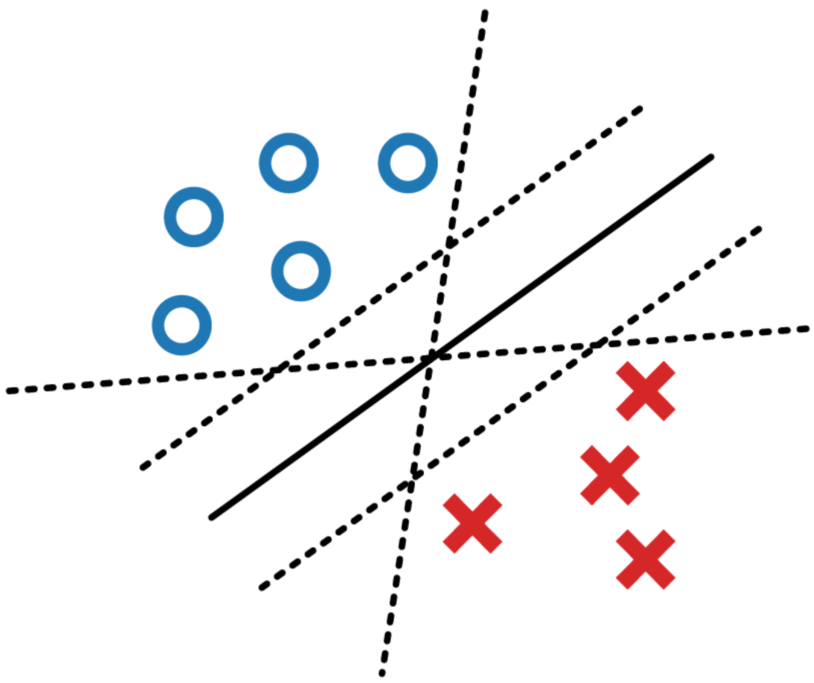


$\mathbb{R}^1$ $\mathbb{R}^2$ $\mathbb{R}^3$

超平面 $\mathbf{w}^{\mathsf{T}}\boldsymbol{x}+b=0$

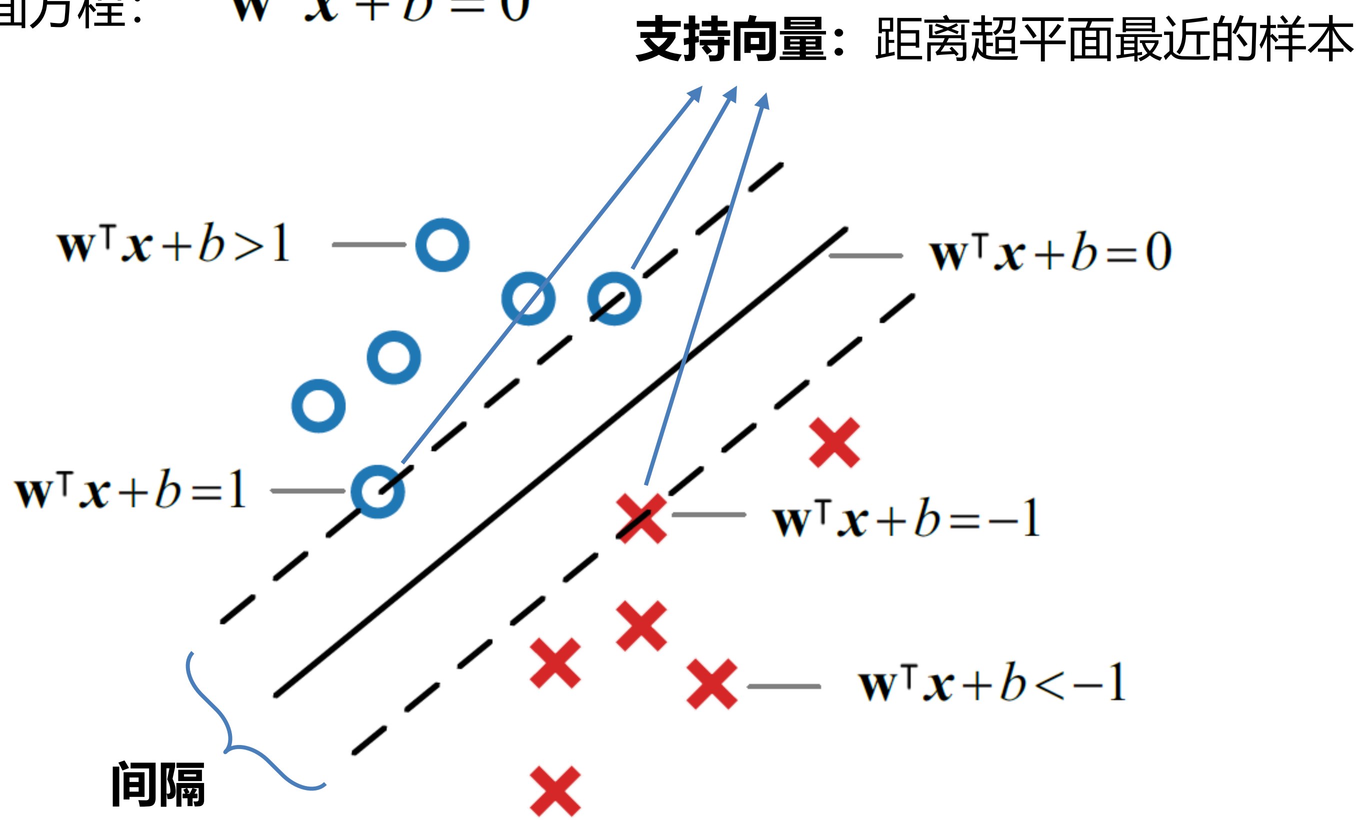
基本概念

- 哪一条是**最优分隔超平面**？为什么？



基本概念

- 超平面方程： $\mathbf{w}^{\top}\boldsymbol{x}+b=0$

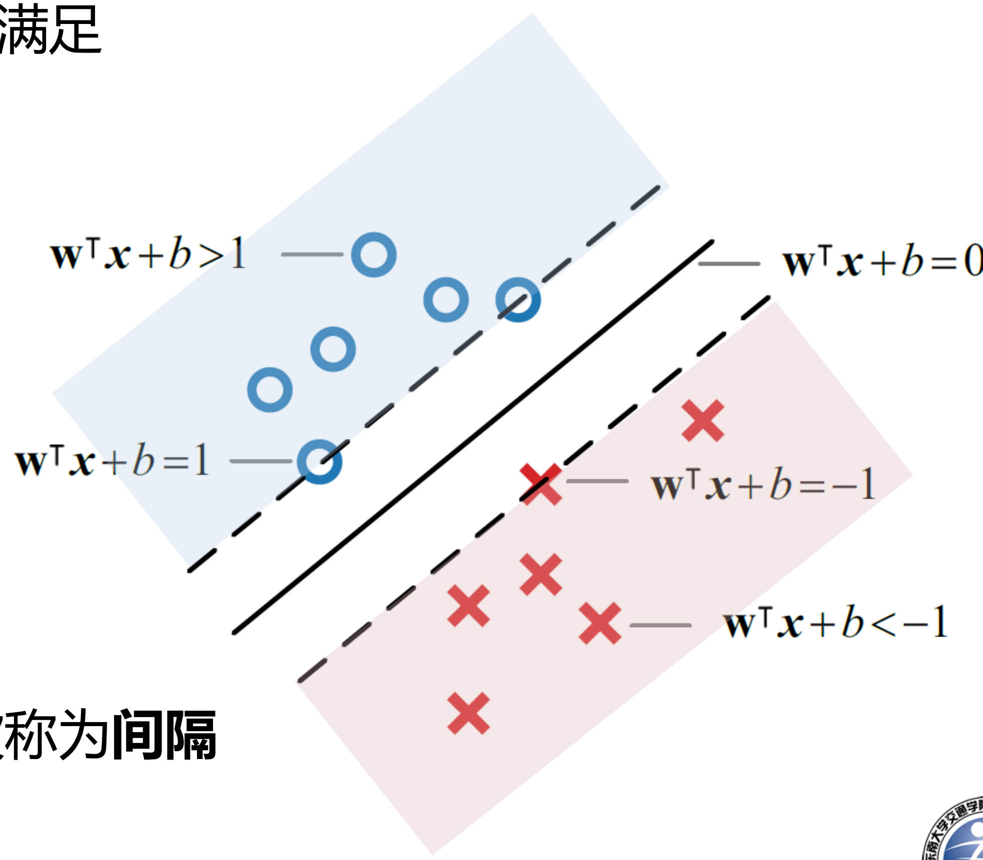


直观上讲，**支持向量机**是一种通过构建超平面来实现分类决策的机器学习模型。

基本概念

- 对于一个线性可分的数据集 D，总能找到一组参数**w**和b，对于任意样本 $(\boldsymbol{x}_i, y_i) \in D$ 都满足

$$\begin{cases} \mathbf{w}^\mathrm{T}\boldsymbol{x}_i + b \geq +1, & y_i = +1; \\ \mathbf{w}^\mathrm{T}\boldsymbol{x}_i + b \leq -1, & y_i = -1. \end{cases}$$



- 这一条件可简化为：

$$y_i(\mathbf{w}^\mathrm{T}\boldsymbol{x}_i + b) \geq 1$$

- 这一边界条件构成区域的宽度被称为**间隔**

基本概念

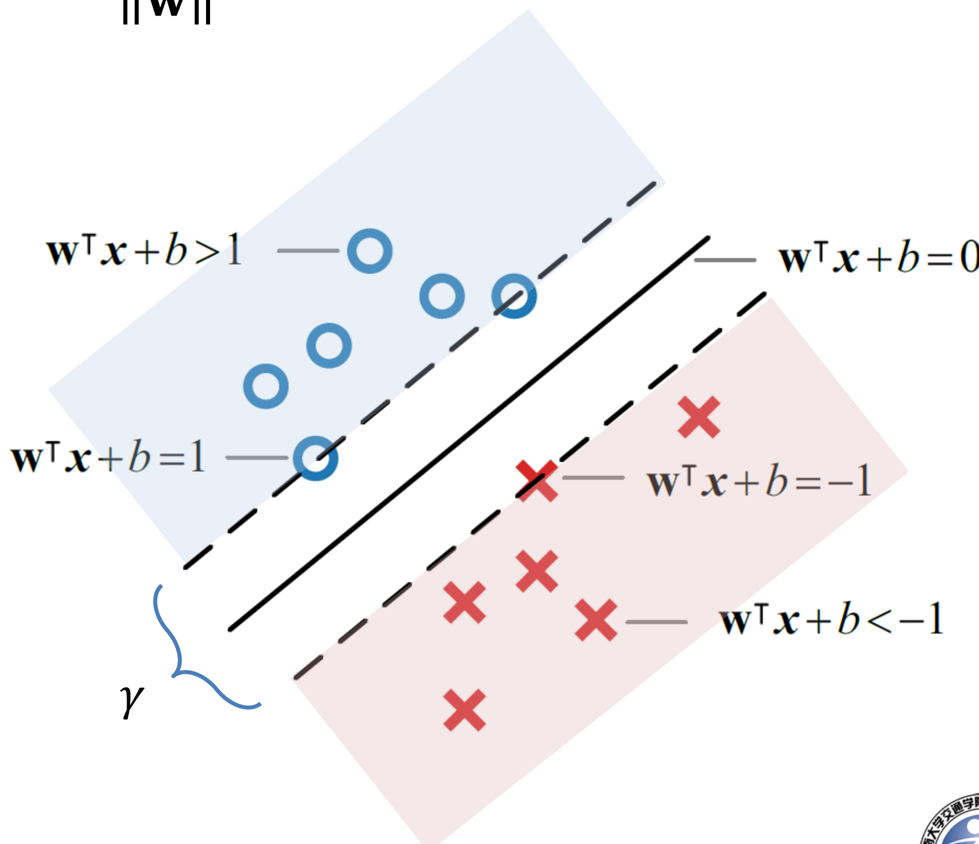
任意样本$\boldsymbol{x}$到超平面的距离：$\delta=\dfrac{|\mathbf{w}^{\top}\boldsymbol{x}+b\,|}{\|\mathbf{w}\|}$

- 根据点到直线距离公式，容易推出间隔大小 γ

$$\delta=\frac{|\mathbf{w}^{\top}\boldsymbol{x}+b\,|}{\|\mathbf{w}\|}$$

$$\Downarrow$$

$$\gamma=\frac{|+1|}{\|\mathbf{w}\|}+\frac{|-1|}{\|\mathbf{w}\|}=\frac{2}{\|\mathbf{w}\|}$$



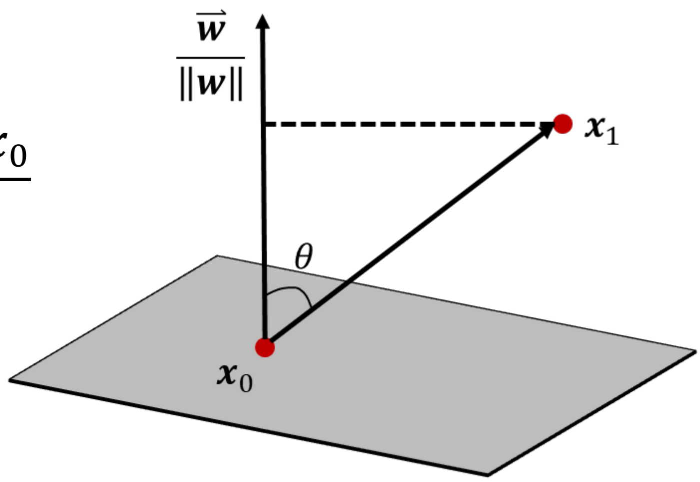
距离公式的推导

给定点$\boldsymbol{x}_1$，求其到超平面$\mathbf{w}^{\top}\boldsymbol{x}+b=0$的距离$\delta$

任取平面上一点$\boldsymbol{x}_0$，那么$\boldsymbol{x}_1$到平面的距离等于向量（$\boldsymbol{x}_1-\boldsymbol{x}_0$）在平面法方向上的投影。根据平面方程可知法方向为：$\vec{\boldsymbol{w}}/\|\boldsymbol{w}\|$

因此，点$\boldsymbol{x}_1$到平面距离为：

$$\delta=\frac{(\boldsymbol{x}_1-\boldsymbol{x}_0)\cdot\boldsymbol{w}}{\|\boldsymbol{w}\|}=\frac{w^T\boldsymbol{x}_1-w^T\boldsymbol{x}_0}{\|\boldsymbol{w}\|}$$



因为点$\boldsymbol{x}_0$在平面上，所以$\boldsymbol{w}^T\boldsymbol{x}_0+b=0$, 所以:

$$\delta=\frac{\boldsymbol{w}^T\boldsymbol{x}_1-\boldsymbol{w}^T\boldsymbol{x}_0}{\|\boldsymbol{w}\|}=\frac{\boldsymbol{w}^T\boldsymbol{x}_1+b}{\|\boldsymbol{w}\|}$$

支持向量机

❑ 学习目标

- 了解支持向量机的基本概念
- **掌握线性可分支持向量机模型**
- 掌握软间隔支持向量机模型
- 理解非线性支持向量机中的核函数
- 介绍序列最小优化算法（阅读内容）

线性可分支持向量机

❑ 支持向量机基本型

- 应用场景：线性可分的数据
- 优化目标：找到具有最大**间隔**的分隔超平面

$$\max_{\mathbf{w},b} \frac{2}{\|\mathbf{w}\|}$$

$$\text{s.t.} \quad y_i(\mathbf{w}^\top \boldsymbol{x}_i + b) \geq 1, \quad i = 1,2,\ldots,m.$$

- 该模型等价于

$$\min_{\mathbf{w},b} \frac{\|\mathbf{w}\|^2}{2}$$

$$\text{s.t.} \quad y_i(\mathbf{w}^\top \boldsymbol{x}_i + b) \geq 1, \quad i = 1,2,\ldots,m.$$

线性约束下的
凸二次规划

线性可分支持向量机

❑ 支持向量机基本型

- 模型存在哪些问题?

