# 支持向量机SVM

黑马程序员
www.itheima.com

传智教育旗下
高端IT教育品牌

# 目录

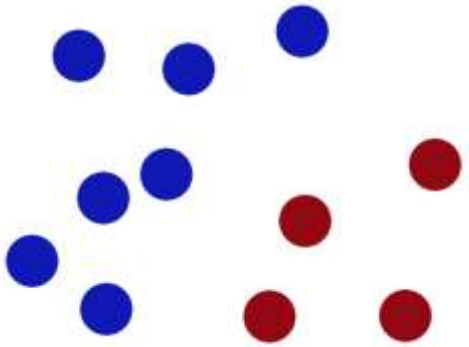
Contents

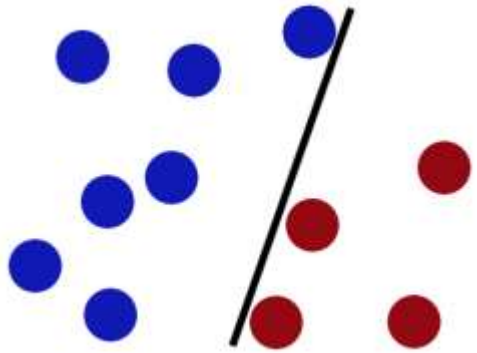
# 学习目标

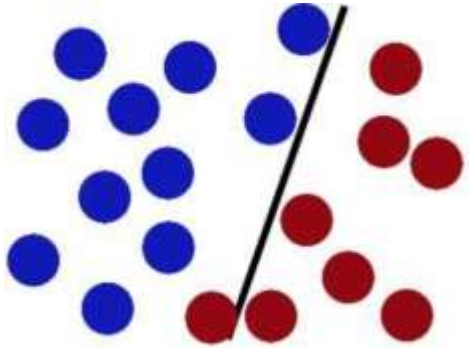
Learning Objectives

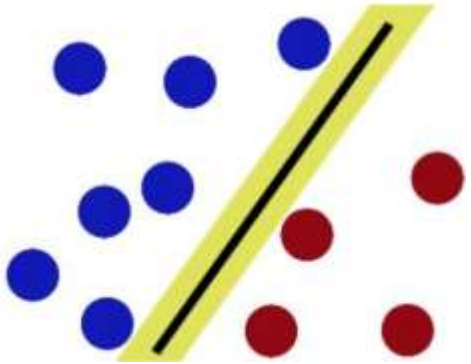
1. 理解SVM算法的思想
2. 知道SVM中的硬间隔
3. 理解SVM中的软间隔和惩罚系数
4. 知道SVM中核函数的作用

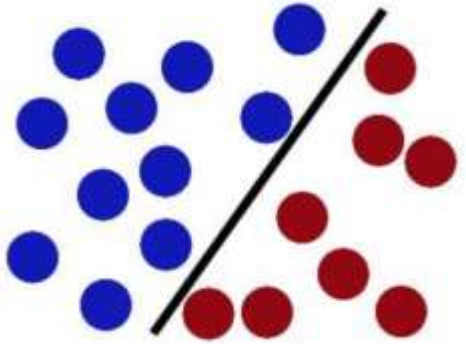
# 支持向量机小故事



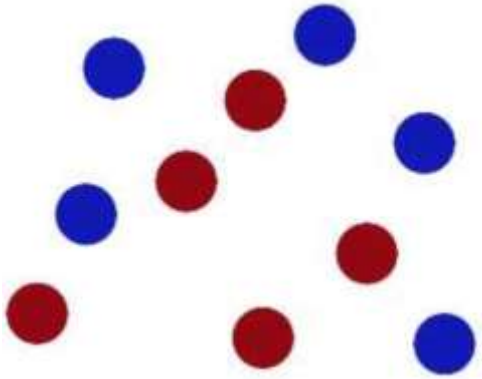


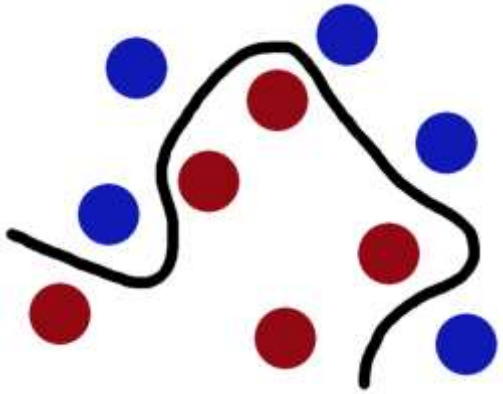


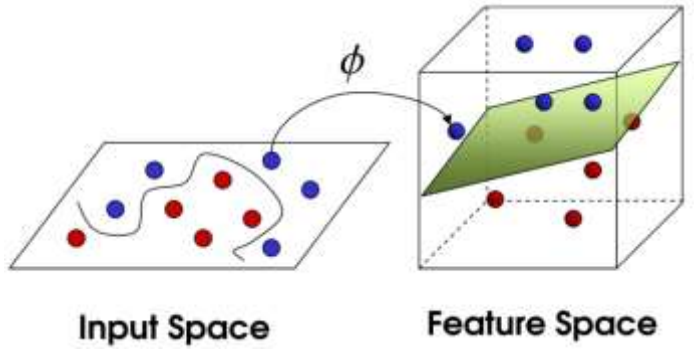




# 支持向量机小故事





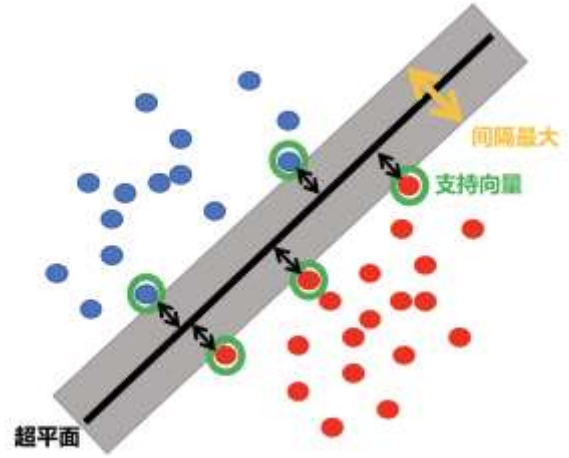


# 支持向量机SVM

SVM全称是Supported Vector Machine（支持向量机）

即寻找到一个超平面使样本分成两类，并且间隔最大。

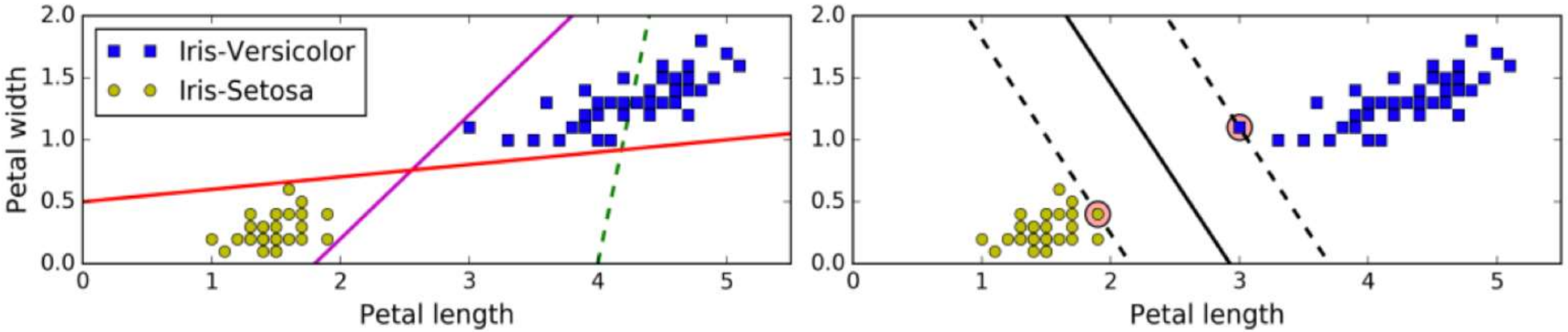
- 是一种监督学习算法，主要用于分类，也可用于回归
- 与逻辑回归和决策树等其他分类器相比，SVM 提供了非常高的准确度




