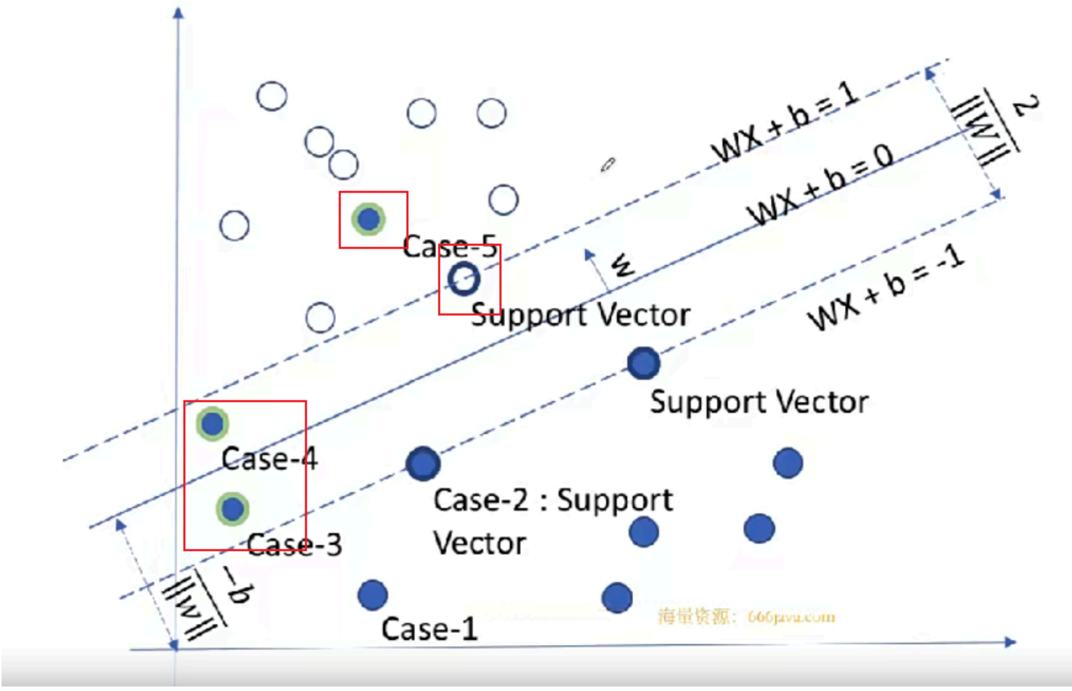
线性可分支持向量机

- 数学模型
  - $$\begin{cases} w^Tx_i + b >= 0, y_i = +1 \\ w^Tx_i + b <= 0, y_i = -1 \end{cases}$$
- 离超平面最近的样本称为**支持向量**
  - 几何间隔
    - $$d_{jihe} = \frac{y_i(w * x_i + b_i)}{||w||}$$
  - 函数间隔
    - $$d_{hanshu} = y_i(w * x_i + b_i)$$
- 目标函数
  - 文字描述
    - **最大**化数据集中的**最小**几何间隔
  - 数学模型
    - $$\frac{\max \qquad d^*_{jihe} = \frac{y^*_i(w * x^*_i + b^*_i)}{||w||}}{st: \qquad d_{jihe} = \frac{y_i(w * x_i + b_i)}{||w||} > d^*_{jihe} \qquad i = 1,2,3\ldots\ldots}$$
    - 对于最小**最小**几何间隔的那个点（超平面/支持向量的唯一性），可以变化w,b使得下式等于1，可以简化目标

$$w * x^*_i + b^*_i$$

- 支持向量机基本型
- $$\frac{\max \qquad d^*_{jihe} = \frac{1}{||w||} \qquad ==> \qquad min \qquad \frac{1}{2}||w||^2}{st: \qquad d_{jihe} = y_i(w * x_i + b_i) >= 1 \qquad i = 1,2,3\ldots\ldots}$$
- 求解二次规划问题
  - 由拉格朗日乘子法可得其对偶问题
    - $$\frac{\max_a \quad \sum_{i=1}^m a_i - \frac{1}{2}\sum_{i=1}^m \sum_{j=1}^m a_i a_j y_i y_j x_i^T x_j \qquad a\text{为每条约束的乘子}}{st: \sum_{i=1}^m a_i y_i = 0 \qquad ; a > 0}$$
  - SMO求解对偶问题
    - 主要思想是将a中的所有变量中选择两个变量作为分析，其余变量忽略为常熟，求取a的近似解
      - 不断更新选择的两个变量
  - 求出a后可得模型
    - $$f（x） = w^Tx + b = \sum_{i=1}^m a_i y_i x_i^T x + b$$

带松弛变量的SVM

- 处理有异常值的线性可分svm
- 异常值
    - 

    - 软间隔主要改变
        - $$y_i(w*x_i+b_i)>=1-t_i \qquad i=1,2,3......$$
    - 目标函数
        - $$\frac{min \qquad \frac{1}{2}||w||^2+C\sum_{i=1}^{N}t_i}{st: y_i(w*x_i+b_i)>=1-t_i, i=1,2,3......,t_i>=0}$$
    - 对偶问题
        - $$\frac{\max_a \quad \sum_{i=1}^{m}a_i-\frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}a_ia_jy_iy_jx_i^Tx_j \qquad a\text{为每条约束的乘子}}{st: \sum_{i=1}^{m}a_iy_i=0 \qquad ;a>0;u_i>0;C-a_i-u_i=0}$$
        - $$
y_iy_j表示符号; \quad
x_i^Tx_j表示两个样本的相似度; \quad

          $$-\text{合页损失函数}-\$\$min_{w,b}\sum_{i=1}^{N}[1-y_i(w*x_i+b)]_++\lambda_i||w||^2$$
- 可由目标函数推导，并且可以使用梯度方法优化取得w,b

核函数

- 对于在样本维度特征空间中无法线性可分的情况下，可以使用一种映射将原本样本映射到更加高位的空间，寻找一个可分的超平面
- 则对偶问题变为

- $$\frac{\max_a \quad \sum_{i=1}^m a_i - \frac{1}{2}\sum_{i=1}^m \sum_{j=1}^m a_i a_j y_i y_j g(x)_i^T g(x)_j \qquad g\text{为映射}}{st:\sum_{i=1}^m a_i y_i = 0 \qquad ;a>0}$$
- $$f（x）=w^T x+b=\sum_{i=1}^m a_i y_i g(x)_i^T g(x)+b$$
- 但是因为经过映射后维度可能会很高，使用g(x)计算会使计算大大增加
- $$k(x_i,x_j)=g(x)_i^T g(x)$$