$$\min_{\mathbf{w},b} \frac{\|\mathbf{w}\|^2}{2}$$

$$\text{s.t.} \quad y_i(\mathbf{w}^\top \boldsymbol{x}_i + b) \geq 1, \quad i = 1,2,\ldots,m.$$

- 不等式约束较为复杂（数量太多），求解难度大

线性可分支持向量机

❑ 线性可分支持向量机的对偶问题

- 拉格朗日乘子： $\boldsymbol{\alpha} := (\alpha_1; \alpha_2; \ldots; \alpha_m)$
- 拉格朗日函数： $\mathcal{L}(\mathbf{w}, b, \boldsymbol{\alpha}) = \frac{1}{2}\|\mathbf{w}\|^2 + \sum_{i=1}^{m} \alpha_i [1 - y_i(\mathbf{w}^\top \boldsymbol{x}_i + b)]$
- KKT条件：

$$\nabla_{\mathbf{w}} \mathcal{L}(\mathbf{w}^*, \mathrm{b}^*, \boldsymbol{\alpha}^*) = \mathbf{0} \quad \rightarrow \quad \mathbf{w}^* = \sum_{i=1}^{m} \alpha_i^* y_i \boldsymbol{x}_i$$

$$\nabla_{\mathrm{b}} \mathcal{L}(\mathbf{w}^*, \mathrm{b}^*, \boldsymbol{\alpha}^*) = 0 \quad \rightarrow \quad 0 = \sum_{i=1}^{m} \alpha_i^* y_i$$

$$\boldsymbol{\alpha}^* \geq \mathbf{0}$$

$$\alpha_i^* [1 - y_i(\mathbf{w}^{*,\top} \boldsymbol{x}_i^* + b)] = 0, i = 1,2, \ldots, m$$

$$1 - y_i(\mathbf{w}^{*,\top} \boldsymbol{x}_i^* + b) \leq 0, i = 1,2, \ldots, m$$

- 进而，化简拉格朗日函数： $\mathcal{L}(\mathbf{w}, b, \boldsymbol{\alpha}) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j \boldsymbol{x}_i^\top \boldsymbol{x}_j$

教材书：141页的详细推导

线性可分支持向量机

线性可分支持向量机的对偶问题

- 对偶问题：

$$\max_{\boldsymbol{\alpha}} \sum_{i=1}^{m}\alpha_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j y_i y_j \boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{x}_j$$

$$\text{s.t.} \quad \sum_{i=1}^{m}\alpha_i y_i = 0,$$

$$\alpha_i \geq 0, \quad i = 1, 2, \ldots, m$$

- 模型存在哪些问题？
  - 对偶变量数量=样本数，求解难度大
  - 解决方法: Sequential Minimal Optimization 序列最小优化算法
  - （自学内容，看教材书7.5节）

线性可分支持向量机

线性可分支持向量机的对偶问题

- 如何从对偶问题的解得到原问题的解？

- 由KKT条件可得： $\mathbf{w}^* = \sum_{i=1}^{m} \alpha_i^* y_i \boldsymbol{x}_i$

$$\alpha_i^*[y_i(\mathbf{w}^{*,\mathsf{T}}\boldsymbol{x}_i + b^*) - 1] = 0$$ （互补松弛条件）

$\Downarrow$

必有 $\alpha_i^* = 0$ 或 $y_i(\mathbf{w}^{*,\mathsf{T}}\boldsymbol{x}_i^* + b) = 1$

① $\alpha_i^* = 0$ ⇨ 对$\mathbf{w}^*$没有影响 ⇨ $\alpha_i^* = 0$ 样本i 不是SV

② $\alpha_i^* > 0$ ⇨ 对$\mathbf{w}^*$有影响 ⇨ $\alpha_i^* > 0$ 样本i 是SV

线性可分支持向量机

线性可分支持向量机的对偶问题

- 如何从对偶问题的解得到原问题的解?

$y_i(\mathbf{w}^{*,\mathsf{T}}\boldsymbol{x}_i + b^*) = 1$ ⟶ 样本$(\boldsymbol{x}_i, y_i)$是**支持向量**!

- 将$\mathbf{w}^*$代入上式可得:

y_i 的取值为1或者-1

$$b^* = \frac{1}{y_i} - \sum_{j=1}^{m} \alpha_j^* y_j \boldsymbol{x}_j^\mathsf{T} \boldsymbol{x}_i = y_i - \sum_{j=1}^{m} \alpha_j^* y_j \boldsymbol{x}_j^\mathsf{T} \boldsymbol{x}_i$$

若 $\alpha_i^* = 0$ ⇨ 对b^*也没有影响

支持向量 $\Leftrightarrow \alpha_i^* > 0$

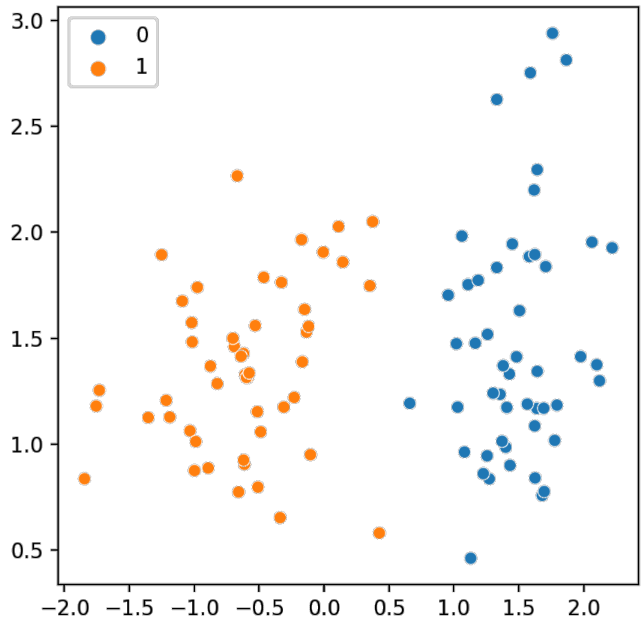
线性可分支持向量机

❑ 试一试

- 使用线性可分支持向量机对案例数据进行分类。

1. 生成案例数据

```
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.datasets import make_classification

X, y = make_classification(
    n_samples=100, n_features=2,
    n_redundant=0, n_informative=2,
    random_state=1, n_clusters_per_class=1
)
rng = np.random.RandomState(2)
X += rng.uniform(size=X.shape)

plt.figure(figsize=(5, 5))
sns.scatterplot(
    x=X[:,0], y=X[:,1], hue=y, legend='full')
plt.show()
```



线性可分支持向量机

❑ 试一试

- 使用线性可分支持向量机对案例数据进行分类。

2. 训练模型

```
from sklearn.preprocessing import StandardScaler
from sklearn.svm import SVC

# 数据标准化
X_ = StandardScaler().fit_transform(X)
data = X_, y

# 构建线性可分支持向量机，训练模型
clf = SVC(C=1e9, kernel='linear', random_state=3)
clf.fit(X_, y)
```

线性可分支持向量机

❑ 试一试

- 使用线性可分支持向量机对案例数据进行分类。

3. 查看分类结果

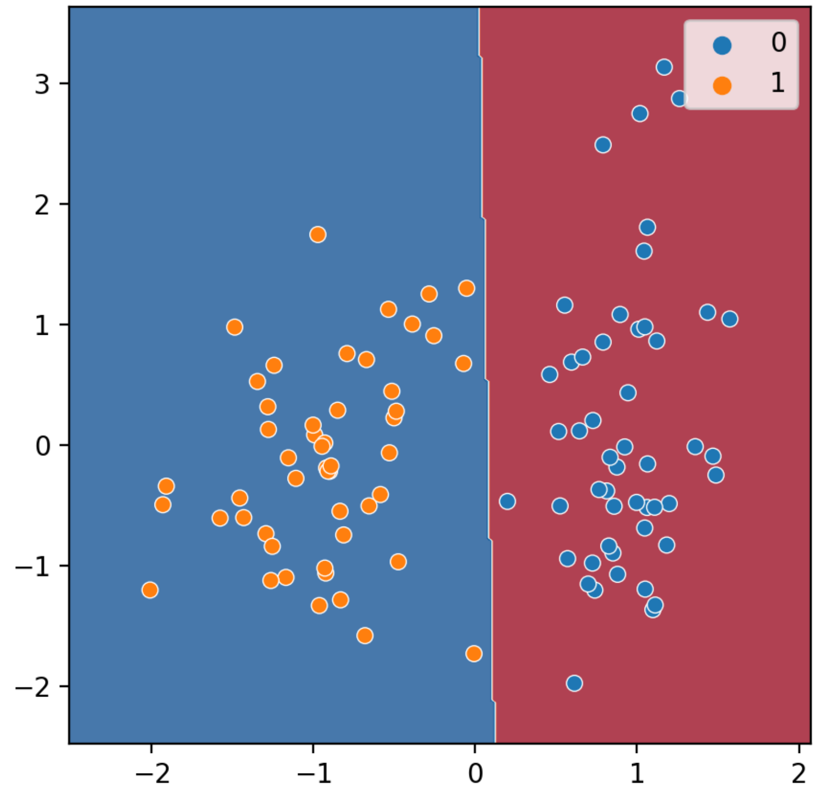
```
def plot_predicted_proba(data, clf, h=0.02):
    X, y = data

    x_min, x_max = X[:, 0].min() - .5, X[:, 0].max() + .5
    y_min, y_max = X[:, 1].min() - .5, X[:, 1].max() + .5
    xx, yy = np.meshgrid(np.arange(x_min, x_max, h),
                         np.arange(y_min, y_max, h))
    Z = clf.predict(np.c_[xx.ravel(), yy.ravel()])
    Z = Z.reshape(xx.shape)

    plt.figure(figsize=(5, 5))
    plt.contourf(xx, yy, Z, cmap=plt.cm.RdBu, alpha=.8)
    sns.scatterplot(x=X[:, 0], y=X[:, 1], hue=y, legend='full')
    plt.xlim(x_min, x_max)
    plt.ylim(y_min, y_max)
    plt.show()

plot_predicted_proba(data, clf)
```

线性可分支持向量机

❑ 试一试

- 使用线性可分支持向量机对案例数据进行分类。

3. 查看分类结果



支持向量机

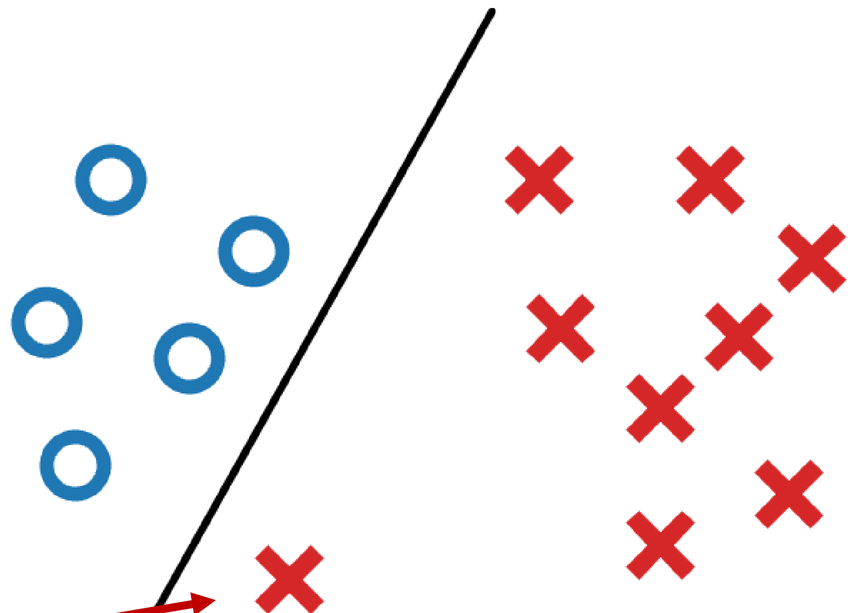
❑ 学习目标

- 了解支持向量机的基本概念
- 掌握线性可分支持向量机模型
- **掌握软间隔支持向量机模型**
- 理解非线性支持向量机中的核函数
- 介绍序列最小优化算法（阅读内容）

软间隔线性支持向量机

❑ 硬间隔与软间隔

- 下面的分隔方式合理吗?

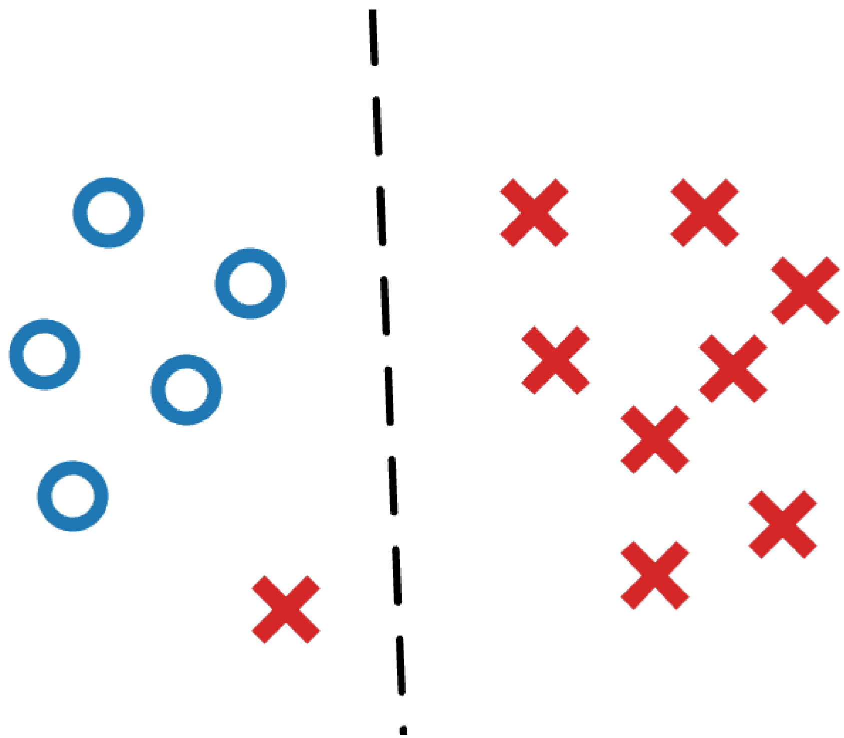


这很可能是数据**噪声**！

软间隔线性支持向量机

❑ 硬间隔与软间隔

- 下面的分隔方式合理吗？



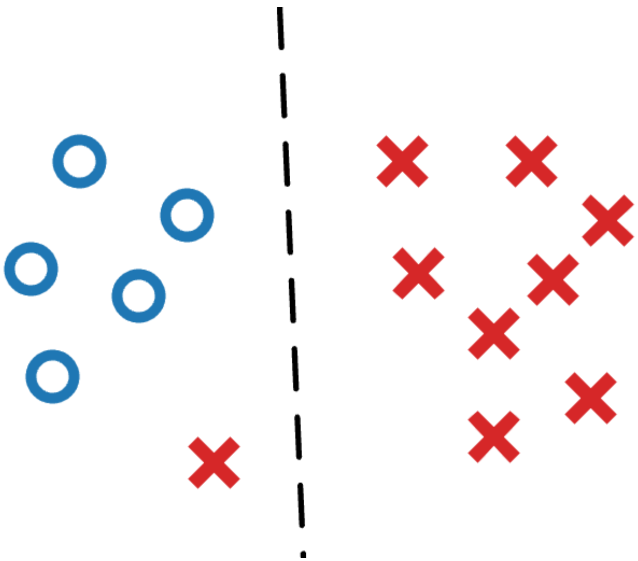
软间隔线性支持向量机

❑ 硬间隔与软间隔

- 支持向量机基本型的约束过于严格：
  - 所有样本必须在间隔**外侧**

$$y_i(\mathbf{w}^\mathrm{T}\boldsymbol{x}_i + b) \geq 1$$

- 可将此约束放松：
  - 允许样本在间隔**内侧**
  - 允许样本分类**出错**
- 对出错的样本增加**惩罚**即可



软间隔线性支持向量机

硬间隔与软间隔

- 改进数学模型：
  - 丢弃原先的约束条件，<u>允许分类出错</u>
  - 允许部分样本被错误划分的条件为软间隔（soft margin）

$$\cancel{y_i(\mathbf{w}^\top \boldsymbol{x}_i + b) \geq 1}$$

- 此时如何衡量模型好坏?