**优缺点**

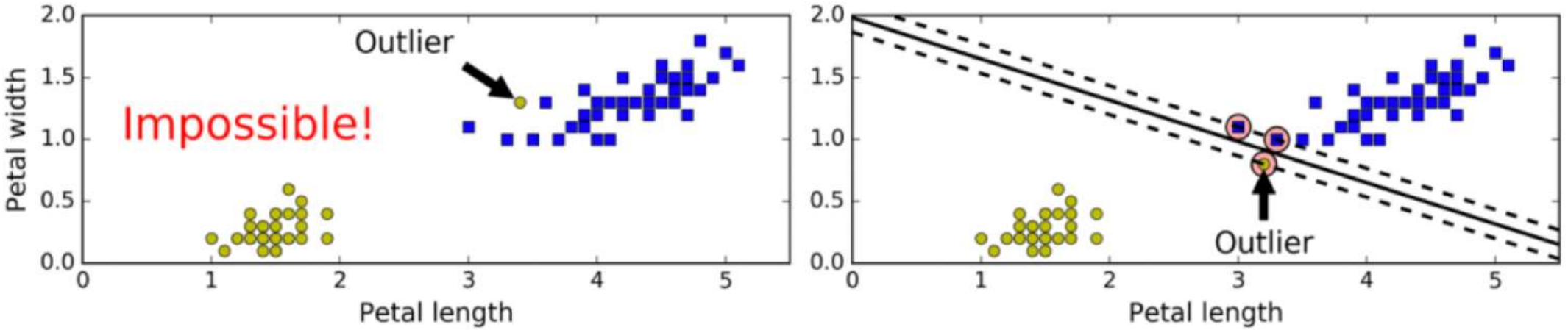
- 优点：
  （1）适合小样本、高纬度数据，比较强泛化能力
  （2）可有效地处理高维数据；可使用不同的核函数来适应不同的数据类型
- 缺点：
  计算复杂度较高，对于大规模数据的处理可能会存在困难

# 超平面最大间隔


Iris-Versicolor
Iris-Setosa
Petal width
Petal length
2.0
1.5
1.0
0.5
0.0
0
1
2
3
4
5

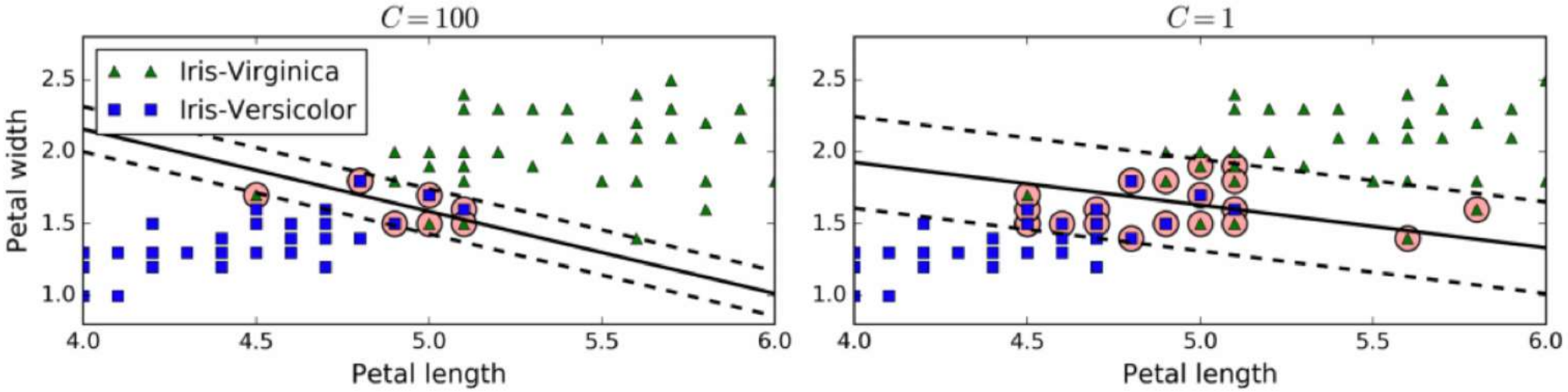
## 硬间隔Hard Margin

- 如果样本线性可分，在所有样本分类都正确的情况下，寻找最大间隔，这就是硬间隔
- 如果出现异常值、或者样本不能线性可分，此时硬间隔无法实现。



# 软间隔Soft Margin和惩罚系数

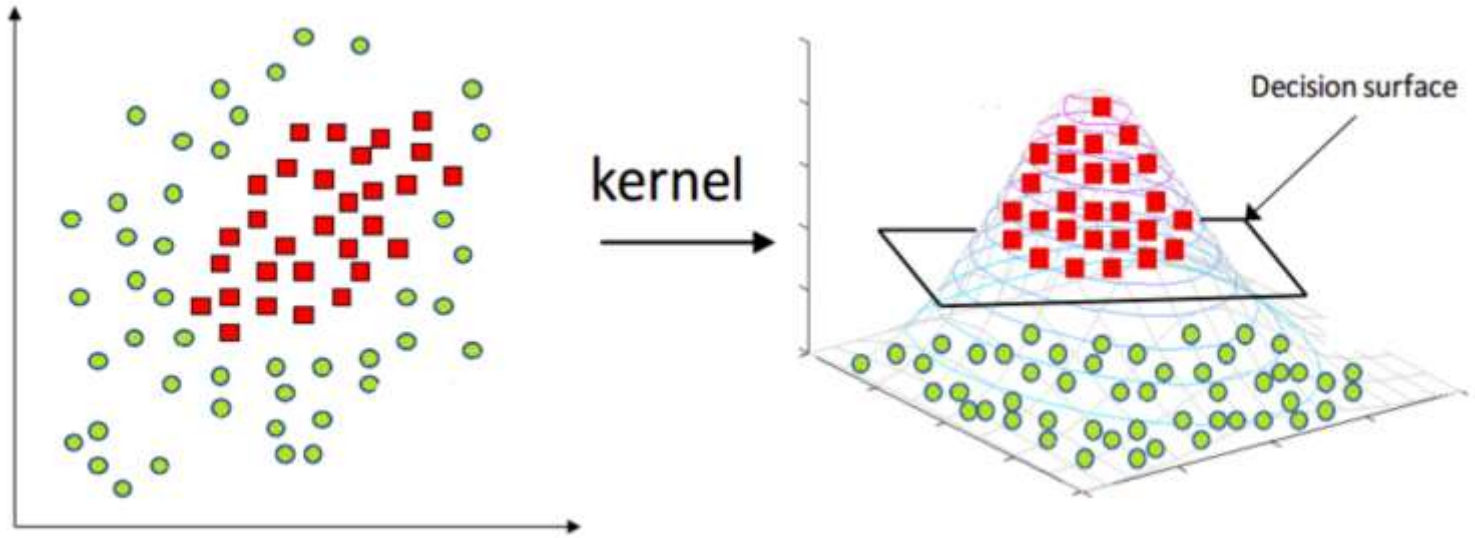
- 允许部分样本，在最大间隔之内，甚至在错误的一边，寻找最大间隔，这就是软间隔
- 目标是尽可能在保持间隔宽阔和限制间隔违例之间找到良好的平衡。




