SVM的代码实现

scikit-learn

小胖

目录

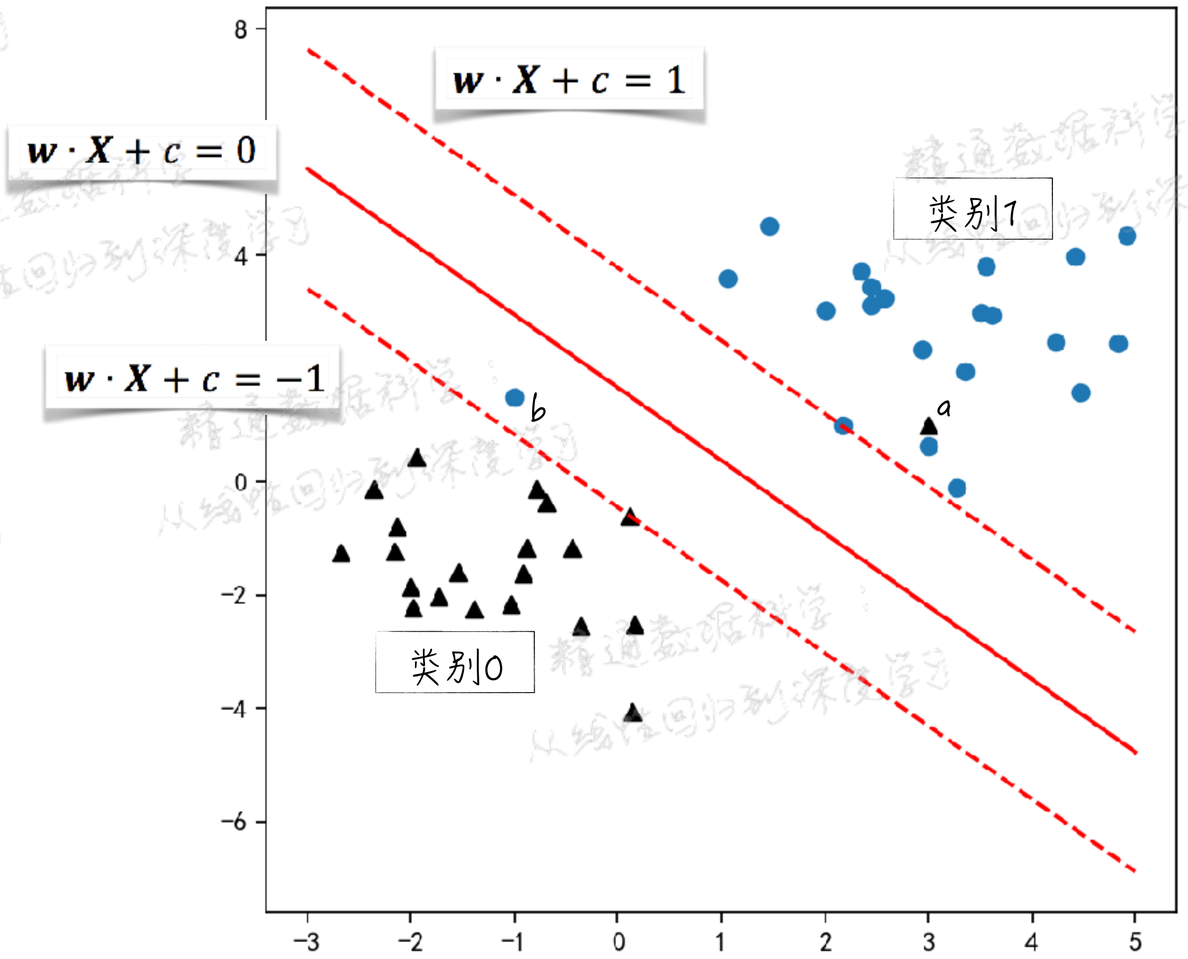
Hard margin v.s. Soft margin

损失系数

$$\min \tfrac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_i \xi_i$$
$$y_i(\boldsymbol{w}\cdot\boldsymbol{X_i} + c) \geqslant 1 - \xi_i;\ \xi_i \geqslant 0$$

损失系数

数据线性不可分时，加入误分类的损失



Hard margin v.s. Soft margin

Decision function

Original SVM

参数估计公式

$$\min_{w,c} \frac{1}{2}\|w\|^2 + C\sum_i \xi_i$$
$$s.t.\ \ y_i(w \cdot X_i + c) \geq 1 - \xi_i$$
$$\xi_i \geq 0, \forall i$$

预测公式

$$\hat{y}_j = \text{sign}(\hat{w} \cdot X_j + \hat{c})$$

Decision function

Dual problem

参数估计公式

$$\max_{\alpha} \sum_i \alpha_i - \frac{1}{2}\sum_{i,j} y_i y_j \alpha_i \alpha_j (X_i \cdot X_j)$$
$$s.t.\ \ 0 \leq \alpha_i \leq C$$
$$\sum_i \alpha_i y_i = 0, \forall i$$

预测公式

$$\hat{y}_j = \text{sign}\left(\sum_i \hat{\alpha}_i y_i (X_i \cdot X_j) + \hat{c}\right)$$

目录

核函数

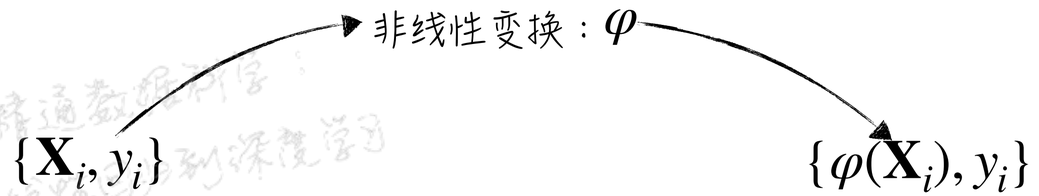
核函数回顾

Dual problem

$$\max_{\alpha} \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} y_i y_j \alpha_i \alpha_j (\boldsymbol{X_i} \cdot \boldsymbol{X_j})$$
$$s.t.\ \ 0 \leq \alpha_i \leq C$$
$$\sum_i \alpha_i y_i = 0, \forall i$$

$$\hat{y}_j = \text{sign}(\sum_i \hat{\alpha}_i y_i (\boldsymbol{X_i} \cdot \boldsymbol{X_j}) + \hat{c})$$

- SVM模型在训练和预测时，只会用到内积运算
- 我们只关心非线性变换后，“新数据”的内积



SVM只需要： $\varphi(\mathbf{X}_i) \cdot \varphi(\mathbf{X}_j)$

$$K(\mathbf{X}_i, \mathbf{X}_j) = \varphi(\mathbf{X}_i) \cdot \varphi(\mathbf{X}_j)$$

这就是核函数

核函数

常用核函数

常用核函数

Linear kernel：$K(X_i, X_j) = X_i \cdot X_j$ → 没做任何空间变换，对应着最经典的线性支持向量学习机

Polynomial kernel：$K(X_i, X_j) = (gamma(X_i \cdot X_j) + coef0)^{degree}$

Sigmoid kernel：$K(X_i, X_j) = \tanh(gamma(X_i \cdot X_j) + coef0)$

Laplacian kernel：$K(X_i, X_j) = \exp(-gamma\|X_i - X_j\|_1)$

RBF kernel：$K(X_i, X_j) = \exp(-gamma\|X_i - X_j\|^2)$

Sigmoid kernel、Laplacian kernel、RBF kernel → 将数据映射到无限维空间

RBF kernel → 也称为Gaussian kernel，是魔力十足的核函数

THANK YOU