改造**目标函数**!

$$\min_{\mathbf{w},b} \frac{1}{2} \|\mathbf{w}\|^2 + \boxed{C \sum_{i=1}^{m} \ell\left(y_i(\mathbf{w}^\top \boldsymbol{x}_i + b)\right)}$$

超参数$C > 0$，**权衡**分类错误的影响大小
(当$C \rightarrow +\infty$，退化为线性可分支持向量机)

损失函数，度量分类的**错误程度**

软间隔线性支持向量机

❑ 硬间隔与软间隔

- 如何选择损失函数?
- 直接的想法：对错误分类加以**惩罚**

能否继续改进?

$$\ell_{0\text{-}1}(z) = \begin{cases} 1, & z < 0; \\ 0, & z \geq 0. \end{cases}$$

其中$z = y_i(\mathbf{w}^T \boldsymbol{x} + b)$

y_i：真实标签　$\mathbf{w}^T \boldsymbol{x} + b$：预测值

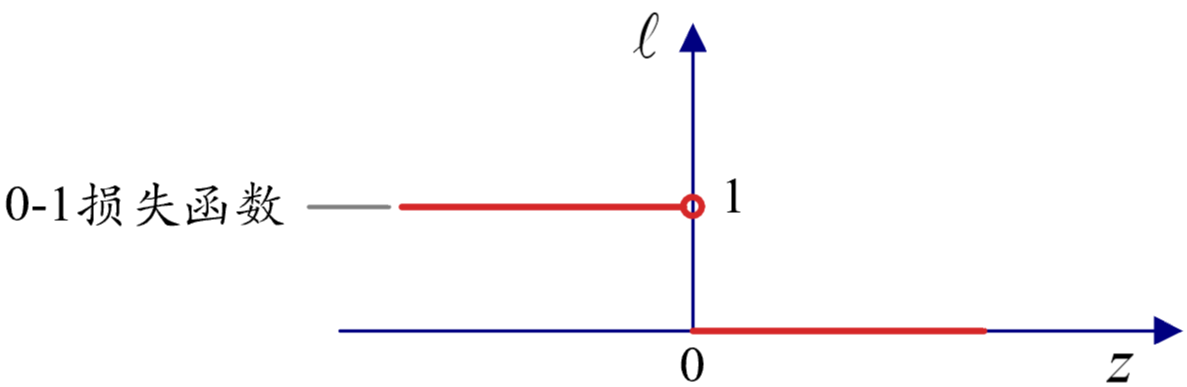
- 真实值与预测值**同号**，分类正确，不惩罚；
- 真实值与预测值**异号**，分类错误，进行惩罚

软间隔线性支持向量机

❑ 硬间隔与软间隔

- 0-1损失函数

$$\ell_{0\text{-}1}(z) = \begin{cases} 1, & z < 0; \\ 0, & z \geq 0. \end{cases}$$

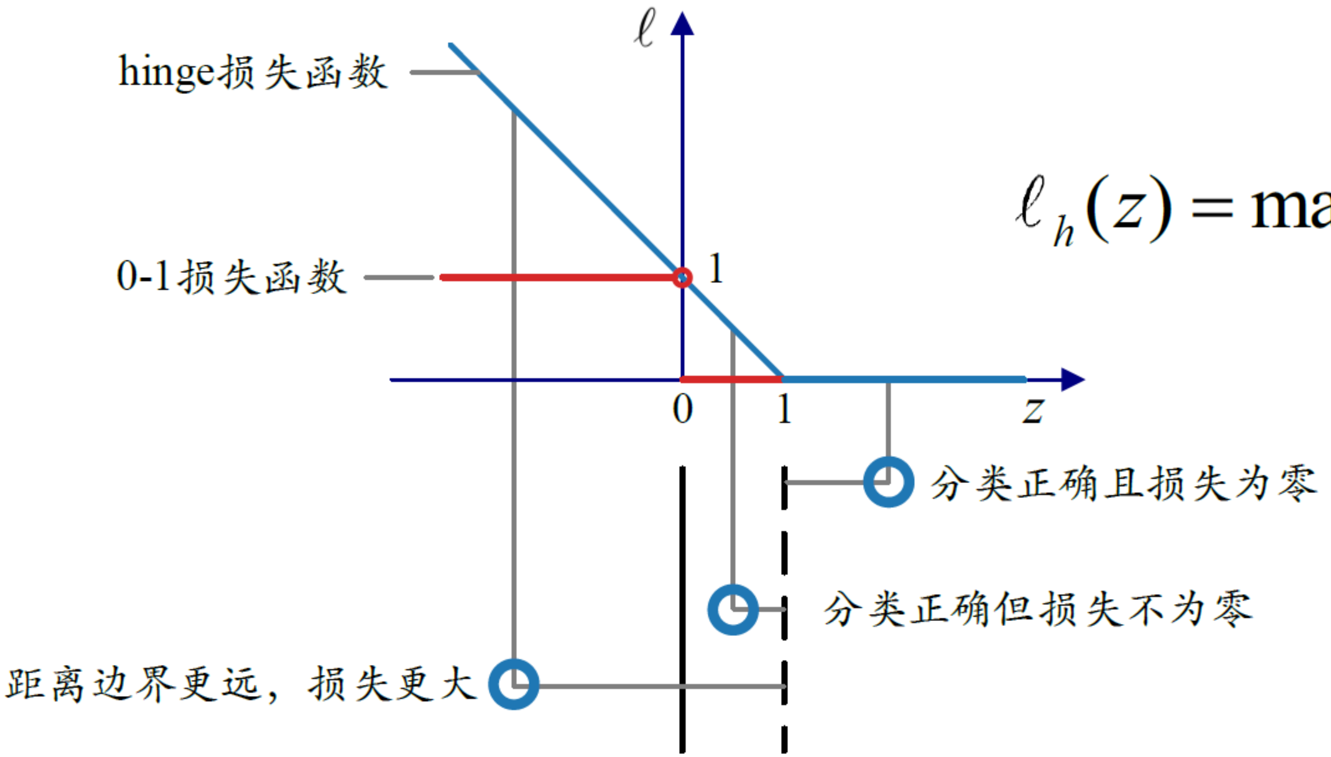


- 不连续
- 非凸

软间隔线性支持向量机

❑ 硬间隔与软间隔

- hinge损失函数！
- 进一步考虑样本分类的**错误程度**，离分隔超平面越远，错误越离谱



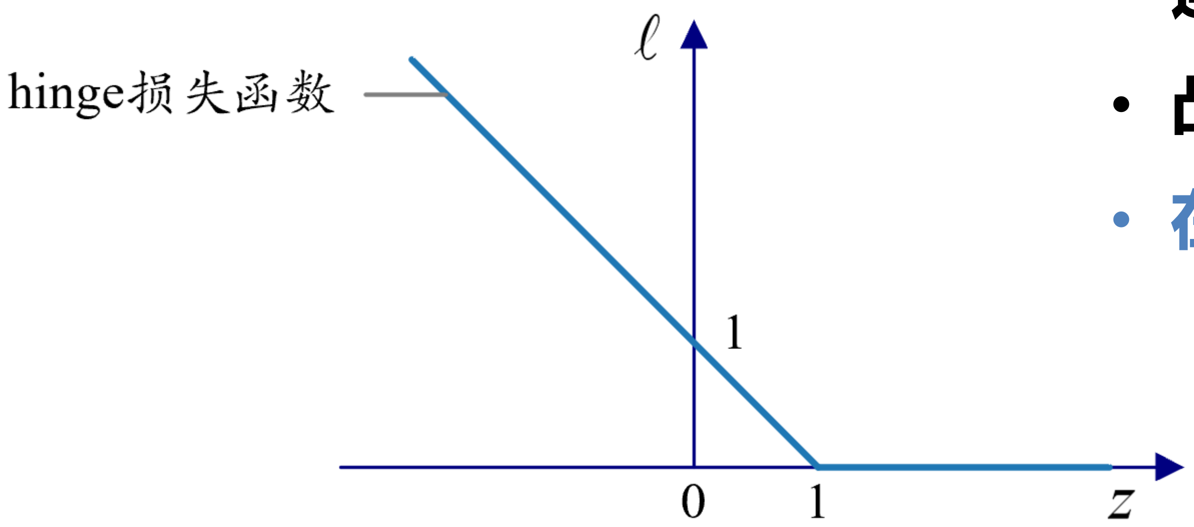
$$\ell_h(z) = \max(0, 1 - z)$$

软间隔线性支持向量机

❑ 硬间隔与软间隔

- hinge损失函数

$$\ell_h(z) = \max(0, 1 - z)$$



- **连续**
- **凸性**
- **在z=1处不可微!**

软间隔线性支持向量机

❑ 硬间隔与软间隔

- 使用hinge损失函数的线性支持向量机被称为**软间隔线性支持向量机**
- 数学模型如下：

$$\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{m}\max\left(0, 1 - y_i(\mathbf{w}^{\mathrm{T}}\boldsymbol{x}_i + b)\right)$$

- 如何解决目标函数在$z = 1$不可微的问题?
  - 引入松弛变量，对模型进行改造（一个重要的技巧）

软间隔线性支持向量机

❑ 硬间隔与软间隔

- 改进的数学模型如下：

$$\max\left(0, 1 - y_i(\mathbf{w}^\top \boldsymbol{x}_i + b)\right)$$

$$\min_{\mathbf{w},b,\xi} \frac{1}{2} \|\mathbf{w}\|^2 + C\sum_{i=1}^{m} \xi_i$$

$$\text{s.t.} \quad y_i(\mathbf{w}^\top \boldsymbol{x}_i + b) \geq 1 - \xi_i,$$

松弛变量 $\leftarrow$ $\xi_i \geq 0, \quad i = 1, 2, \ldots, m.$

对比 $\max\left(0, 1 - y_i(\mathbf{w}^\top \boldsymbol{x}_i + b)\right)$

软间隔线性支持向量机

软间隔线性支持向量机的对偶问题

$$\min_{\mathbf{w},b,\xi} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{m}\xi_i$$

$$\text{s.t.} \quad y_i(\mathbf{w}^\top \boldsymbol{x}_i + b) \geq 1 - \xi_i,$$

$$\xi_i \geq 0, \quad i = 1, 2, \ldots, m.$$

- 又出现了**复杂约束**
- 如何解决?
  - 找对偶问题!
  - 再次使用拉格朗日乘子法构造二次规划问题的对偶问题

试一试

写出该问题的拉格朗日函数。

软间隔线性支持向量机

软间隔线性支持向量机的对偶问题

- 对偶变量： $\boldsymbol{\alpha},\boldsymbol{\mu}$
- 不等式约束1： $y_i(\mathbf{w}^\top \boldsymbol{x}_i + b) \geq 1 - \xi_i \;, i = 1,2,\ldots,m$ $\qquad \boldsymbol{\alpha}$
- 不等式约束2： $\xi_i \geq 0 \;, i = 1,2,\ldots,m$ $\qquad \boldsymbol{\mu}$
- 拉格朗日函数：

$$\mathcal{L}(\mathbf{w},b,\boldsymbol{\alpha},\xi,\boldsymbol{\mu}) = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{m}\xi_i + \sum_{i=1}^{m}\alpha_i\left(1-\xi_i - y_i(\mathbf{w}^\top \boldsymbol{x}_i + b)\right) - \sum_{i=1}^{m}\mu_i\xi_i$$

再试一试

写出软间隔线性支持向量机的KKT条件。

软间隔线性支持向量机

❑ 软间隔线性支持向量机的对偶问题

- （1）基本约束

① 原问题的可行约束

$$\xi_i^* \geq 0 \quad ,i=1,2,\ldots,m$$

$$y_i(\mathbf{w}^{*,\mathrm{T}}\boldsymbol{x}_i+b^*)-1+\xi_i^* \geq 0 \quad ,i=1,2,\ldots,m$$

② 对偶变量的非负约束

$$\alpha_i^* \geq 0 \ ,\mu_i^* \geq 0 \quad ,i=1,2,\ldots,m$$

软间隔线性支持向量机

❑ 软间隔线性支持向量机的对偶问题

- （2）一阶偏导条件（KKT条件）

$$\mathcal{L}(\mathbf{w},b,\boldsymbol{\alpha},\boldsymbol{\xi},\boldsymbol{\mu})=\frac{1}{2}\|\mathbf{w}\|^2+C\sum_{i=1}^{m}\xi_i+\sum_{i=1}^{m}\alpha_i\left(1-\xi_i-y_i(\mathbf{w}^{\top}\boldsymbol{x}_i+b)\right)-\sum_{i=1}^{m}\mu_i\xi_i$$

$$\nabla_{\mathbf{w}}\mathcal{L}(\mathbf{w}^*,b^*,\boldsymbol{\alpha}^*,\boldsymbol{\xi}^*,\boldsymbol{\mu}^*)=\mathbf{w}^*-\sum_{i=1}^{m}\alpha_i^*y_i\boldsymbol{x}_i=\mathbf{0}$$

$$\nabla_{b}\mathcal{L}(\mathbf{w}^*,b^*,\boldsymbol{\alpha}^*,\boldsymbol{\xi}^*,\boldsymbol{\mu}^*)=\sum_{i=1}^{m}\alpha_i^*y_i=0$$

$$\nabla_{\xi}\mathcal{L}(\mathbf{w}^*,b^*,\boldsymbol{\alpha}^*,\boldsymbol{\xi}^*,\boldsymbol{\mu}^*)=\boldsymbol{C}-\boldsymbol{\alpha}^*-\boldsymbol{\mu}^*=0$$

软间隔线性支持向量机

❑ 软间隔线性支持向量机的对偶问题

- （3）互补松弛条件

$$\mathcal{L}(\mathbf{w},b,\boldsymbol{\alpha},\boldsymbol{\xi},\boldsymbol{\mu})=\frac{1}{2}\|\mathbf{w}\|^2+C\sum_{i=1}^{m}\xi_i+\sum_{i=1}^{m}\alpha_i\left(1-\xi_i-y_i(\mathbf{w}^\top\boldsymbol{x}_i+b)\right)-\sum_{i=1}^{m}\mu_i\xi_i$$

$$\Downarrow$$

$$\mu_i^*\xi_i^*=0 \quad ,i=1,2,\ldots,m$$

$$\alpha_i^*[y_i(\mathbf{w}^{*,\top}\boldsymbol{x}_i+b^*)-1+\xi_i^*]=0 \quad ,i=1,2,\ldots,m$$

软间隔线性支持向量机

软间隔线性支持向量机的对偶问题

- KKT条件:

$$\nabla_{\mathbf{w}}\mathcal{L}(\mathbf{w}^*,b^*,\boldsymbol{\alpha}^*,\boldsymbol{\xi}^*,\boldsymbol{\mu}^*)=\mathbf{w}^*-\sum_{i=1}^{m}\alpha_i^* y_i\boldsymbol{x}_i=\mathbf{0}$$

$$\nabla_{b}\mathcal{L}(\mathbf{w}^*,b^*,\boldsymbol{\alpha}^*,\boldsymbol{\xi}^*,\boldsymbol{\mu}^*)=\sum_{i=1}^{m}\alpha_i^* y_i=0$$

$$\nabla_{\xi}\mathcal{L}(\mathbf{w}^*,b^*,\boldsymbol{\alpha}^*,\boldsymbol{\xi}^*,\boldsymbol{\mu}^*)=\boldsymbol{C}-\boldsymbol{\alpha}^*-\boldsymbol{\mu}^*=0$$

$$\alpha_i^*\geq 0,\xi_i^*\geq 0,\mu_i^*\geq 0\quad ,i=1,2,\ldots,m$$

$$y_i(\mathbf{w}^{*,\mathsf{T}}\boldsymbol{x}_i+b^*)-1+\xi_i^*\geq 0\quad ,i=1,2,\ldots,m$$

$$\mu_i^*\xi_i^*=0\quad ,i=1,2,\ldots,m$$

$$\alpha_i^*[y_i(\mathbf{w}^{*,\mathsf{T}}\boldsymbol{x}_i+b^*)-1+\xi_i^*]=0\quad ,i=1,2,\ldots,m$$