通过惩罚系数C来控制这个平衡：C值越小，则间隔越宽，但是间隔违例也会越多。

# 核函数Kernel

核函数将原始输入空间映射到新的特征空间，使得原本线性不可分的样本在核空间可分



# 总结

## 1 支持向量机

寻找到一个超平面使样本分成两类，并且间隔最大

## 2 硬间隔

若样本线性可分，在所有样本分类都正确的情况下，寻找最大间隔

## 3 软间隔、惩罚参数C

软间隔：容忍一部分样本在最大间隔之内，甚至在错误的一边

惩罚参数C影响：C惩罚力度越大，则间隔越小；C惩罚力度越小，则间隔越大

## 4 核函数

当数据线性不可分时，将原始数据映射到高维空间，使得样本能够线性可分

练习

1、下列关于SVM的描述错误的是？（单选题）

A）它的全称为支撑向量机（Supported Vector Machine）

B）它的主要任务是找到一个超平面将不同的样本划分开来

C）硬间隔和软间隔都是SVM分割超平面中的一种

D）SVM模型可以通过调小C参数的取值，来减少间隔违例

答案解析： C值越小，间隔违例越多

答案：D

# 目录

Contents

# 学习目标

Learning Objectives

1. 能使用LinearSVC API函数进行分类
2. 能使用参数C进行问题调参

## 支持向量机LinearSVC API

sklearn.svm.LinearSVC(C=1.0)

- C:惩罚系数，类似于线性回归中的正则化系数。
- LinearSVC API的使用方式：

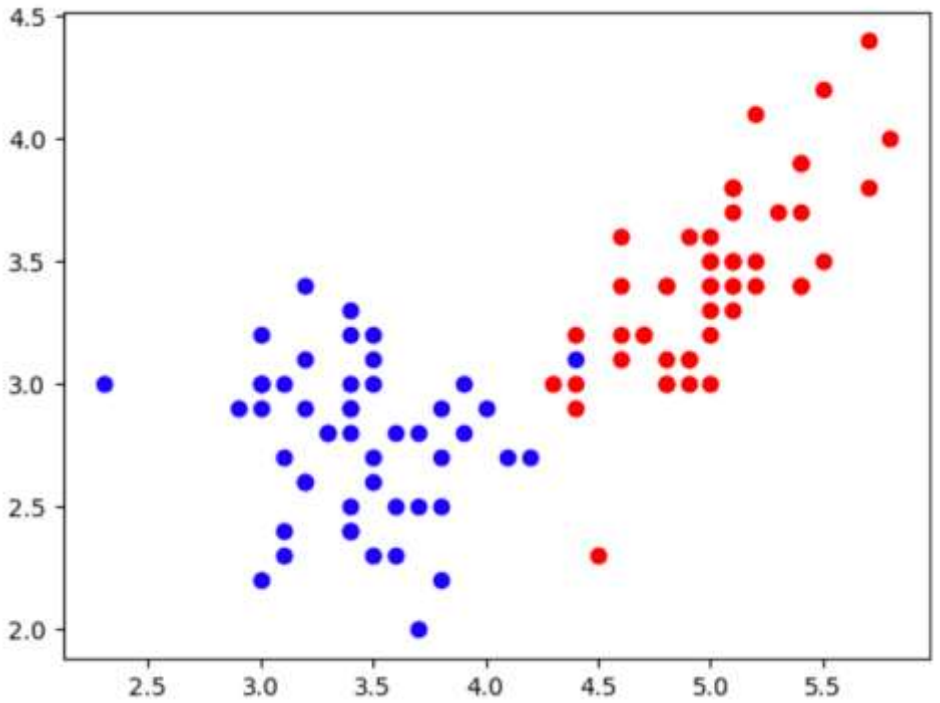
```
from sklearn.svm import LinearSVC
mysvc = LinearSVC(C=30)
mysvc.fit(X_standard, y)
print(mysvc.score(X_standard, y))
```

## 案例 使用LinearSVC探索鸢尾花分类 – API初步使用

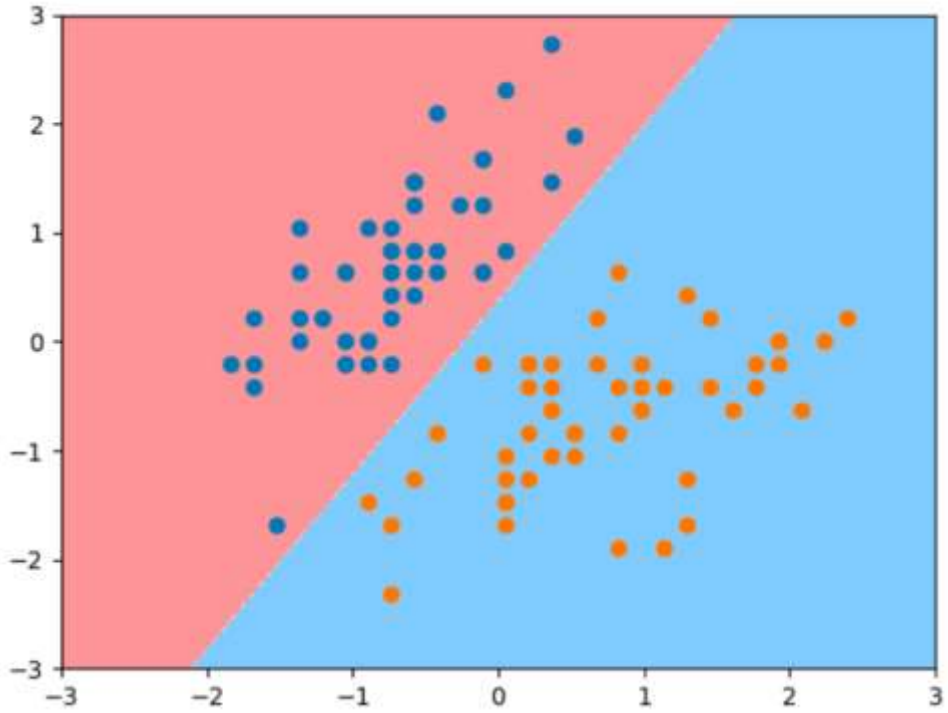
```
# 1.导入依赖包
import numpy as np
import matplotlib.pyplot as plt
from sklearn import datasets
from sklearn.preprocessing import StandardScaler
from sklearn.svm import LinearSVC

def dm01_鸢尾花分类svm():
    # 2.加载数据及数据预处理
    # 2.1 加载数据
    X, y = datasets.load_iris(return_X_y=True)
    print('x.shape-->', X.shape)
    print('y.shape-->', y.shape)
    # 2.2 数据处理，降维
    X = X[y<2, :2]
    y = y[y<2]
    print('x.shape-->', X.shape)
    print('y.shape-->', y.shape)
    # 数据可视化
    plt.scatter(X[y==0, 0], X[y==0, 1], color='red')
    plt.scatter(X[y==1, 0], X[y==1, 1], color='blue')
    plt.show()
```

```
    # 2.3 数据标准化
    transformer = StandardScaler()
    X_standard = transformer.fit_transform(X=X)

    # 3.模型训练及评估
    mysvc = LinearSVC(C=30)
    mysvc.fit(X_standard, y)
    print(mysvc.score(X_standard, y))

    # 4.绘制模型边界
    plot_decision_boundary(mysvc, axis=[-3, 3, -3, 3])
    plt.scatter(X_standard[y == 0, 0], X_standard[y == 0, 1], c='red')
    plt.scatter(X_standard[y == 1, 0], X_standard[y == 1, 1], c='blue')
    # plt.scatter(X_standard[:, 0], X_standard[:, 1], c=y)
    plt.show()
```

案例 **使用LinearSVC探索鸢尾花分类 - 效果展示**




鸢尾花数据展示



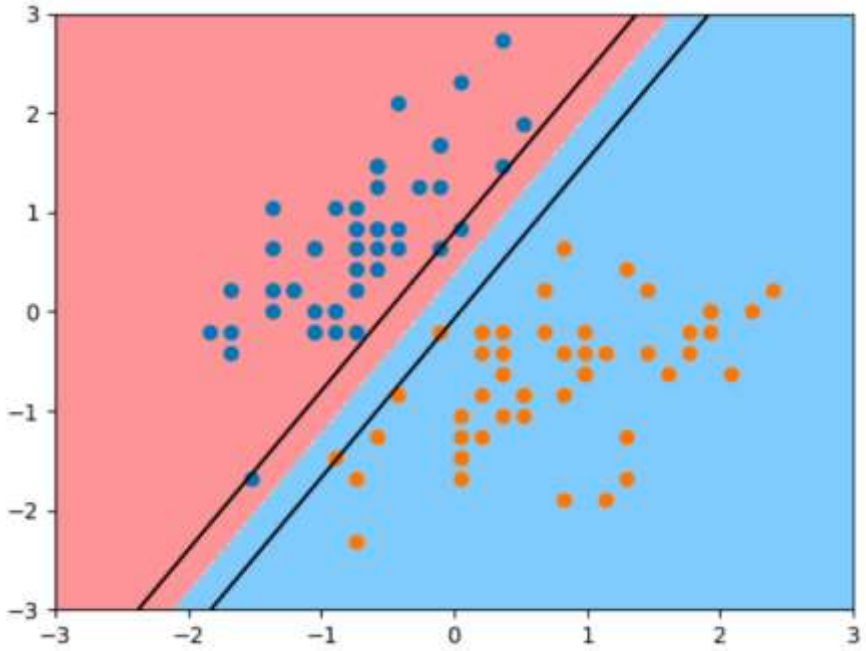

LiearSVC分类边界-超平面

## 案例 使用LinearSVC探索鸢尾花分类 – 惩罚参数C对超平面的影响

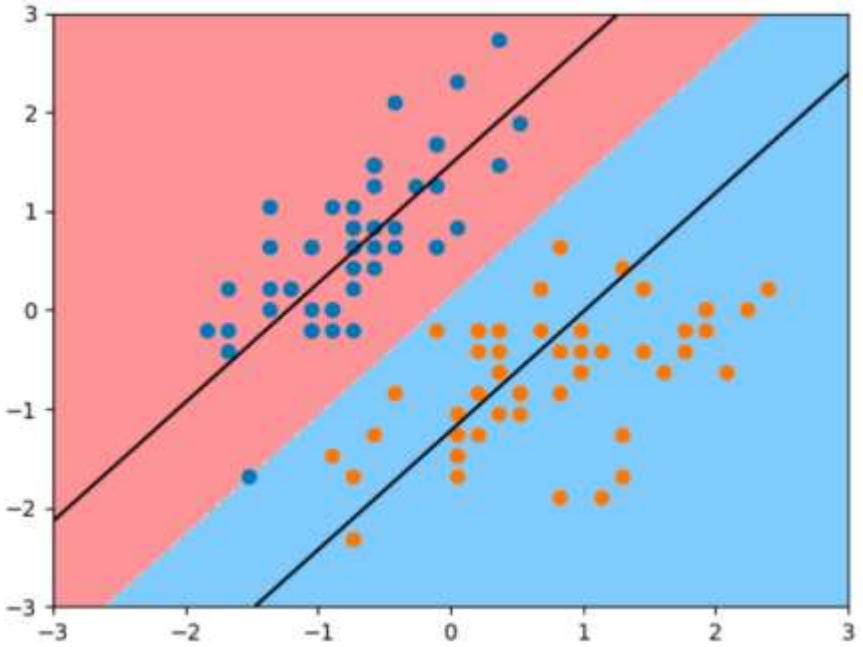
```
def dm02_鸢尾花分类svm():
    # 2.读取数据及数据预处理
    # 2.1 加载数据
    X, y = datasets.load_iris(return_X_y=True)
    print('x.shape-->', X.shape)
    print('y.shape-->', y.shape)

    # 2.2 数据预处理，降维
    X = X[y < 2, :2]
    y = y[y < 2]
    print('x.shape-->', X.shape)
    print('y.shape-->', y.shape)

    # 数据可视化
    plt.scatter(X[y == 0, 0], X[y == 0, 1], color='red')
    plt.scatter(X[y == 1, 0], X[y == 1, 1], color='blue')
    plt.show()

    # 2.3 数据标准化
    transformer = StandardScaler()
    X_standard = transformer.fit_transform(X=X)
```

```
    # 3.1模型训练 c=30
    mysvc = LinearSVC(C=30 , dual='auto')
    mysvc.fit(X_standard, y)
    print(mysvc.score(X_standard, y))

    # 3.2 绘制模型边界
    plot_decision_boundary_svc(mysvc, axis=[-3, 3, -3, 3])
    plt.scatter(X_standard[y == 0, 0], X_standard[y == 0, 1], c='red')
    plt.scatter(X_standard[y == 1, 0], X_standard[y == 1, 1], c='blue')
    # plt.scatter(X_standard[:, 0], X_standard[:, 1], c=y)
    plt.show()

    # 4.1 模型训练2
    mysvc2 = LinearSVC(C=0.1, dual='auto')
    mysvc2.fit(X_standard, y)
    print(mysvc2.score(X_standard, y))

    # 4.2 绘制模型边界2
    plot_decision_boundary_svc(mysvc2, axis=[-3, 3, -3, 3])
    plt.scatter(X_standard[y == 0, 0], X_standard[y == 0, 1], c='red')
    plt.scatter(X_standard[y == 1, 0], X_standard[y == 1, 1], c='blue')
    # plt.scatter(X_standard[:, 0], X_standard[:, 1], c=y)
    plt.show()
```

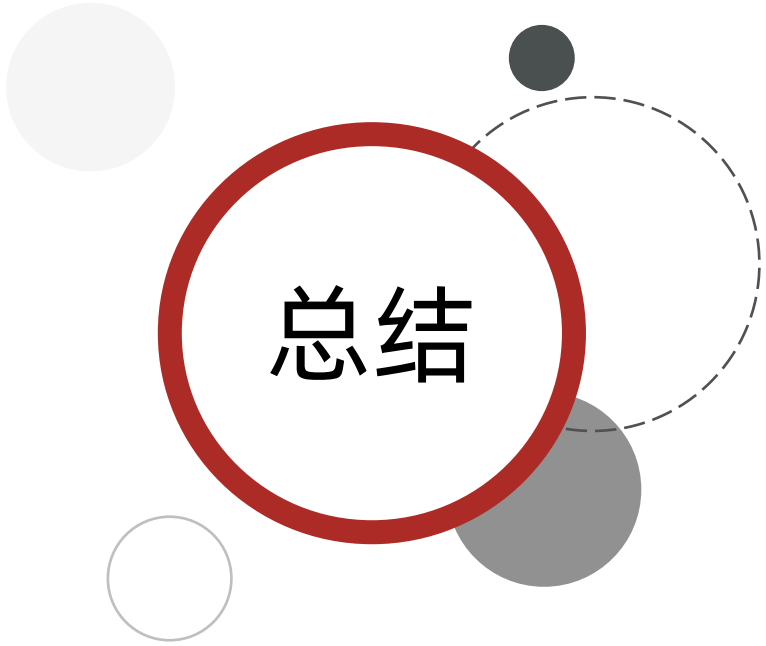
案例 **使用LinearSVC探索鸢尾花分类 – 惩罚参数C对超平面的影响展示**