软间隔线性支持向量机

❑ 软间隔线性支持向量机的对偶问题

- 拉格朗日对偶函数：

$$g(\boldsymbol{\alpha},\boldsymbol{\mu}) = \inf_{\mathbf{w},b,\xi} \mathcal{L}(\mathbf{w},b,\boldsymbol{\alpha},\boldsymbol{\xi},\boldsymbol{\mu})$$

- 根据一阶偏导最优性条件，

$$\nabla_{\mathbf{w}}\mathcal{L}(\mathbf{w}^*,b^*,\boldsymbol{\alpha}^*,\boldsymbol{\xi}^*,\boldsymbol{\mu}^*) = \mathbf{w}^* - \sum_{i=1}^{m}\alpha_i^* y_i \boldsymbol{x}_i = \mathbf{0}$$

$$\nabla_{b}\mathcal{L}(\mathbf{w}^*,b^*,\boldsymbol{\alpha}^*,\boldsymbol{\xi}^*,\boldsymbol{\mu}^*) = \sum_{i=1}^{m}\alpha_i^* y_i = 0$$

$$\nabla_{\xi}\mathcal{L}(\mathbf{w}^*,b^*,\boldsymbol{\alpha}^*,\boldsymbol{\xi}^*,\boldsymbol{\mu}^*) = \boldsymbol{C} - \boldsymbol{\alpha}^* - \boldsymbol{\mu}^* = 0$$

$\Rightarrow$

$$\mathbf{w}^* = \sum_{i=1}^{m}\alpha_i^* y_i \boldsymbol{x}_i$$

$$0 = \sum_{i=1}^{m}\alpha_i^* y_i$$

$$\mu_i = C - \alpha_i \geq 0$$

教材书：144页的详细推导

$\Downarrow$ 代入拉格朗日对偶函数并化简

$$g(\boldsymbol{\alpha},\boldsymbol{\mu}) = \sum_{i=1}^{m}\alpha_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j y_i y_j \boldsymbol{x}_i^{\mathsf{T}}\boldsymbol{x}_j$$

软间隔线性支持向量机

软间隔线性支持向量机的对偶问题

- 拉格朗日对偶问题:

$$\max_{\boldsymbol{\alpha}} \sum_{i=1}^{m}\alpha_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j y_i y_j \boldsymbol{x}_i^{\top}\boldsymbol{x}_j$$

$$\text{s.t.} \quad \sum_{i=1}^{m}\alpha_i y_i = 0,$$

$$\boxed{\begin{array}{c}\mu_i = C - \alpha_i \geq 0 \\ \alpha_i \geq 0\end{array}} \Longrightarrow \quad 0 \leq \alpha_i \leq C, \quad i = 1, 2, \ldots, m.$$

软间隔线性支持向量机

软间隔线性支持向量机的对偶问题

- 根据互补松弛条件:

$$\alpha_i^*[y_i(\mathbf{w}^{*,\mathrm{T}}\boldsymbol{x}_i+b^*)-1+\xi_i^*]=0 \quad ,i=1,2,\ldots,m$$

支持向量 $\Leftrightarrow \alpha_i^*>0$

① $\alpha_i^*=0$

$\boxed{\mu_i=C-\alpha_i\geq 0} \Rightarrow \mu_i=C>0$

$\boxed{\mu_i^*\xi_i^*=0} \Rightarrow \xi_i^*=\max(0,1-y_i(\mathbf{w}^{*,\mathrm{T}}\boldsymbol{x}_i+b^*))=0 \Rightarrow y_i(\mathbf{w}^{*,\mathrm{T}}\boldsymbol{x}_i+b^*)\geq 1$

hinge损失为0

样本可能在间隔边界上 $y_i(\mathbf{w}^{*,\mathrm{T}}\boldsymbol{x}_i+b^*)=1$

也可能在边界外侧 $y_i(\mathbf{w}^{*,\mathrm{T}}\boldsymbol{x}_i+b^*)>1$

软间隔线性支持向量机

❑ 软间隔线性支持向量机的对偶问题

- 根据互补松弛条件:

$$\alpha_i^*[y_i(\mathbf{w}^{*,\top}\boldsymbol{x}_i+b^*)-1+\xi_i^*]=0 \quad ,i=1,2,\ldots,m$$

$$\Downarrow$$

支持向量 $\Leftrightarrow \alpha_i^* > 0$

② $0<\alpha_i^*<C \Rightarrow y_i(\mathbf{w}^{*,\top}\boldsymbol{x}_i+b^*)-1+\xi_i^*=0$

$\mu_i=C-\alpha_i\geq 0 \Rightarrow \mu_i>0$

$\mu_i^*\xi_i^*=0 \Rightarrow \xi_i^*=\max(0,1-y_i(\mathbf{w}^{*,\top}\boldsymbol{x}_i+b^*))=0 \Rightarrow y_i(\mathbf{w}^{*,\top}\boldsymbol{x}_i+b^*)-1=0$

hinge损失为0 样本恰在间隔边界上

软间隔线性支持向量机

❑ 软间隔线性支持向量机的对偶问题

- 根据互补松弛条件:

$$\alpha_i^*[y_i(\mathbf{w}^{*,\top}\boldsymbol{x}_i+b^*)-1+\xi_i^*]=0 \quad ,i=1,2,\dots,m$$

支持向量 $\Leftrightarrow \alpha_i^*>0$

③ $\alpha_i^*=C \Rightarrow y_i(\mathbf{w}^{*,\top}\boldsymbol{x}_i+b^*)-1+\xi_i^*=0$

$\boxed{\mu_i=C-\alpha_i\geq 0} \Rightarrow \mu_i=0$ $\quad \boxed{\mu_i^*\xi_i^*=0} \Rightarrow \xi_i^*\geq 0$

若 $0\leq\xi_i^*<1 \Rightarrow 0<y_i(\mathbf{w}^{*,\top}\boldsymbol{x}_i+b^*)=1-\xi_i^*\leq 1$ 分类正确，样本在间隔边界与分隔超平面之间

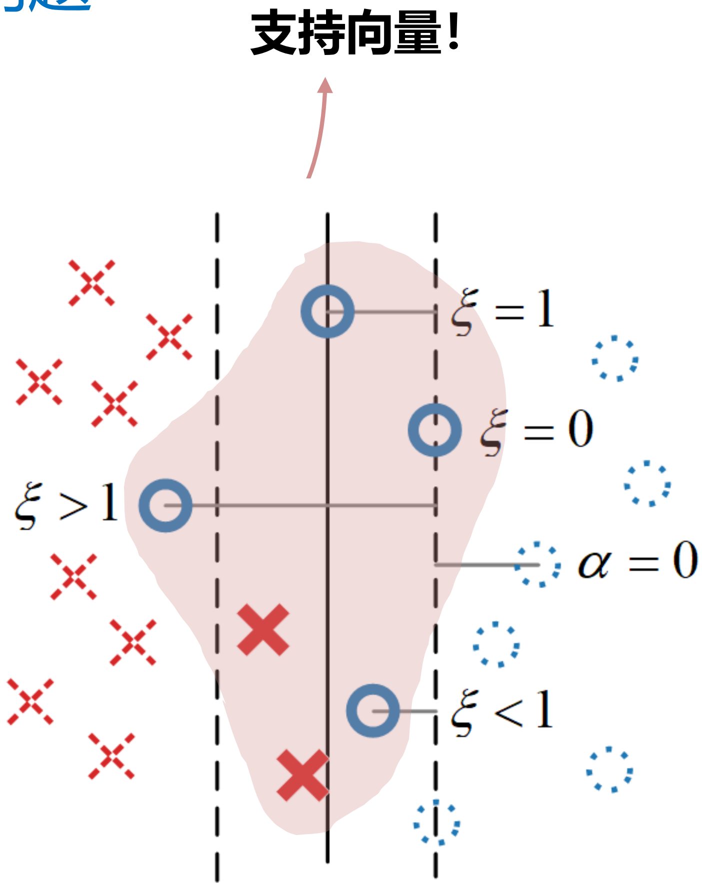
若 $\xi_i^*=1 \Rightarrow y_i(\mathbf{w}^{*,\top}\boldsymbol{x}_i+b^*)=1-\xi_i^*=0$ 样本恰在分隔超平面上

若 $\xi_i^*>1 \Rightarrow y_i(\mathbf{w}^{*,\top}\boldsymbol{x}_i+b^*)=1-\xi_i^*<0$ 分类错误，样本在分隔超平面分类错误的一侧

软间隔线性支持向量机

❑ 软间隔线性支持向量机的对偶问题

- 软间隔支持向量总结：



(1) $\alpha_i^* = 0, \xi_i^* = 0, y_i(\mathbf{w}^{*,\top}\boldsymbol{x}_i + b^*) \geq 1$

非支持向量，hinge损失=0

(2) $0 < \alpha_i^* < C, \xi_i^* = 0, y_i(\mathbf{w}^{*,\top}\boldsymbol{x}_i + b^*) = 1$

支持向量，hinge损失=0

(3) $\alpha_i^* = C, \xi_i^* \geq 0, y_i(\mathbf{w}^{*,\top}\boldsymbol{x}_i + b^*) \leq 1$

支持向量，hinge损失≥0

软间隔线性支持向量机

❑ 软间隔线性支持向量机的对偶问题

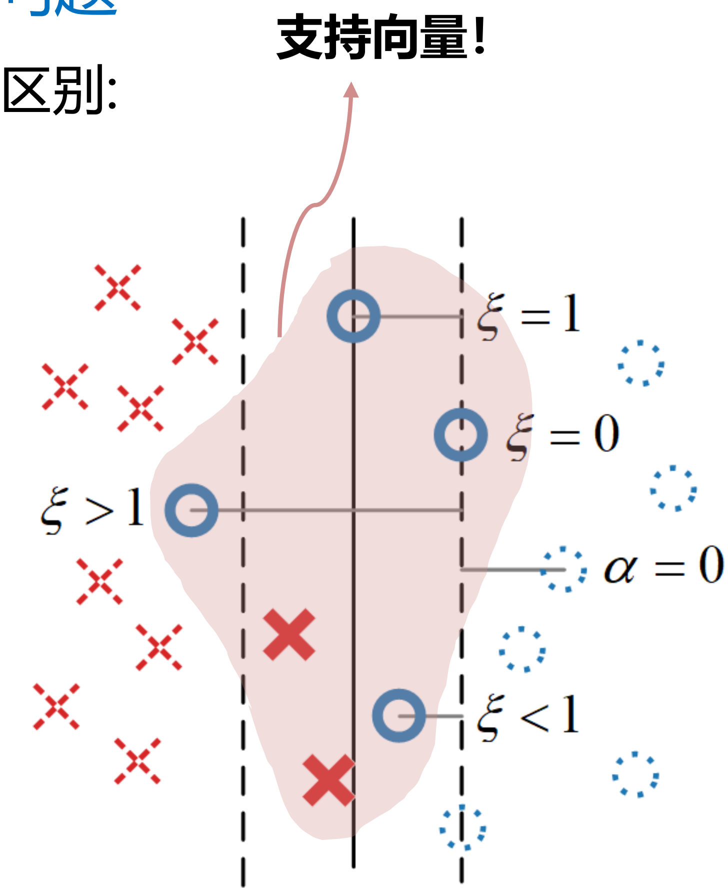
- 软间隔支持向量与硬间隔支持向量的区别:

硬间隔

- 支持向量出现在间隔的**边界**上

软间隔

- 支持向量可能在间隔**边界与分隔超平面**之间
- 也可能在分隔超平面**分类错误的一侧**



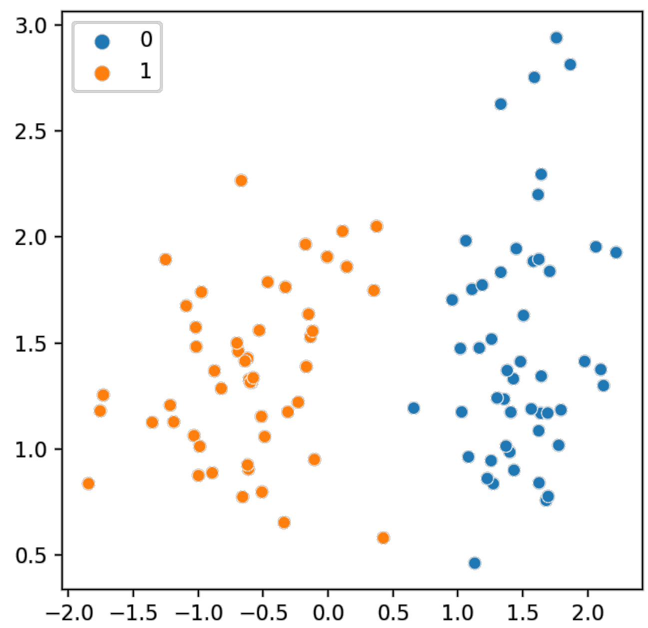
软间隔线性支持向量机

❑ 试一试

- 使用软间隔线性支持向量机对案例数据进行分类。

1. 生成案例数据

```
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.datasets import make_classification

X, y = make_classification(
    n_samples=100, n_features=2,
    n_redundant=0, n_informative=2,
    random_state=1, n_clusters_per_class=1
)
rng = np.random.RandomState(2)
X += rng.uniform(size=X.shape)

plt.figure(figsize=(5, 5))
sns.scatterplot(
    x=X[:,0], y=X[:,1], hue=y, legend='full')
plt.show()
```



软间隔线性支持向量机

❑ 试一试

- 使用软间隔线性支持向量机对案例数据进行分类。

2. 训练模型

```
from sklearn.preprocessing import StandardScaler
from sklearn.svm import SVC

# 数据标准化
X_ = StandardScaler().fit_transform(X)
data = X_, y

# 构建软间隔线性支持向量机，训练模型
clf = SVC(C=1, kernel='linear', random_state=3)
clf.fit(X_, y)
```

考虑了参数C!