C=30 margin显示图




C=0.01 margin显示图




## 1 LinearSVM Api的使用

from sklearn.svm import LinearSVC

## 2 惩罚参数C对超平面的影响

惩罚参数C影响：

C惩罚力度越大，则间隔越小；

C惩罚力度越小，则间隔越大

# 目录

Contents

# 学习目标

Learning Objectives

1.了解支持向量机的推导过程

## 支持向量机原理

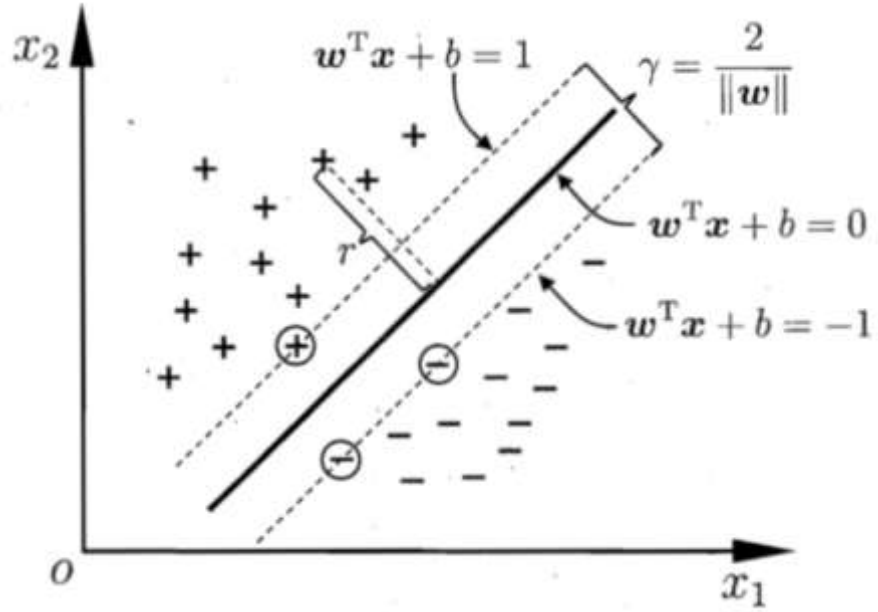
SVM思想：要去求一组参数（$w,b$),使其构建的超平面函数能够最优地分离两个集合。

样本空间中任意点x到超平面（$w,b$）的距离可写成：

$$r = \frac{|\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} + b|}{\|\boldsymbol{w}\|}$$

欲找到具有最大间隔的划分超平面，也就是要找到能满足下式中约束的参数$w$和$b$，使得间隔$\gamma$最大。

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




距离超平面最近的几个训练样本点使上式等号成立，他们被称为“支持向量”，两个异类支持向量到超平面的距离之和为：

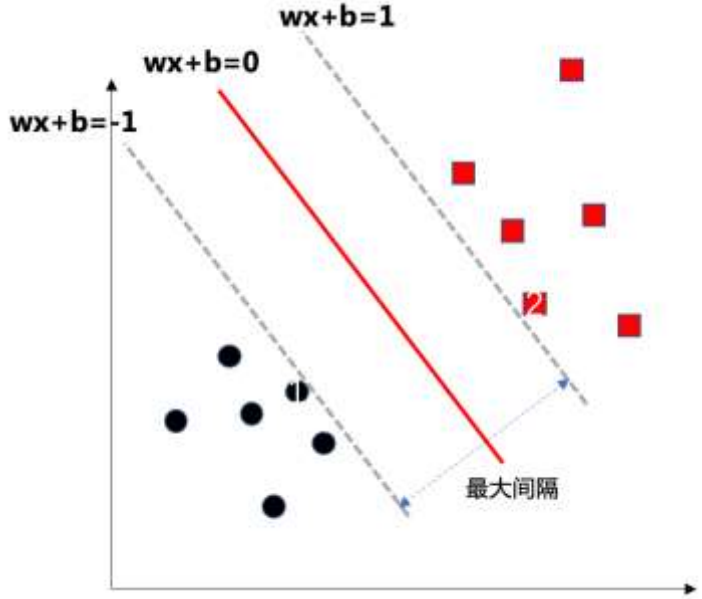
$$\gamma = \frac{2}{\|\boldsymbol{w}\|}$$

# 支持向量机原理

- 将所有样本正确分割开的基础上，求解最大间隔
  - 最大间隔距离表示：$\frac{2}{||w||}$
  - 训练样本 $\begin{cases} w^T x_i + b \geq +1, & y_i = +1 \\ w^T x_i + b \leq -1, & y_i = -1 \end{cases}$
  - 目标函数可写成

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

s.t. $y_i\ (w^T x_i + b) \geq 1$, 其中i = 1, 2, 3,...,m




- 目标函数进一步优化
  - $min_{w,b} = \frac{1}{2}\|w\|^2$

s.t. $y_i\ (w^T x_i + b) \geq 1$, 其中i = 1, 2, 3,...,m

## 支持向量机原理

添加核函数，将目标函数转换为以下形式

$$min_{w,b} = \frac{1}{2}\|w\|^2$$

$$\text{s.t.} \quad \sum_{i=1}^{n}\left(1 - y_i\left(w^T \cdot \Phi\left(x_i\right) + b\right)\right) \leq 0$$

构建拉格朗日函数： 其中 $a_i$ 为拉格朗日乘子（相当于$\lambda_i$ ）

- 拉格朗日乘子法构建的拉格朗日函数将目标优化函数和优化约束条件放在了一起，从而将带约束的极值问题转换为不带约束极值问题

$$L(w,b,\alpha) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{n}\alpha_i\left(y_i\left(w^T \cdot \Phi\left(x_i\right) + b\right) - 1\right)$$

要想求得极小值，上式后半部分应该取的极大值，最后转换成对偶问题

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

# 支持向量机原理

## 利用KKT求拉格朗日函数的极值

使用拉格朗日乘子法计算不等式约束的极值，结果必须满足3个条件，就是KKT条件

- $\frac{\partial L}{\partial x} = 0$ # 第1个条件：拉格朗日函数对变量求导为0
- $\lambda_i >= 0$ # 第2个条件：所有乘子大于等于0
- $\lambda_i \, g_i(x) = 0$ # 第3个条件：表示乘子和约束（ $g_i(x)$ 为约束）至少一个为零

第3个条件一般被称为互补松弛条件

**1 对 w 求偏导，并令其等于0：**

$$L = \frac{1}{2}||w||^2 - \sum_{i=1}^{n} \alpha_i \left(y_i w^T \varphi(x_i) + y_i b - 1\right)$$

$$= \frac{1}{2}||w||^2 - \sum_{i=1}^{n} \alpha_i y_i w^T \varphi(x_i) + \alpha_i y_i b - \alpha_i$$

$$\frac{\partial L}{\partial w} = w - \sum_{i=1}^{n} \alpha_i y_i \varphi(x_i) = 0$$

得到：$w = \sum_{i=1}^{n} a_i y_i \boldsymbol{\Phi}(x_i)$

**2 对 b 求偏导，并令其等于0：**

$$L = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{n} \alpha_i y_i w^T \varphi(x_i) + \alpha_i y_i b - \alpha_i$$

$$\frac{\partial L}{\partial b} = \sum_{i=1}^{n} \alpha_i y_i = 0$$

得到 $\sum_{i=1}^{n} a_i y_i = 0$

**3 将以上对$w$，$b$的偏导结果代入到L($w$，$b$，$\alpha$)中**

## 支持向量机原理

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

$$=\frac{1}{2}w^Tw-\sum_{i=1}^{n}(\alpha_i y_i w^T\varphi(x_i)+\alpha_i y_i b-\alpha_i)$$

$$=\frac{1}{2}w^Tw-\sum_{i=1}^{n}\alpha_i y_i w^T\varphi(x_i)-b\sum_{i=1}^{n}a_i y_i+\sum_{i=1}^{n}\alpha_i$$

$$=\frac{1}{2}w^Tw-\sum_{i=1}^{n}\alpha_i y_i w^T\varphi(x_i)+\sum_{i=1}^{n}\alpha_i$$

$$=\frac{1}{2}w^T\sum_{i=1}^{n}\alpha_i y_i\varphi(x_i)-w^T\sum_{i=1}^{n}\alpha_i y_i\varphi(x_i)+\sum_{i=1}^{n}\alpha_i$$

$$=\sum_{i=1}^{n}\alpha_i-\frac{1}{2}\left(\sum_{i=1}^{n}\alpha_i y_i\varphi(x_i)\right)^T\cdot\sum_{i=1}^{n}\alpha_i y_i\varphi(x_i)$$

$$=\sum_{i=1}^{n}\alpha_i-\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j\varphi^T(x_i)\,\varphi(x_j)$$

(1) $w=\sum_{i=1}^{n}a_i y_i\boldsymbol{\Phi}(x_i)$

(2) $\sum_{i=1}^{n}a_i y_i=0$

# 多元支持向量机原理-确定超平面

1 求解当 $\alpha$ 是什么值时拉格朗日函数最大

$$a^* = \arg\max_{\alpha} \left( \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \Phi^T(x_i) \Phi(x_j) \right)$$

2 将上述公式化简成极小值问题

$$\min_{\alpha} \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \left( \Phi^T(x_i) \cdot \Phi(x_j) \right) - \sum_{i=1}^{n} \alpha_i$$

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

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

3 将训练样本代入上面2步骤公式，求解出 $\alpha_i$ 值。将 $\alpha_i$ 值代入下面公式计算 $w, b$ 的值

$$w^* = \sum_{i=1}^{N} \alpha_i^* y_i \Phi(x_i) \qquad b^* = y_i - \sum_{i=1}^{N} \alpha_i^* y_i \left( \Phi(x_i) \cdot \Phi(x_j) \right)$$