软间隔线性支持向量机

❑ 试一试

- 使用软间隔线性支持向量机对案例数据进行分类。

3. 查看分类结果

```
def plot_predicted_proba(data, clf, h=0.02):
    X, y = data

    x_min, x_max = X[:, 0].min() - .5, X[:, 0].max() + .5
    y_min, y_max = X[:, 1].min() - .5, X[:, 1].max() + .5
    xx, yy = np.meshgrid(np.arange(x_min, x_max, h),
                         np.arange(y_min, y_max, h))
    Z = clf.predict(np.c_[xx.ravel(), yy.ravel()])
    Z = Z.reshape(xx.shape)

    plt.figure(figsize=(5, 5))
    plt.contourf(xx, yy, Z, cmap=plt.cm.RdBu, alpha=.8)
    sns.scatterplot(x=X[:, 0], y=X[:, 1], hue=y, legend='full')
    plt.xlim(x_min, x_max)
    plt.ylim(y_min, y_max)
    plt.show()

plot_predicted_proba(data, clf)
```

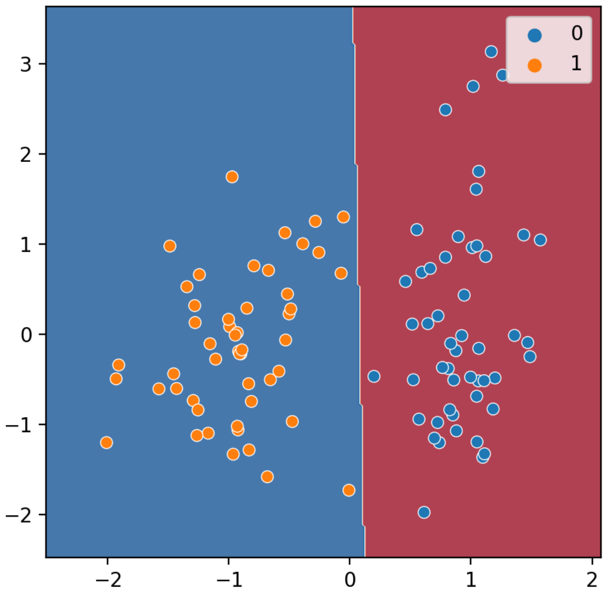
软间隔线性支持向量机

❑ 试一试

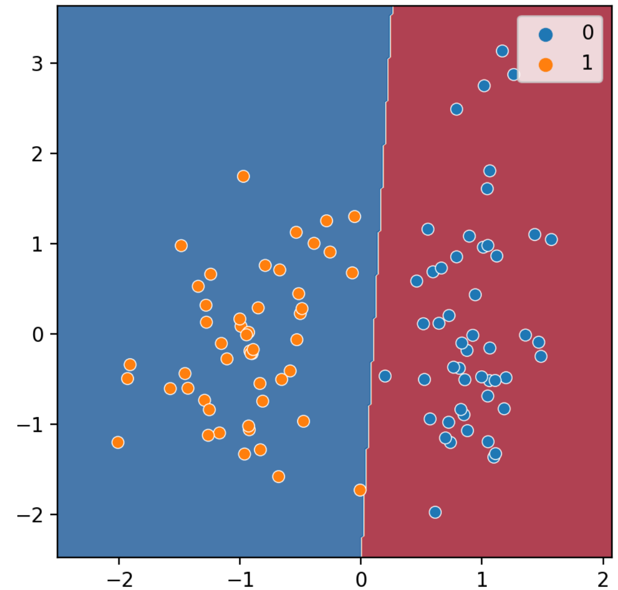
- 使用软间隔线性支持向量机对案例数据进行分类。

3. 查看分类结果

硬间隔
SVM
分类结果



软间隔
SVM
分类结果



对于线性可分且无明显噪声的数据，软间隔与硬间隔SVM的结果区别不大

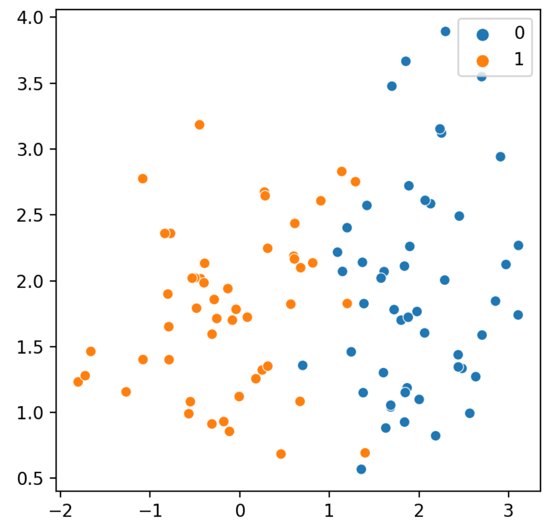
软间隔线性支持向量机

❑ 试一试

- 使用软间隔线性支持向量机对案例数据进行分类。

尝试更换一组训练数据

```
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.datasets import make_classification

X, y = make_classification(
    n_samples=100, n_features=2,
    n_redundant=0, n_informative=2,
    random_state=1, n_clusters_per_class=1
)
rng = np.random.RandomState(2)
X += rng.uniform(size=X.shape) * 2

plt.figure(figsize=(5, 5))
sns.scatterplot(
    x=X[:,0], y=X[:,1], hue=y, legend='full')
plt.show()
```



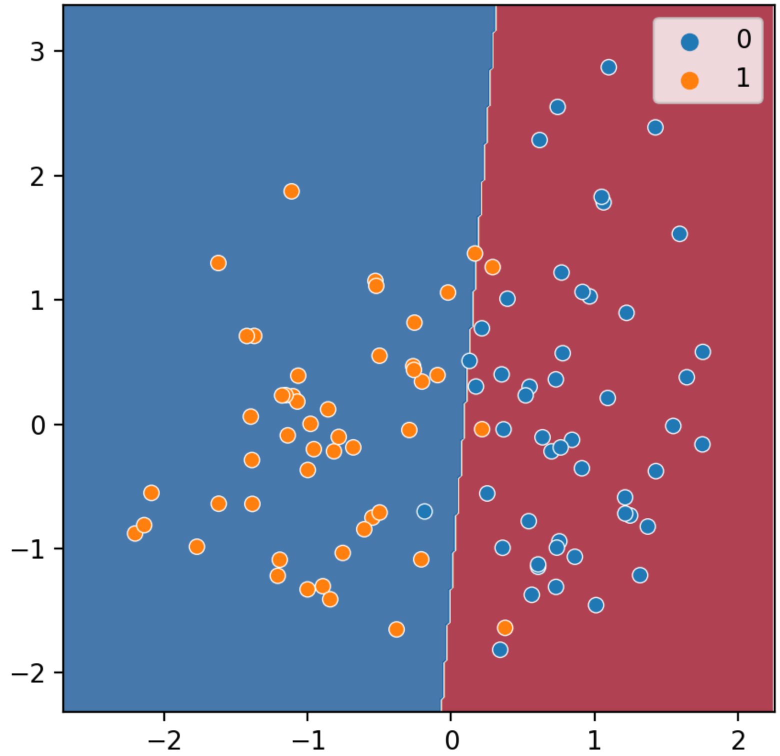
线性不可分！

软间隔线性支持向量机

❑ 试一试

- 使用软间隔线性支持向量机对案例数据进行分类。

尝试更换一组训练数据



- **线性可分支持向量机无法训练**
- **软间隔线性支持向量机可以给出合理结果**

支持向量机

❑ 学习目标

- 了解支持向量机的基本概念
- 掌握线性可分支持向量机模型
- 掌握软间隔支持向量机模型
- **理解非线性支持向量机中的核函数**
- 介绍序列最小优化算法（阅读内容）

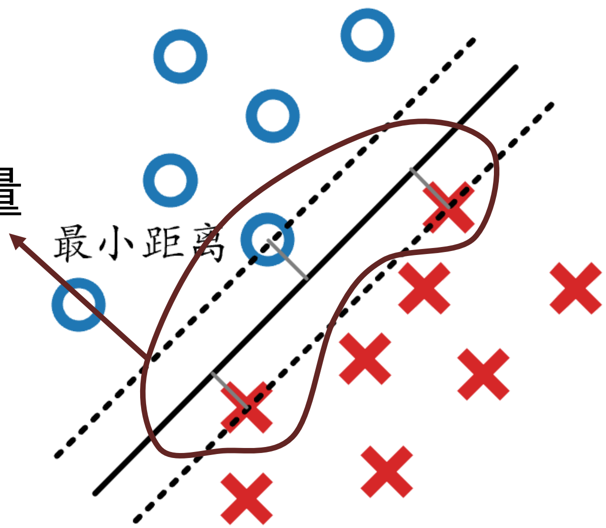
非线性支持向量机

❑ 核技巧（kernel trick）

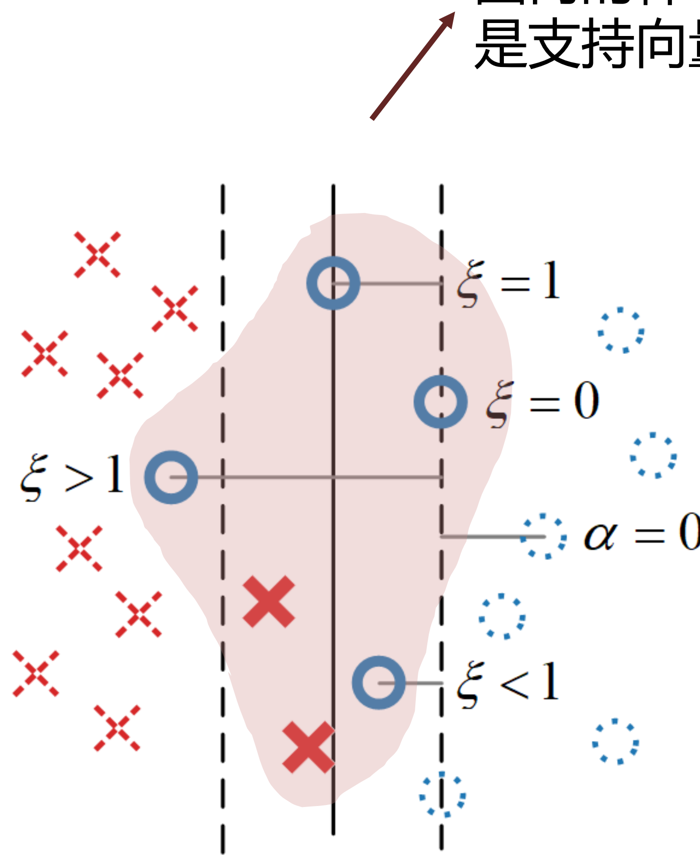
- 线性支持向量机能够处理的问题:

圈内的样本都是支持向量

支持向量



完全线性可分

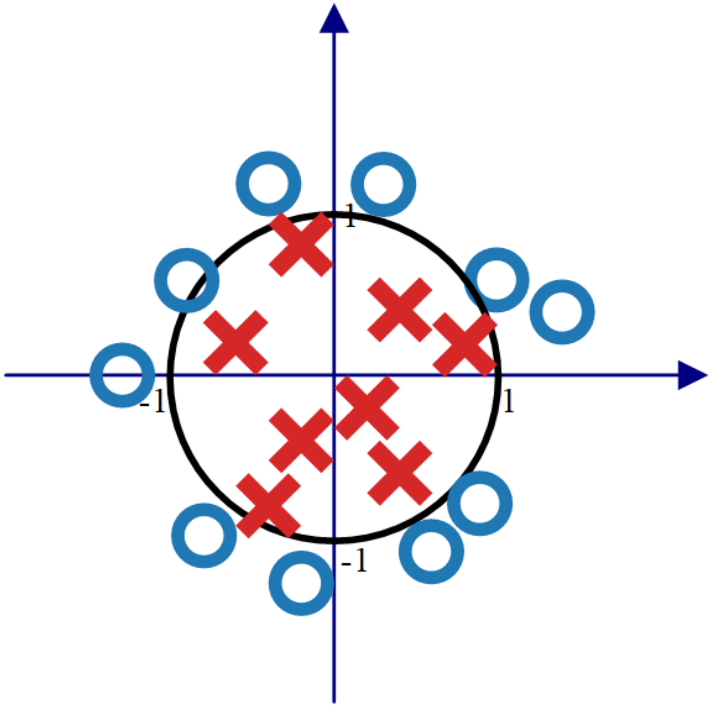


近似线性可分

非线性支持向量机

❑ 核技巧

- 线性支持向量机不能处理的问题:

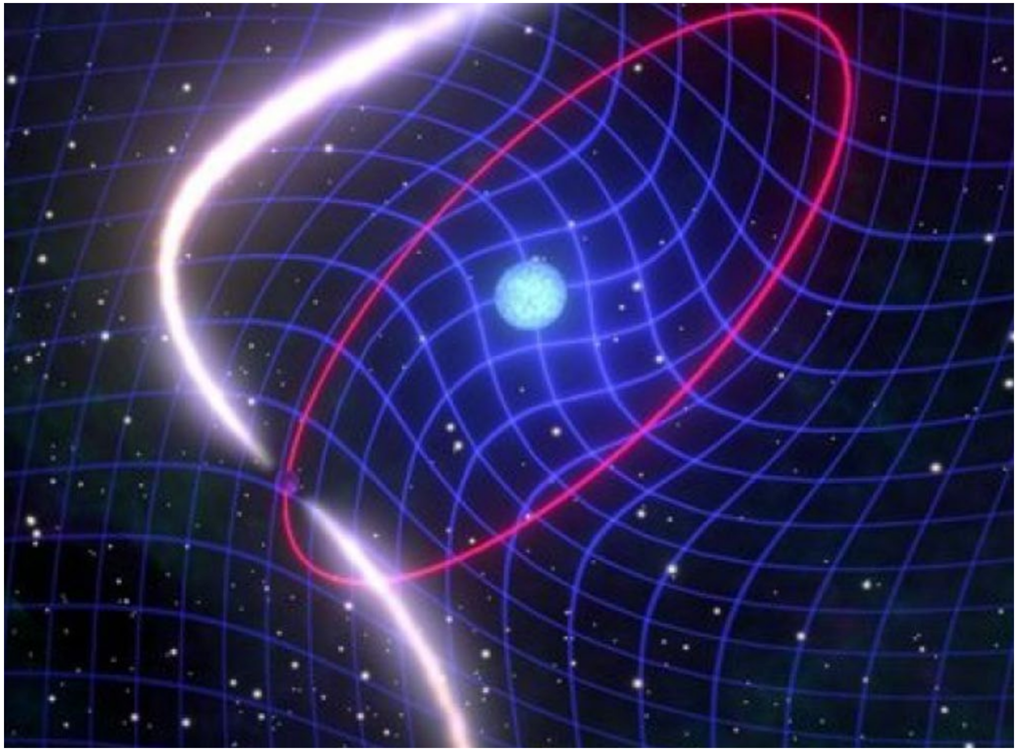


如何处理?