4 求得分类超平面

$$w^* \Phi(x) + b^* = 0$$

## 练习

1、下列关于SVM实现原理的描述正确的是（**多选**）：

A）SVM的分类决策函数就是它的分类超平面的代数表达式

B）核函数的作用就是将特征映射到更高维度的特征空间

C）离SVM分割超平面最近且满足条件的样本点叫做支撑向量

D）分割间距越小SVM的分割性能越好

答案解析： 分割间距越大代表SVM的分割性能越好 D错

答案：ABC

# 目录

Contents

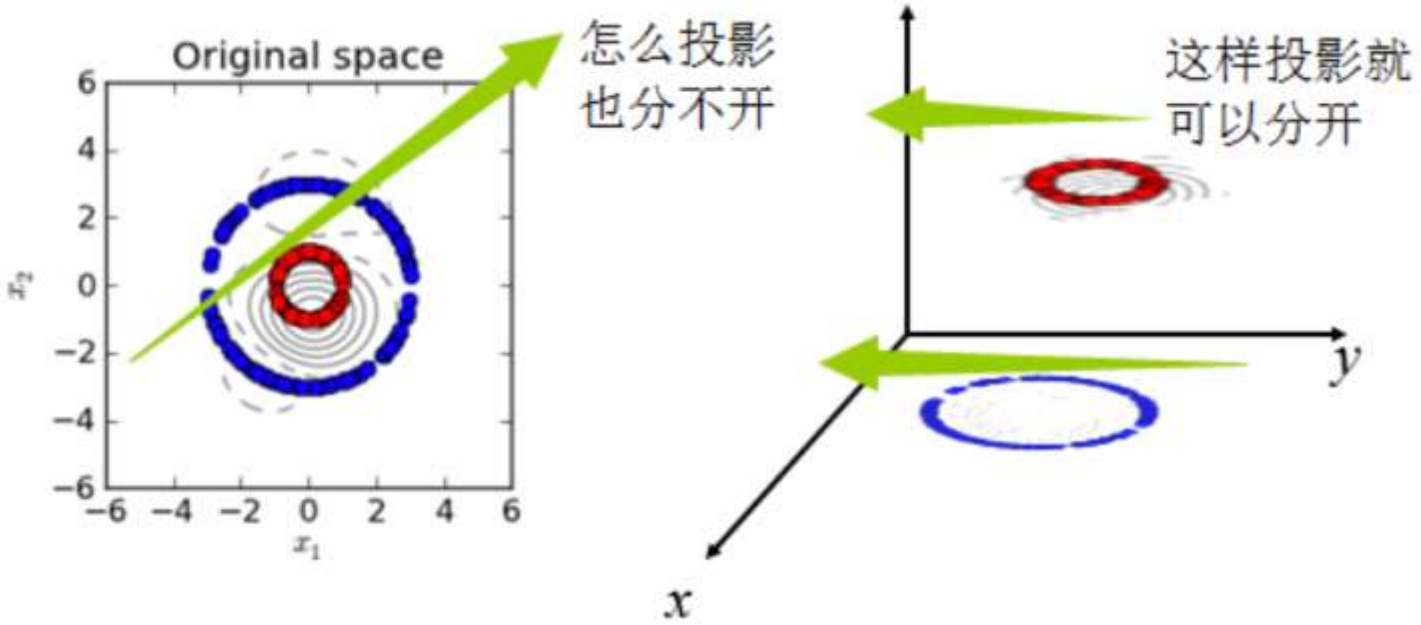
# 学习目标

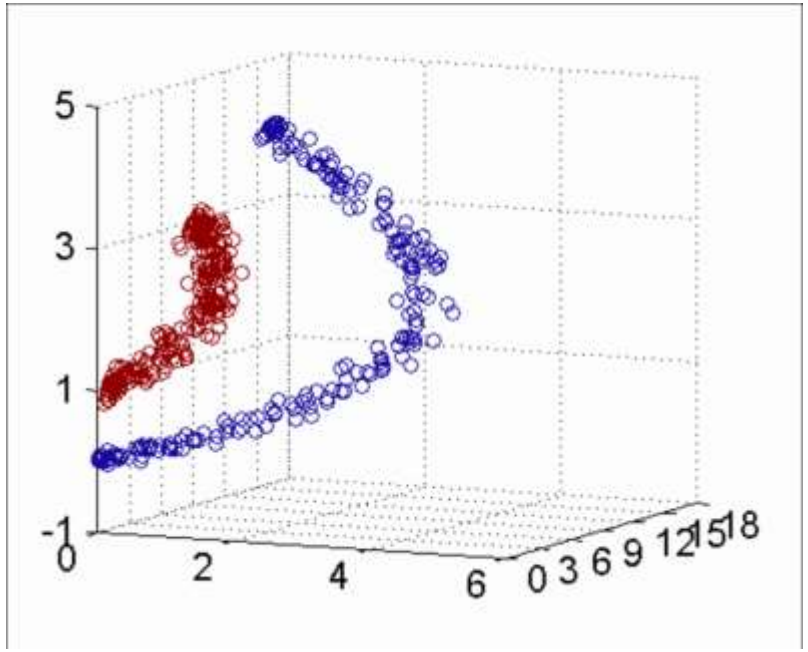
Learning Objectives

1. 理解核函数的作用
2. 知道核函数的分类
3. 知道高斯核函数中γ的作用
4. 实践高斯核函数的API

# 核函数的作用

核函数将原始输入空间映射到新的特征空间，从而，使原本线性不可分的样本可能在核空间可分。



## 核函数分类

| 名称 | 表达式 | 参数 |
| --- | --- | --- |
| 线性核 | $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{x}_j$ | |
| 多项式核 | $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = (\boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{x}_j)^d$ | $d \geqslant 1$ 为多项式的次数 |
| 高斯核 | $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}{2\sigma^2}\right)$ | $\sigma > 0$ 为高斯核的带宽(width) |
| 拉普拉斯核 | $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|}{\sigma}\right)$ | $\sigma > 0$ |
| Sigmoid 核 | $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \tanh(\beta \boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{x}_j + \theta)$ | tanh 为双曲正切函数, $\beta > 0, \theta < 0$ |

- 线性核：一般是不增加数据维度，而是预先计算内积，提高速度
- 多项式核：一般是通过增加多项式特征，提升数据维度，并计算内积
- 高斯核（RBF、径向基函数）：产生将样本投射到无限维空间的运算效果，使得原来不可分的数据变得可分。使用最多
- 其他了解即可

## 高斯核函数 - 基本原理1

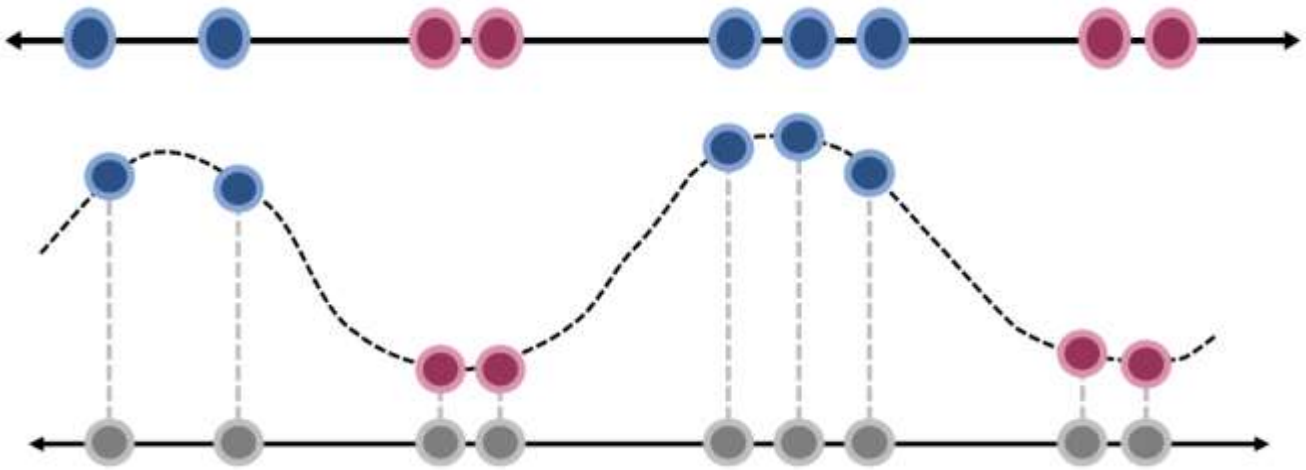
- 高斯核 Radial Basis Function Kernel (径向基函数，又称RBF核)

$$K(x, y) = e^{-\gamma ||x - y||^2}$$

gamma是超参数，作用与标准差相反

**结论：**gamma越大，高斯分布越窄，gamma越小，高斯分布越宽

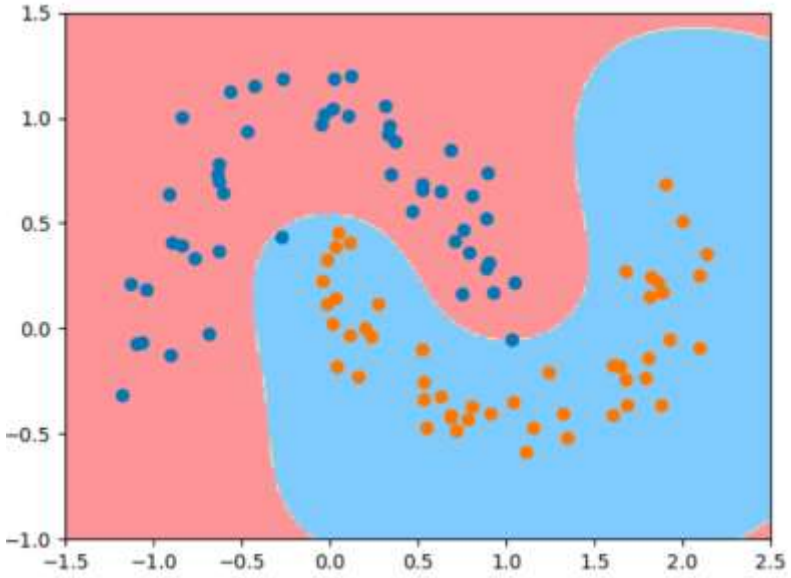
- 举个栗子



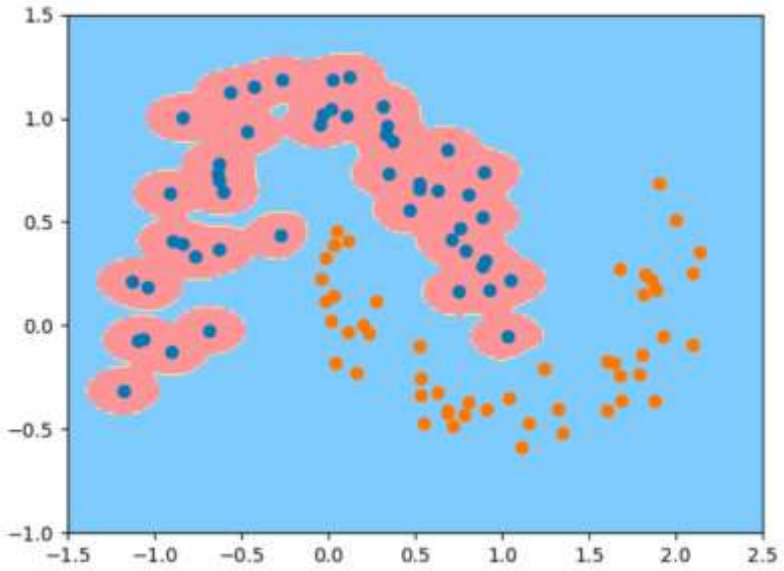
# 高斯核函数 - sklearn API和超参数gamma

- 高斯核API
  - **from sklearn.svm import SVC**
  - SVC(kernel='rbf',gamma=gamma)

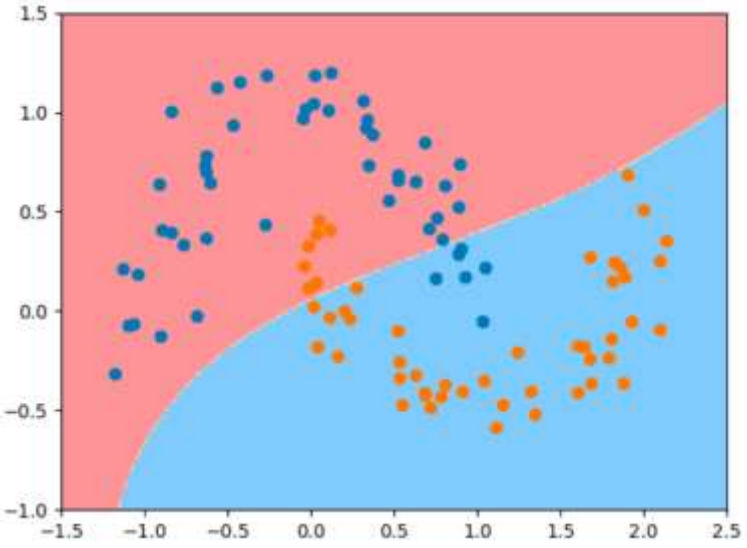
举个栗子 来说明gamma对模型的影响



gamma =1.0 模型效果合适



gamma =100模型效果过拟合



gamma = 0.1模型效果欠拟合

# 高斯核函数 - sklearn API和超参数gamma

- Gammar参数意义解释

高斯核函数(RBF)，其中$\gamma$ 为超参数

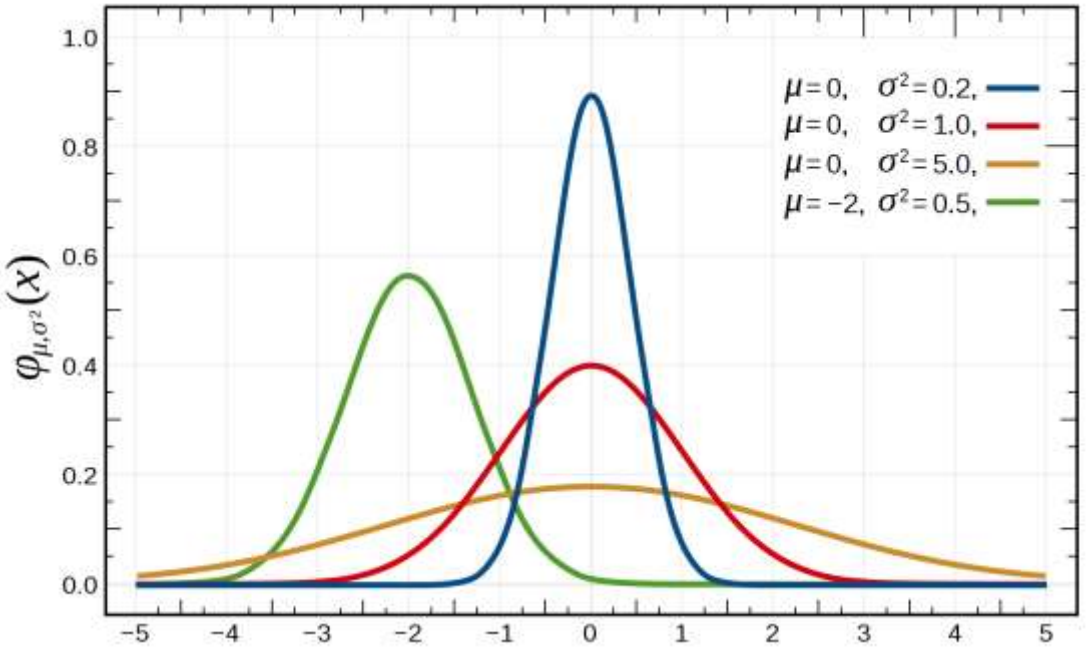
对比高斯函数，以及正态分布的图

$$K(x,y) = e^{-\gamma||x-y||^2}$$

$$g(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{x-\mu}{\sigma})^2}$$

高斯核函数中的超参数gammar ($\gamma$)

等价于高斯函数中的 $\frac{1}{2\sigma^2}$




- 标准差越大，数据越分散，图像越宽
- 标准差越小，数据越集中，图像越窄
- gamma越大，高斯分布越窄 （数据变化越剧烈，易过拟合）
- gamma越小，高斯分布越宽（数据变化越平缓，易欠拟合）

# 高斯核函数

```
# 1.导入依赖包
from sklearn import datasets
import matplotlib.pyplot as plt
from sklearn.svm import SVC
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler
import numpy as np

def dm01_svm高斯核():
    # 2.构建数据
    X, y = datasets.make_moons(noise=0.15, random_state=22)
    plt.scatter(X[y == 0, 0], X[y == 0, 1])
    plt.scatter(X[y == 1, 0], X[y == 1, 1])
    plt.show()

    # 3.定义RBF核
    def RBFKernelSVC(gamma=1.0):
        return Pipeline([
            ('std_scaler', StandardScaler()),
            ('svc', SVC(kernel='rbf', gamma=gamma)) ])

    # 4.1 实例化模型1
    mysvc1 = RBFKernelSVC(gamma=1.0)
    mysvc1.fit(X, y)
    # 画图
    plot_decision_boundary(mysvc1, axis=[-1.5, 2.5, -1.0, 1.5])
    plt.scatter(X[y == 0, 0], X[y == 0, 1])
    plt.scatter(X[y == 1, 0], X[y == 1, 1])
    plt.show()
```

```
    # 4.2 实例化模型2 - 过拟合
    mysvc2 = RBFKernelSVC(gamma=100)
    mysvc2.fit(X, y)
    plot_decision_boundary(mysvc2, axis=[-1.5, 2.5, -1.0, 1.5])
    plt.scatter(X[y == 0, 0], X[y == 0, 1])
    plt.scatter(X[y == 1, 0], X[y == 1, 1])
    plt.show()

    # 4.3 实例化模型3 - 欠拟合
    mysvc3 = RBFKernelSVC(gamma=0.1)
    mysvc3.fit(X, y)
    plot_decision_boundary(mysvc3, axis=[-1.5, 2.5, -1.0, 1.5])
    plt.scatter(X[y == 0, 0], X[y == 0, 1])
    plt.scatter(X[y == 1, 0], X[y == 1, 1])
    plt.show()
```

# 总结

## 1 核函数的作用

是将原始输入空间映射到新的特征空间，使得原本线性不可分的样本可能在核空间可分

## 2 核函数分类

| 名称 | 表达式 | 参数 |
| --- | --- | --- |
| 线性核 | $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{x}_j$ | |
| 多项式核 | $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = (\boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{x}_j)^d$ | $d \geqslant 1$ 为多项式的次数 |
| 高斯核 | $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}{2\sigma^2}\right)$ | $\sigma > 0$ 为高斯核的带宽(width) |
| 拉普拉斯核 | $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|}{\sigma}\right)$ | $\sigma > 0$ |
| Sigmoid 核 | $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \tanh(\beta \boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{x}_j + \theta)$ | tanh 为双曲正切函数, $\beta > 0, \theta < 0$ |

## 3 高斯核的γ 超参数

- gamma越大，高斯分布越窄，易过拟合
- gamma越小，高斯分布越宽，易欠拟合

## 练习

1、下列关于SVM中核函数的描述错误的是：

A）引入核函数的目的就是为了解决线性不可分的问题

B）核函数往往是将原始特征向更低维度的空间进行映射

C）常用的核函数有线性核、高斯核、sigmoid核

D） 在大多数场景下，高斯核都能取得不错的效果

答案解析：应该是向更高维度映射

答案：B


黑马程序员
www.itheima.com
传智教育旗下高端IT教育品牌