非线性支持向量机

❑ 核技巧

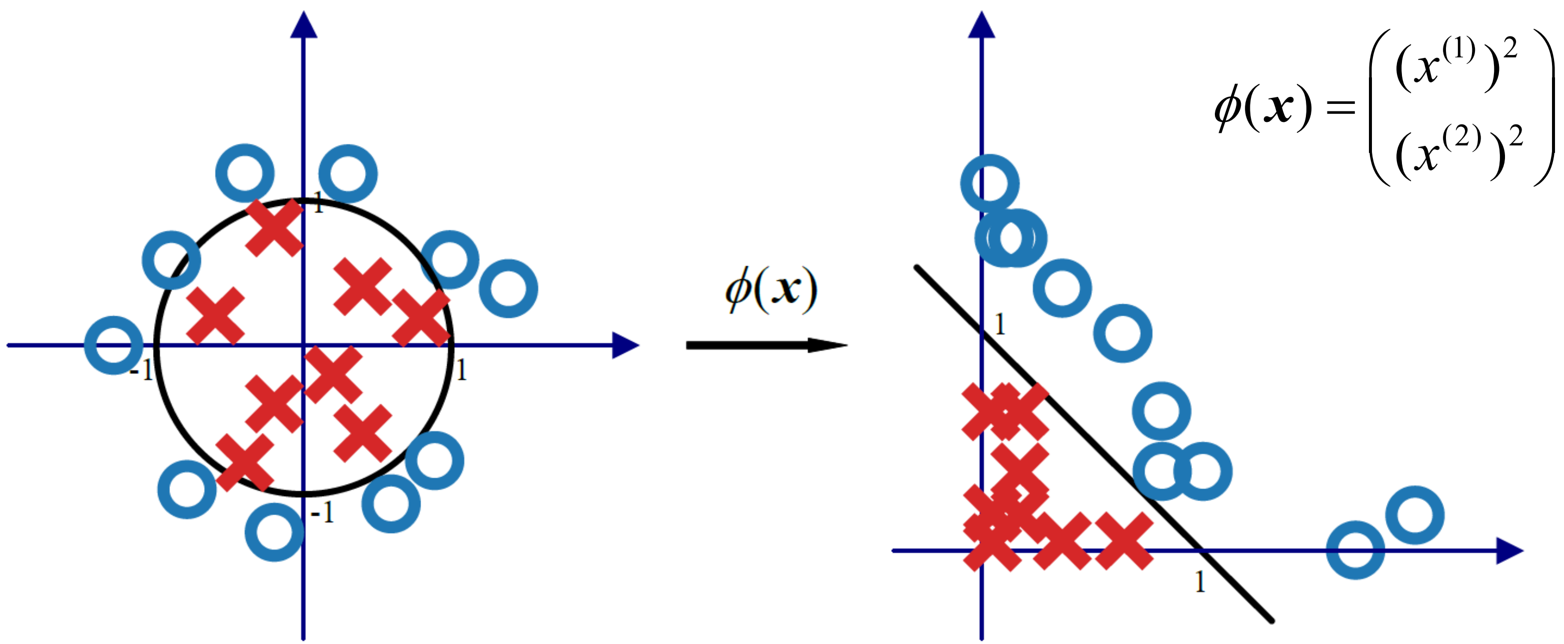
- 空间扭曲！



非线性支持向量机

核技巧

- 构造映射函数$\phi(\cdot): \mathbb{R}^s \rightarrow \mathbb{R}^t$对原特征空间做变换



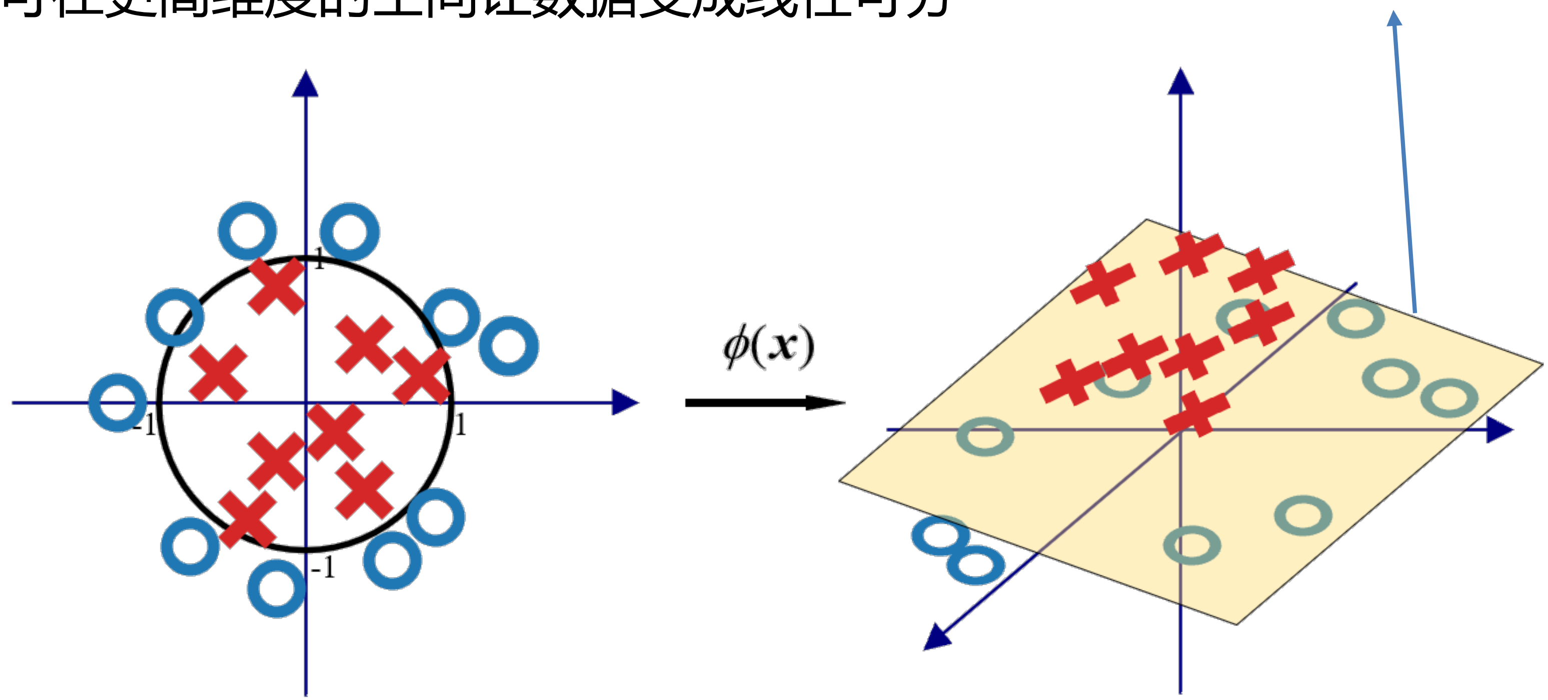
非线性支持向量机

❑ 核技巧

- 原空间和变换后空间的维度可不一致
- 可在更高维度的空间让数据变成线性可分

分隔超平面方程

$$f(\boldsymbol{x}) = \mathbf{w}^{\mathrm{T}} \phi(\boldsymbol{x}) + b$$



非线性支持向量机

❑ 核技巧

- 对数据变换后，可使用线性支持向量机求解

$$\min_{\mathbf{w},b,\xi} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{m}\xi_i$$

$$\text{s.t.} \quad y_i(\mathbf{w}^{\mathrm{T}}\phi(\boldsymbol{x}_i) + b) \geq 1 - \xi_i,$$

$$\xi_i \geq 0, \quad i = 1,2,\ldots,m.$$

非线性支持向量机

核技巧

- 对偶问题

$$\max_{\boldsymbol{\alpha}} \sum_{i=1}^{m} \alpha_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j \phi(\boldsymbol{x}_i)^{\mathrm{T}} \phi(\boldsymbol{x}_j)$$

$$\text{s.t.} \quad \sum_{i=1}^{m} \alpha_i y_i = 0,$$

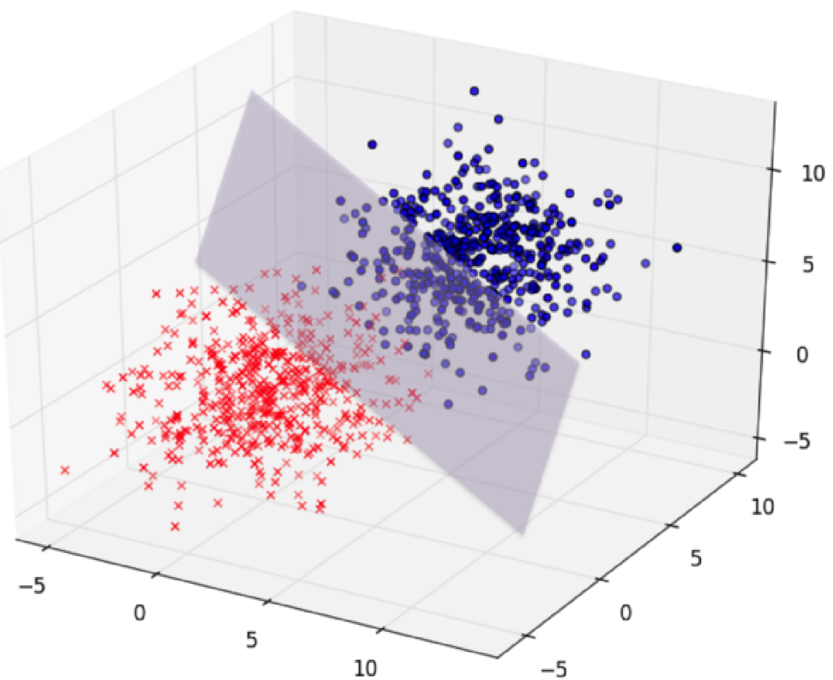
$$0 \le \alpha_i \le C, \quad i = 1, 2, \ldots, m.$$

输入样本在新特征空间中的**内积**! $\langle \phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j) \rangle$

非线性支持向量机

核技巧

- 使用映射函数$\phi(\cdot)$并不能一劳永逸地解决所有问题。
  - 输入数据维度较低时，$\phi(\cdot)$容易确定
  - **高维**输入**难以找到**合适的$\phi(\cdot)$
  - **高维**空间下**计算量**巨大



非线性支持向量机

核技巧

- 重新观察非线性支持向量机的对偶问题：
  - 是否有必要知道确切的映射函数$\phi(\cdot)$?

$$\max_{\boldsymbol{\alpha}} \sum_{i=1}^{m}\alpha_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j y_i y_j \phi(\boldsymbol{x}_i)^{\mathrm{T}}\phi(\boldsymbol{x}_j)$$

$$\text{s.t.} \quad \sum_{i=1}^{m}\alpha_i y_i = 0,$$

$$0 \le \alpha_i \le C, \quad i = 1,2,\ldots,m.$$

非线性支持向量机

❑ 核技巧

- 替代：$\kappa(x_i, x_j) = \phi(x_i)^\top \phi(x_j)$
  - 跳过先映射、再求内积的过程

$$\max_{\boldsymbol{\alpha}} \sum_{i=1}^{m} \alpha_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)$$

$$\text{s.t.} \quad \sum_{i=1}^{m} \alpha_i y_i = 0,$$

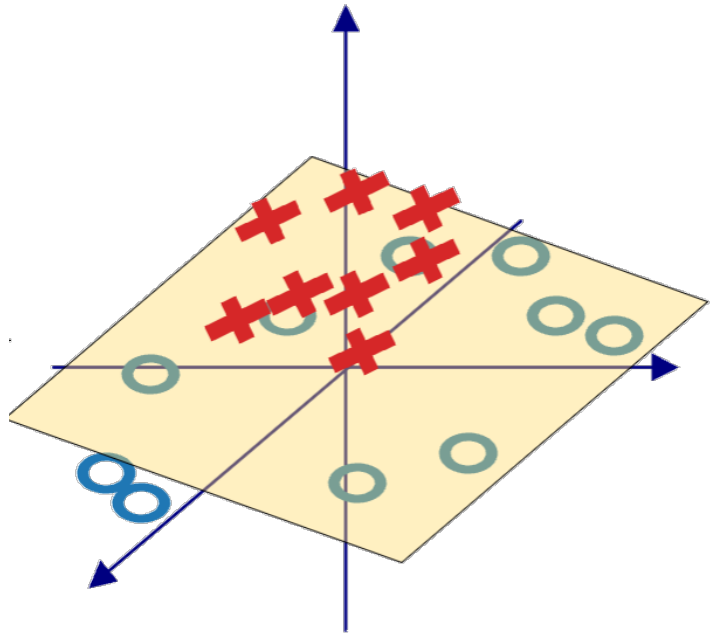
$$0 \leq \alpha_i \leq C, \quad i = 1, 2, \ldots, m.$$

非线性支持向量机

❑ 核技巧

- 类似地，可以改写分隔超平面的方程：

$$
\begin{aligned}
f(\boldsymbol{x}) &= \mathbf{w}^{*,\top}\phi(\boldsymbol{x}) + b^* \\
&= \sum_{i=1}^{m} \alpha_i y_i \phi(\boldsymbol{x}_i)^\top \phi(\boldsymbol{x}) + b^* \\
&= \sum_{i=1}^{m} \alpha_i y_i \kappa(\boldsymbol{x}_i, \boldsymbol{x}) + b^*
\end{aligned}
$$

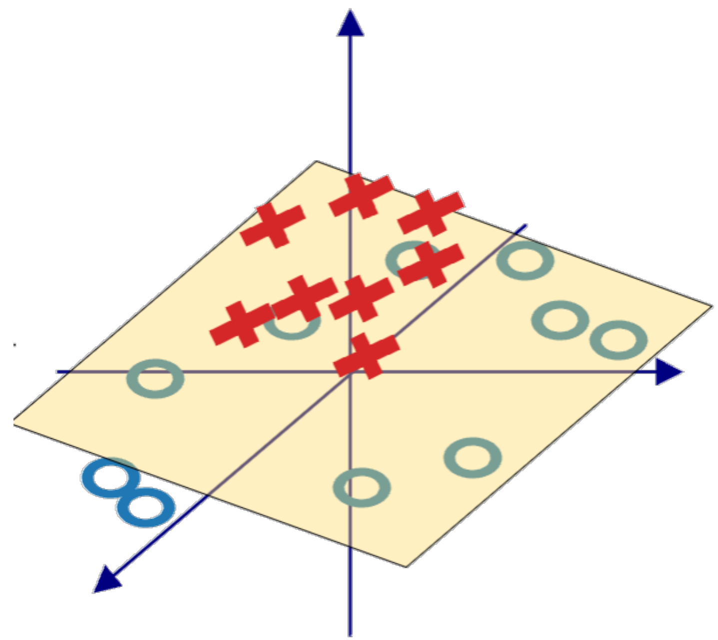


非线性支持向量机

❑ 核技巧

想一想

核函数的作用是什么?



- 核函数替代了对任意两个向量“做映射—求内积”的繁琐运算

非线性支持向量机

❑ 核技巧

- 核函数的定义
  - 若存在一个从输入空间 $\mathcal{X}$ 到特征空间 $\mathcal{H}$ 的映射 $\phi\colon \mathcal{X} \to \mathcal{H}$，对于任意 $\boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathcal{X}$，有函数 $\kappa\colon \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ 满足

$$\kappa(\boldsymbol{x}_1, \boldsymbol{x}_2) = \langle \phi(\boldsymbol{x}_1), \phi(\boldsymbol{x}_2) \rangle = \phi(\boldsymbol{x}_1)^\top \phi(\boldsymbol{x}_2)$$

  - 则称该函数 κ 为一个核函数。

其中：$\langle\cdot,\cdot\rangle$为内积运算，$\mathbb{R}$为实数集

非线性支持向量机

❑ 核技巧

- 核函数的性质
  - 首先定义核函数$\kappa(\cdot,\cdot)$关于样本$(x_1, x_2, \dots, x_n)$的核矩阵：

$$\boldsymbol{K}=\begin{pmatrix} \kappa(x_1,x_1) & \kappa(x_1,x_2) & \dots & \kappa(x_1,x_n) \\ \kappa(x_2,x_1) & \kappa(x_2,x_2) & \dots & \kappa(x_2,x_n) \\ \vdots & \vdots & \ddots & \vdots \\ \kappa(x_n,x_1) & \kappa(x_n,x_2) & \dots & \kappa(x_n,x_n) \end{pmatrix}$$

非线性支持向量机

核技巧

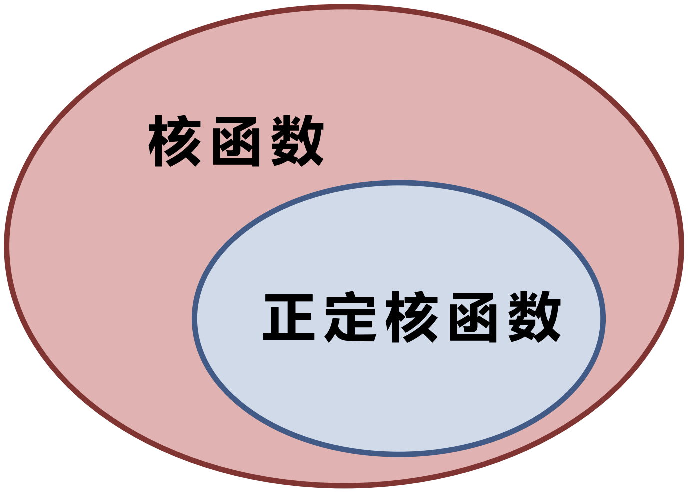
- **正定核函数**的充要条件：
  - 给定对称函数$\kappa: \mathcal{X} \times \mathcal{X} \to \mathbb{R}$，则该函数是正定核函数的充要条件是，对任意长度的一组数据$(x_1, x_2, \ldots) \in \mathcal{X}$，其核矩阵$\boldsymbol{K}$半正定。

* 存在部分核函数不是正定核函数

* **对称函数：**

$$\kappa(x_i, x_j) = \kappa(x_j, x_i)$$

* 延伸阅读：Mercer定理（关于正定核函数更严谨的定义）



非线性支持向量机

❑ 常用核函数

- 如何设计**核函数**?
  - 直接设计新的核函数难度很大：帽子核
  - 常用的核函数包括：
    - 线性核、多项式核、**高斯核**、有理二次核、Matérn核、正弦平方核、神经网络核等。
  - 针对非结构化输入，如文本等数据，常用的核函数有：
    - 字符串核、Fisher核等。

Liu, Z., Lyu, C., Huo, J., Wang, S., and Chen, J.*, 2022. Gaussian Process Regression for Transportation System Estimation and Prediction Problems: the Deformation and a Hat Kernel, *IEEE Transactions on Intelligent Transportation Systems*, DOI:[10.1109/TITS.2022.3155527](https://doi.org/10.1109/TITS.2022.3155527).

非线性支持向量机

❑ 常用核函数

- **线性核函数**
  - 不做任何映射，特征空间不做变换。

$$\kappa_{\text{linear}}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{x}_i^{\top} \boldsymbol{x}_j$$

- **多项式核函数**

$$\kappa_{\text{poly},d}(\boldsymbol{x}_i, \boldsymbol{x}_j) = (\boldsymbol{x}_i^{\top} \boldsymbol{x}_j)^d$$

试一试

当多项式次数为2，特征维度也为2时，能否找到多项式核的映射$\phi(\cdot)$？

非线性支持向量机

常用核函数

- **多项式核函数**
  - 不妨记 $\boldsymbol{x}_i = (x_i^{(1)}, x_i^{(2)})^\top$

$$
\begin{aligned}
\kappa_{\text{poly},2}(\boldsymbol{x}_i, \boldsymbol{x}_j) &= (\boldsymbol{x}_i^\top \boldsymbol{x}_j)^2 \\
&= \left(x_i^{(1)} x_j^{(1)} + x_i^{(2)} x_j^{(2)}\right)^2 \\
&= \left(x_i^{(1)} x_j^{(1)}\right)^2 + 2x_i^{(1)} x_j^{(1)} x_i^{(2)} x_j^{(2)} + \left(x_i^{(2)} x_j^{(2)}\right)^2 \\
&= \left(\left(x_i^{(1)}\right)^2, \sqrt{2}x_i^{(1)} x_i^{(2)}, \left(x_i^{(2)}\right)^2\right)^\top \left(\left(x_j^{(1)}\right)^2, \sqrt{2}x_j^{(1)} x_j^{(2)}, \left(x_j^{(2)}\right)^2\right)
\end{aligned}
$$

$$
\Downarrow
$$

$$
\phi(\boldsymbol{x}_i) = \left(\left(x_i^{(1)}\right)^2, \sqrt{2}x_i^{(1)} x_i^{(2)}, \left(x_i^{(2)}\right)^2\right)
$$

非线性支持向量机

❑ 常用核函数

- **高斯核函数 / RBF(Radial Basis Function)核函数**

$$\kappa_{\mathrm{rbf}}(\boldsymbol{x}_i,\boldsymbol{x}_j)=\exp\left(-\frac{\|\boldsymbol{x}_i-\boldsymbol{x}_j\|^2}{2\sigma^2}\right)$$

  - σ为带宽，是高斯核函数的一个超参数。
  - 高斯核可改造为各向异性的核函数，以适应数据在不同特征维度上变化程度的差异。

$$\kappa_{\mathrm{rbf}}(\boldsymbol{x}_i,\boldsymbol{x}_j)=\exp\left(-\frac{1}{2}(\boldsymbol{x}_i-\boldsymbol{x}_j)^{\top}\Sigma(\boldsymbol{x}_i-\boldsymbol{x}_j)\right)$$

$$\Sigma=\mathrm{diag}(\sigma_1^2,\sigma_2^2,\ldots,\sigma_k^2)$$

    - Σ为对角阵，k为特征维度。

非线性支持向量机

❑ 常用核函数

- **神经网络核函数 / Sigmoid核函数**

$$\kappa(\boldsymbol{x}_i,\boldsymbol{x}_j)=\tanh(\beta\boldsymbol{x}_i^{\top}\boldsymbol{x}_j+\theta)$$

- 超参数β和θ需要满足$\beta>0,\theta<0$。
- 与线性核和多项式核一样，神经网络核可以写成关于$x_i^{\top}x_j$的函数，此类核函数又称为点积核。
- **点积核正定性定理**：一个点积核$\kappa(x_i,x_j)$是正定核，当且仅当它的泰勒级数$\sum_{n=0}^{\infty}a_n\left(x_i^{\top}x_j\right)^n$中，有任意$a_n>0$。

非线性支持向量机

常用核函数

- **神经网络核的正定性**
  - 令$t = x_i^{\mathrm{T}} x_j$，设$\beta = 1$。
  - 神经网络核的三阶麦克劳林级数:

$$
\begin{aligned}
f\left(\boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{x}_j\right) = f(t) \\
&= \tanh(t+\theta) \\
&= \tanh\theta + \frac{1}{\cosh^2\theta} t - \frac{\tanh\theta}{\cosh^2\theta} t^2 \\
&\quad -\frac{1}{3}\left(1-\tanh^2\theta\right)\left(1-3\tanh^2\theta\right) t^3 + o(t^4)
\end{aligned}
$$

- $\theta \geq 0$
- $\theta \in \mathbb{R}$
- $\theta \leq 0$
- $\theta \in \left(-\infty, \ln\frac{2-\sqrt{3}}{2}\right] \cup \left[\ln\frac{2-\sqrt{3}}{2}, +\infty\right)$

- 假设麦克劳林展开各项系数均非负
- θ取值范围矛盾
- 该核函数**非正定核**

非线性支持向量机

❑ 常用核函数

- **核函数构造规则**
  - 除了基础的核函数，也可以通过一些规则来构造新的核函数
  - 已知$\kappa_1(\cdot,\cdot)$和$\kappa_2(\cdot,\cdot)$均为核函数，则它们的**数乘、求和、乘积**仍为核函数。

$$\kappa(\boldsymbol{x}_1,\boldsymbol{x}_2)=\begin{cases} c\kappa_1(\boldsymbol{x}_1,\boldsymbol{x}_2) \\ \kappa_1(\boldsymbol{x}_1,\boldsymbol{x}_2)+\kappa_2(\boldsymbol{x}_1,\boldsymbol{x}_2) \\ \kappa_1(\boldsymbol{x}_1,\boldsymbol{x}_2)\cdot\kappa_2(\boldsymbol{x}_1,\boldsymbol{x}_2) \end{cases}$$

非线性支持向量机

❑ 常用核函数

- **核函数选择**
  - 训练样本较少，特征维度较高时，较为简单的核函数通常足以拟合训练集数据，此时复杂的核函数极易产生过拟合现象；
  - 训练样本较多，特征维度较低时，更适合使用较为复杂的非线性核函数，它们能够更充分地挖掘数据之间的非线性关系；
  - 训练样本极多，特征维度不高时，适合使用线性核函数，甚至更简单的逻辑回归模型，否则运算效率难以接受；
  - 结合数据特点选择，如对于时间序列预测问题，可使用具有周期性特点的正弦平方核

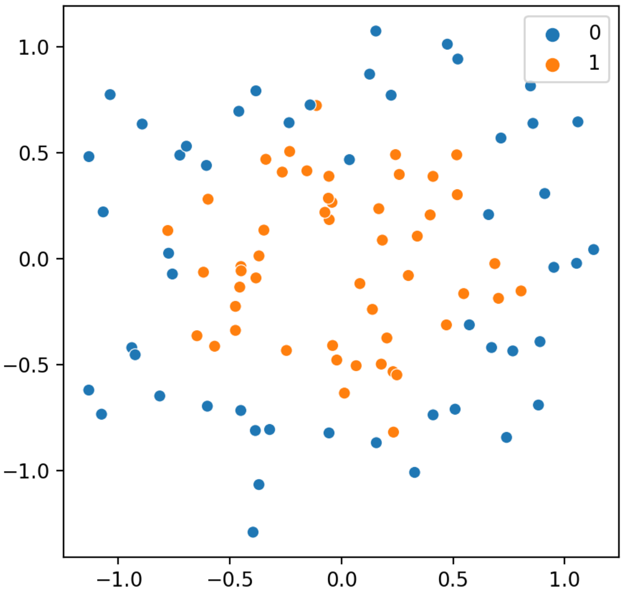
非线性支持向量机

❑ 试一试

- 使用非线性支持向量机对案例数据进行分类。

1. 生成案例数据

```
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.datasets import make_circles

X, y = make_circles(
    n_samples=100,
    noise=0.2, factor=0.5,
    random_state=1
)

plt.figure(figsize=(5, 5))
sns.scatterplot(
    x=X[:,0], y=X[:,1], hue=y, legend='full')
plt.show()
```

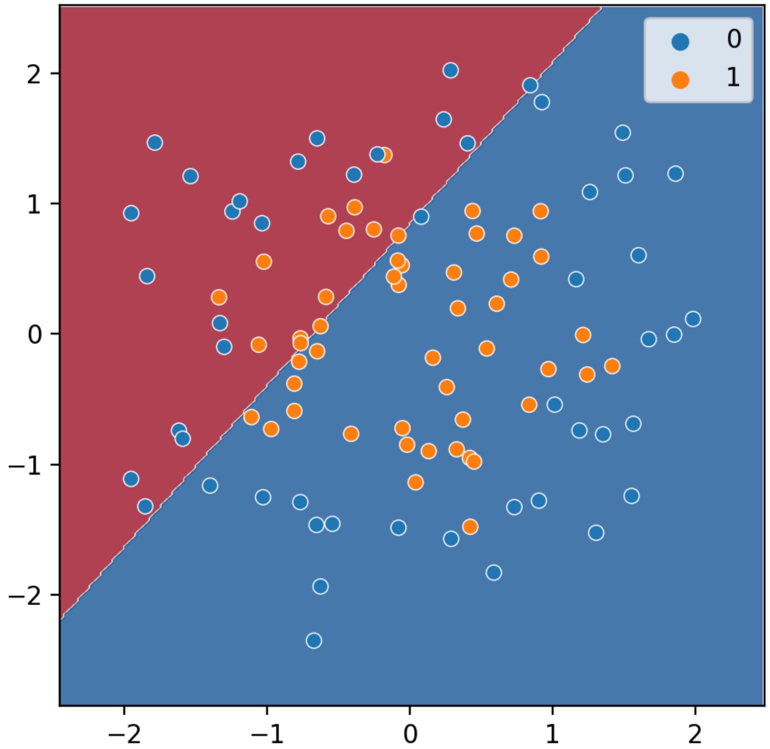


非线性支持向量机

❑ 试一试

- 使用非线性支持向量机对案例数据进行分类。

2. 线性SVM



显然不能对数据进行合理分类

非线性支持向量机

❑ 试一试

- 使用非线性支持向量机对案例数据进行分类。

3. 训练非线性SVM

```
from sklearn.preprocessing import StandardScaler
from sklearn.svm import SVC

# 数据标准化
X_ = StandardScaler().fit_transform(X)
data = X_, y

# 构建非线性支持向量机，训练模型
clf = SVC(C=1, kernel='rbf', random_state=3)
clf.fit(X_, y)
```

设置了核函数!

非线性支持向量机

❑ 试一试

- 使用非线性支持向量机对案例数据进行分类。

4. 查看分类结果

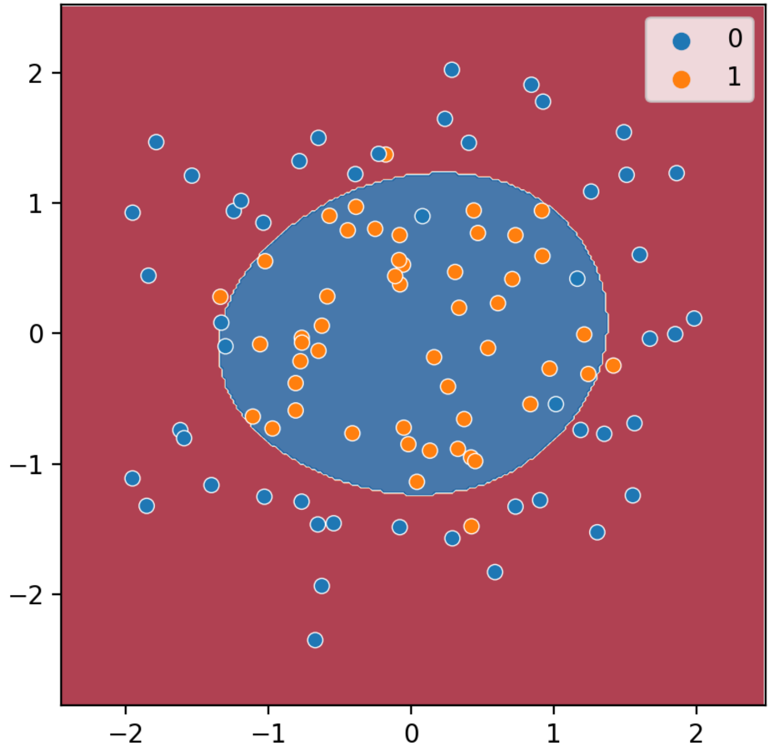
```
def plot_predicted_proba(data, clf, h=0.02):
    X, y = data

    x_min, x_max = X[:, 0].min() - .5, X[:, 0].max() + .5
    y_min, y_max = X[:, 1].min() - .5, X[:, 1].max() + .5
    xx, yy = np.meshgrid(np.arange(x_min, x_max, h),
                         np.arange(y_min, y_max, h))
    Z = clf.predict(np.c_[xx.ravel(), yy.ravel()])
    Z = Z.reshape(xx.shape)

    plt.figure(figsize=(5, 5))
    plt.contourf(xx, yy, Z, cmap=plt.cm.RdBu, alpha=.8)
    sns.scatterplot(x=X[:, 0], y=X[:, 1], hue=y, legend='full')
    plt.xlim(x_min, x_max)
    plt.ylim(y_min, y_max)
    plt.show()

plot_predicted_proba(data, clf)
```

非线性支持向量机

❑ 试一试

- 使用非线性支持向量机对案例数据进行分类。

4. 查看分类结果



Bingo~

支持向量机

❑ 学习目标

- 了解支持向量机的基本概念
- 掌握线性可分支持向量机模型
- 掌握软间隔支持向量机模型
- 理解非线性支持向量机中的核函数
- **介绍序列最小优化算法（阅读内容）**

支持向量机模型求解

❑ 序列最小优化（SMO）算法（阅读内容）

- **非线性支持向量机的对偶问题**

$$\max_{\alpha} \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j \kappa(\boldsymbol{x}_1, \boldsymbol{x}_2)$$

$$\text{s.t.} \quad \sum_{i=1}^{m} \alpha_i y_i = 0,$$

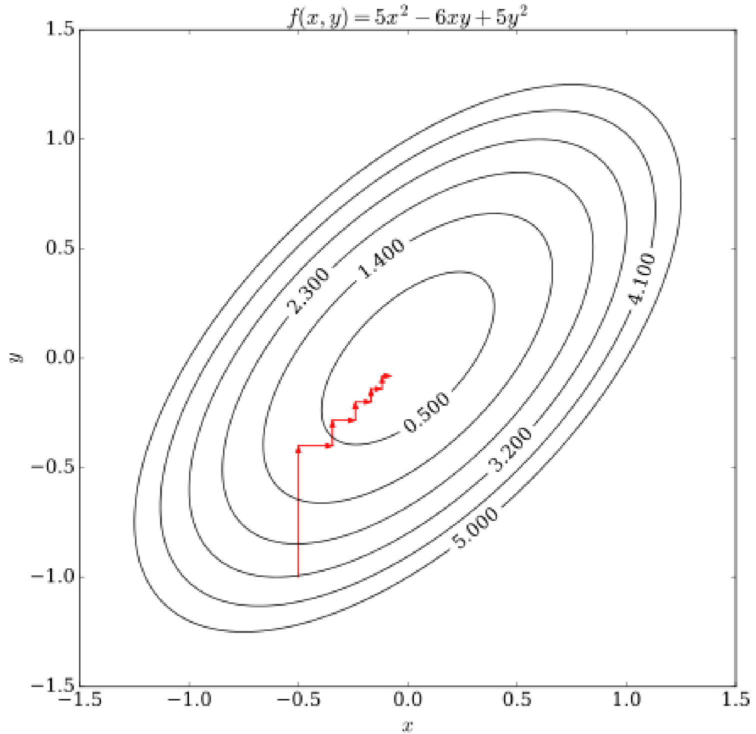
$$0 \le \alpha_i \le C, \quad i = 1, 2, \ldots, m.$$

  - 决策变量数 = 样本数
  - 采用通用二次规划程序求解计算效率低

支持向量机模型求解

❑ 序列最小优化（SMO）算法（阅读内容）

- **坐标上升（下降）法**
  - 每次只优化一个变量
  - 记优化目标函数为$\max_{\boldsymbol{\alpha}} g(\boldsymbol{\alpha})$
  - While 未收敛：
    - For $i = 1,2,\ldots,m$：
      - $\alpha_i \leftarrow \underset{\alpha}{\operatorname{argmax}}\, g(\alpha_1,\ldots,\alpha_{i-1},\alpha,\alpha_{i+1},\ldots,\alpha_m)$



支持向量机模型求解

❑ 序列最小优化（SMO）算法（阅读内容）

- **坐标上升（下降）法是否适用于支持向量机求解?**

$$\sum_{i=1}^{m} \alpha_i y_i = 0$$

例如，我们希望更新变量 α_1 ，但受上面的约束影响，

有 $\alpha_1 = -y_i \sum_{i=2}^{m} a_i y_i$ ， 因此 α_1 被固定住，无法更新。

如何解决?

支持向量机模型求解

❑ 序列最小优化（SMO）算法（阅读内容）

- **SMO**：每次优化两个变量!
- 每次迭代时，选择两个不同的待优化变量α_i和α_j，固定其他变量，求解使目标函数最优的α_i和α_j。
- 选择另外两个不同的待优化变量，重复以上步骤，直至目标函数最优值收敛。

关键问题

- ❑ 如何同时优化两个变量?
- ❑ 每一轮迭代选择哪两个变量进行优化?

支持向量机模型求解

❑ 序列最小优化（SMO）算法（阅读内容）

- **双变量二次规划**
- 考虑选定的两个待优化变量为α_i和α_j，本轮迭代的优化问题为：

$$\max_{\alpha_i,\alpha_j} \quad \alpha_i+\alpha_j-\frac{1}{2}\alpha_i^2\kappa(\boldsymbol{x}_i,\boldsymbol{x}_i)-\frac{1}{2}\alpha_j^2\kappa(\boldsymbol{x}_j,\boldsymbol{x}_j)-\alpha_i\alpha_j y_i y_j\kappa(\boldsymbol{x}_i,\boldsymbol{x}_j)$$

$$-\sum_{k\neq i,j}\alpha_i\alpha_k y_i y_k\kappa(\boldsymbol{x}_i,\boldsymbol{x}_k)-\sum_{k\neq i,j}\alpha_j\alpha_k y_j y_k\kappa(\boldsymbol{x}_j,\boldsymbol{x}_k)$$

$$\text{s.t.}\quad \alpha_i y_i+\alpha_j y_j=-\sum_{k\neq i,j}\alpha_k y_k=\varsigma,$$

$$0\le\alpha_i,\alpha_j\le C.$$

支持向量机模型求解

❑ 序列最小优化（SMO）算法（阅读内容）

- **双变量二次规划**
- 仅考虑第一个约束条件：

$$\max_{\alpha_i,\alpha_j} \ \alpha_i+\alpha_j-\frac{1}{2}\alpha_i^2\kappa(\boldsymbol{x}_i,\boldsymbol{x}_i)-\frac{1}{2}\alpha_j^2\kappa(\boldsymbol{x}_j,\boldsymbol{x}_j)-\alpha_i\alpha_j y_i y_j\kappa(\boldsymbol{x}_i,\boldsymbol{x}_j)$$
$$-\sum_{k\neq i,j}\alpha_i\alpha_k y_i y_k\kappa(\boldsymbol{x}_i,\boldsymbol{x}_k)-\sum_{k\neq i,j}\alpha_j\alpha_k y_j y_k\kappa(\boldsymbol{x}_j,\boldsymbol{x}_k)$$

- 对于输入特征向量$\boldsymbol{x}$，模型预测值记为$u(\boldsymbol{x})$

$$u(\boldsymbol{x})=\sum_{i=1}^{m}\alpha_i y_i\kappa(\boldsymbol{x}_i,\boldsymbol{x})+b$$

支持向量机模型求解

❑ 序列最小优化（SMO）算法（阅读内容）

- **双变量二次规划**

- 模型预测值$u(\boldsymbol{x}_\mathrm{i})$与真实值$y_i$之差记为$E_i$

$$E_i = u(\boldsymbol{x}_i) - y_i = \sum_{j=1}^{m} \alpha_j y_j \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) + b - y_i$$

- 不考虑第二个约束时α_j的最优解为

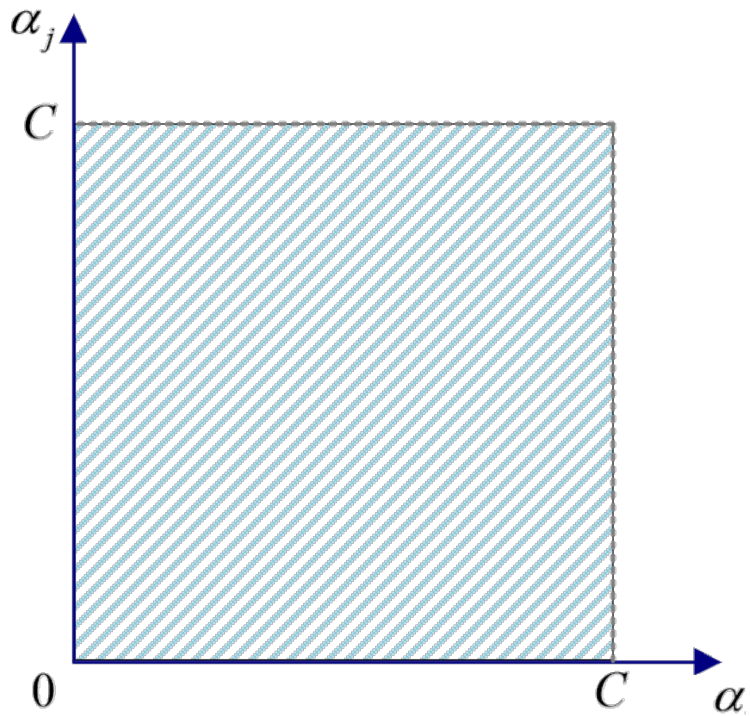
$$\alpha_j^u = \alpha_j + \frac{y_j(E_i - E_j)}{\eta}$$

$$\eta = \kappa(\boldsymbol{x}_i, \boldsymbol{x}_i) + \kappa(\boldsymbol{x}_j, \boldsymbol{x}_j) - 2\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)$$

支持向量机模型求解

❑ 序列最小优化（SMO）算法（阅读内容）

- **双变量二次规划**
- 对第二个约束可视化展示 $0 \leq \alpha_i, \alpha_j \leq C$

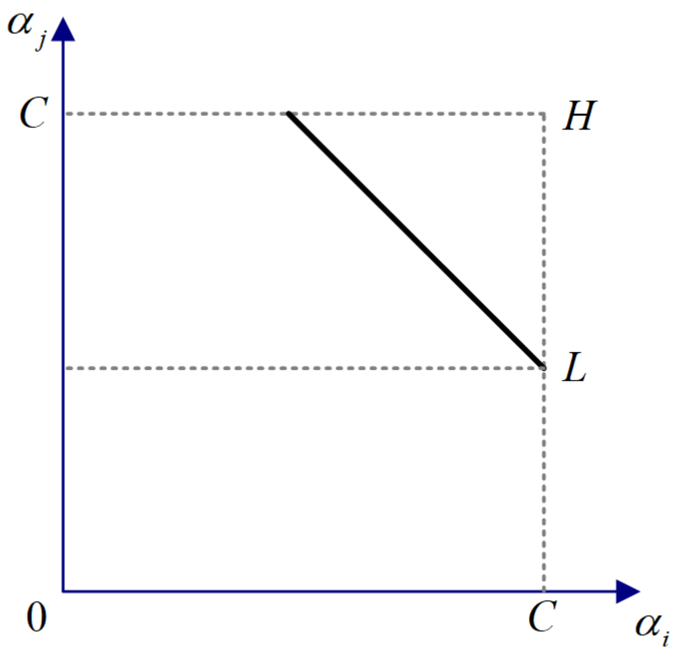


- 蓝色区域为可行域

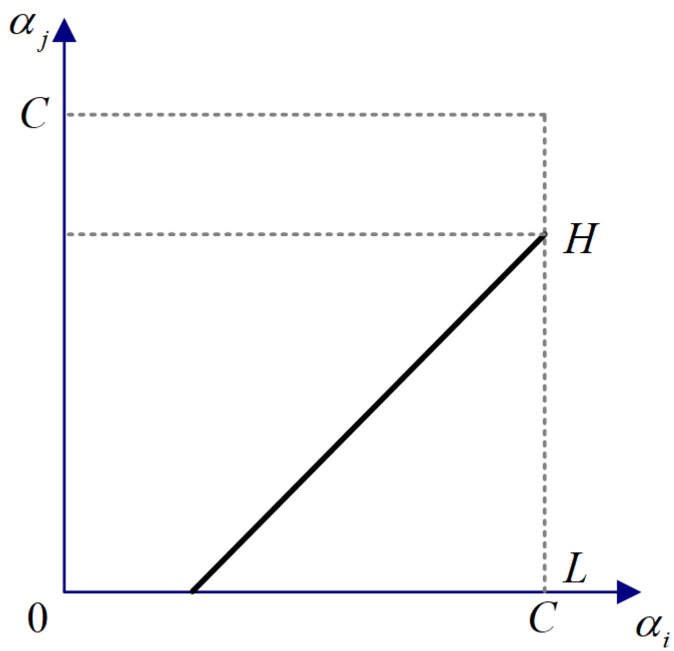
支持向量机模型求解

❑ 序列最小优化（SMO）算法（阅读内容）

- **双变量二次规划**
- 考虑第一个约束 $\alpha_i y_i + \alpha_j y_j = \varsigma$
- 可行域减小到一条线段



$y_i = y_j$



$y_i \neq y_j$

支持向量机模型求解

□ 序列最小优化（SMO）算法（阅读内容）

- **双变量二次规划**
  - 考虑第二个约束时 α_j的最优解为

$$\alpha_j^* = \begin{cases} L, & \alpha_j^u < L; \\ \alpha_j^u, & L \le \alpha_j^u \le H; \\ H, & \alpha_j^u > H. \end{cases}$$

  - 根据第一个约束，得到α_i的最优解为

$$\alpha_i^* = \frac{1}{y_i}\left(\alpha_i y_i + y_j(\alpha_j - \alpha_j^*)\right) = \alpha_i + y_i y_j(\alpha_j - \alpha_j^*)$$

支持向量机模型求解

❑ 序列最小优化（SMO）算法（阅读内容）

- **双变量二次规划**

- 模型预测还需要计算偏置项b

$$\boldsymbol{u}(\boldsymbol{x})=\sum_{i=1}^{m}\alpha_i y_i \kappa(\boldsymbol{x}_i,\boldsymbol{x})+b$$

- 考虑支持向量 $(\boldsymbol{x}_s, y_s)$，满足

$$y_s\left(\sum_{i=1}^{m}\alpha_i y_i \kappa(\boldsymbol{x}_i,\boldsymbol{x}_s)+b\right)=1$$

- 对所有支持向量计算偏置项后取其平均值，提高数值稳定性

$$S=\{i\mid 0<\alpha_i<C, i=1,2,\ldots,m\}$$

$$b=\frac{1}{|S|}\sum_{s\in S}\left(y_s-\sum_{i=1}^{m}\alpha_i y_i \kappa(\boldsymbol{x}_i,\boldsymbol{x}_s)\right)$$

支持向量机模型求解

❑ 序列最小优化（SMO）算法（阅读内容）

- **选择待优化变量的启发式思路**

- 若所有变量满足KKT条件，则也满足约束 $0 \le \alpha_i, \alpha_j \le C$

$$\begin{cases} \alpha_i = 0 \Leftrightarrow y_i u(\boldsymbol{x}_i) \ge 1 \\ 0 < \alpha_i < C \Leftrightarrow y_i u(\boldsymbol{x}_i) = 1 \\ \alpha_i = C \Leftrightarrow y_i u(\boldsymbol{x}_i) \le 1 \end{cases}$$

- 有研究指出，$0 < \alpha_i < C$对应的样本更容易违反KKT条件
- 第一个待优化变量：遍历整个训练集和满足$0 < \alpha_i < C$的样本，挑选其中不满足KKT条件的变量

支持向量机模型求解

序列最小优化（SMO）算法（阅读内容）

- **选择待优化变量的启发式思路**

- 回顾变量更新的公式：
- 变量更新幅度与$|E_i - E_j|$有关

$$\alpha_j^u = \alpha_j + \frac{y_j(E_i - E_j)}{\eta}$$

- 第二个待优化变量：选择$|E_i - E_j|$最大的变量进行更新

支持向量机

❑ 实践环节

- **使用Scikit-learn中的SVM模型进行网格拥堵分类**

- (1) 读取数据
  - 导入基础模块

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

支持向量机

❑ 实践环节

- **使用Scikit-learn中的SVM模型进行网格拥堵分类**

- (1) 读取数据
  - 读取、查看数据

```
df = pd.read_csv("DATASET-B.csv")
print(len(df))
```

```
print(df.sample(10, random_state=233))
```

支持向量机

❑ 实践环节

- **使用Scikit-learn中的SVM模型进行网格拥堵分类**

- (2) 探索性数据分析
  - 缺失值检查

```
print(sum(df.isnull().any()))
```

  - 描述性统计

```
features = ['aveSpeed', 'gridAcc', 'volume', 'speed_std', 'stopNum']
print(df[features].describe())
```

支持向量机

❑ 实践环节

- **使用Scikit-learn中的SVM模型进行网格拥堵分类**

- (2) 探索性数据分析
  - 查看数据相关性

```
plt.figure(dpi=200)
corr = df[features + ["labels"]].corr()
sns.heatmap(corr, square=True)
plt.show()
```

支持向量机

❑ 实践环节

- **使用Scikit-learn中的SVM模型进行网格拥堵分类**

- (2) 探索性数据分析
  - 查看数据分布

```
fig, ax = plt.subplots(3, 2, figsize=(10, 3.5), dpi=200)
for axi, feature in zip(ax.ravel(), features):
    sns.boxplot(x=feature, y='labels', orient='h', data=df, fliersize=1, ax=axi)
plt.tight_layout()
plt.show()
```

支持向量机

❑ 实践环节

- **使用Scikit-learn中的SVM模型进行网格拥堵分类**

- (3) 模型训练
  - 导入机器学习模块

```
from sklearn.svm import SVC
from sklearn.model_selection import GridSearchCV, train_test_split
from sklearn.metrics import accuracy_score
from sklearn.preprocessing import StandardScaler
```

支持向量机

❑ 实践环节

- **使用Scikit-learn中的SVM模型进行网格拥堵分类**

- (3) 模型训练
  - 数据采样（减少运算量，可尝试选取不同数据或全样数据）
  - 划分训练集-测试集

```
df_sample = df.sample(20000, random_state=233)
X = df_sample[features]
y = df_sample["labels"]
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.3, random_state=233)
```

支持向量机

❑ 实践环节

- **使用Scikit-learn中的SVM模型进行网格拥堵分类**

- (3) 模型训练
  - 数据标准化

```
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

支持向量机

❑ 实践环节

- **使用Scikit-learn中的SVM模型进行网格拥堵分类**

- (3) 模型训练
  - **网格搜索**超参数

课后练习

❑ 使用**贝叶斯优化**调参

```
params = {
    "kernel": ["rbf"],
    "gamma": [0.1, 0.2, 0.5, 1],
    "C": [10, 100, 1e3]
}
clf = GridSearchCV(SVC(), params, cv=5)
clf.fit(X_train, y_train)
print(clf.best_params_)
```

支持向量机

❑ 实践环节

- **使用Scikit-learn中的SVM模型进行网格拥堵分类**

- (3) 模型训练
  - 模型预测、输出预测精度

试一试

❑ 尝试其他**核函数、超参数**，对比预测精度

```python
y_pred = clf.predict(X_test)
acc = accuracy_score(y_test, y_pred)
print(acc)
```

支持向量机

❑ 学习总结

- 支持向量机的基本概念
  - **线性可分 | 间隔 | 支持向量 | 分隔超平面**
- 线性可分支持向量机模型
  - **支持向量机基本型 | 支持向量机基本型的对偶问题**
- 软间隔支持向量机模型
  - **软间隔 | hinge损失 | 软间隔支持向量机 | 软间隔支持向量机的对偶问题**
- 非线性支持向量机中的核函数
  - **核技巧 | 非线性支持向量机 | 常见核函数 | 核函数构造方法**
- 序列最小优化算法
  - **坐标上升法 | 双变量优化**

支持向量机

❑ 课后思考

- 支持向量机如何用来解决**回归问题**?

❑ 阅读材料

- Scholkopf, B., Smola, A.J., 2001. **Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond**. MIT Press, Cambridge. 【深入讨论SVM】
- Bishop, C.M., 2006. **Pattern Recognition and Machine Learning**. Springer, New York. 【第6、7章，核函数在机器学习的应用】