SVM

课前准备

- 下载Anaconda软件，请点击这里进行下载。
- 复习逻辑回归算法。
  - 分类原理。
  - 损失函数。
  - 决策函数。
  - 决策边界。

本节要点

- 函数极值的求解。
  - 无条件极值。
  - 条件极值。
    - 拉格朗日乘数法。
    - KKT条件。

函数极值

极值点与驻点

- 函数的极值点可能在驻点与不可导点获得。
  - 驻点并一定是极值点，极值点也不一定是驻点。
  - x^3在0处为函数的驻点，但不是极值点。
  - $|x|$在0处为函数的极值点，但不是驻点。
- 如果函数在定义域内处处可导，则驻点是函数取得极值的必要条件，但不是充分条件。
  - 驻点不一定是极值点。
  - 极值点一点是驻点。
- 如果函数是凸函数，则驻点是函数取得极值点的充要条件。
  - 驻点是极值点。
  - 极值点也是驻点。

极值分类

在求解函数极值时，可以分为以下两种：

1. 无条件极值。
2. 条件极值。
   - 等式约束条件。
   - 不等式约束条件。

无条件极值

对于函数的自变量，除了限定在函数的定义域以内，没有其他限定条件，这样的极值问题称为**无条件极值**。

课堂练习

从机器学习的角度看，如果在无约束条件下，我们求解损失函数的极小值（无条件极值），可以使用（ ）。【不定项】

A 对损失函数求导，令导函数为0。

B 对损失函数取对数。

C 梯度下降。

D 穷举迭代。

示例

给定函数$fx)$：

$$f(x) = x^2 - 4*x + 4$$

$f(x)$是凸函数，具有唯一的极小值，定义域$(-\infty, +\infty)$。根据之前的介绍，凸函数的极值一定在驻点处取得，我们可以对$f(x)$求导，令导函数为0。

$$f'(x) = 2*x - 4 = 0$$

最终的结果为：

$$x = 2。$$

条件极值（一）

对自变量除了限定在函数的定义域以内，还有其他的限制条件，这种极值问题称为**条件极值**。例如，某厂家设计长方体容器，在表面积确定的情况下，如何才能使得体积最大？此时，我们可以通过**拉格朗日乘数法**来求解这种条件极值问题。

拉格朗日乘数法

我们以极小值为例，在求解函数$f(x)$在附加条件$g(x) = 0$下的极值点：

$$\min_{x} f(x)$$

$$s.t. g_i(x) = 0, i = 1, 2, \ldots, m$$

可以引入辅助函数:

$$L(x, \alpha) = f(x) + \sum_{i=1}^{m} \alpha_i g_i(x)$$

其中，$L(x)$称为**拉格朗日函数**，α称为**拉格朗日乘子**。

极值求解方法

然后，我们求$L(x)$对x与α的一阶偏导数，令偏导数为0，如下:

$$\begin{cases} \nabla_x L(x, \alpha) = 0 \\ \nabla_\alpha L(x, \alpha) = 0 \end{cases}$$

由以上方程组求解出x与α，x就是函数$f(x)$在附加条件$g(x) = 0$下可能的极值点，如果函数是凸函数，则x一定是极值点。

说明:

- 如果f是多元函数，则需要对每个自变量x求偏导，联立方程组。
- 如果附加限制条件存在多个，则需要对每个α_i求偏导，联立方程组。
- 最终，联立的方程组含有n个方程与n个未知数。
- $L(x)$对α求偏导，相当于原约束条件$g(x) = 0$。

示例

某长方体的长，宽，高分别为x，y与z，表面积为36，求最大的长方体体积。

由题意可知:

$$\begin{cases} f(x, y, z) = xyz \\ g(x, y, z) = 2xy + 2yz + 2xz - 36 = 0 \end{cases}$$

我们引入拉格朗日函数，如下:

$$L(x, y, z) = xyz + \lambda(2xy + 2yz + 2xz - 36)$$

求拉格朗日函数对x，y与z的偏导数，并使之为0，得到:

$$\begin{cases} yz + 2\lambda(y + z) = 0 \\ xz + 2\lambda(x + z) = 0 \\ xy + 2\lambda(y + x) = 0 \\ 2xy + 2yz + 2xz - 36 = 0 \end{cases}$$

求得结果为:

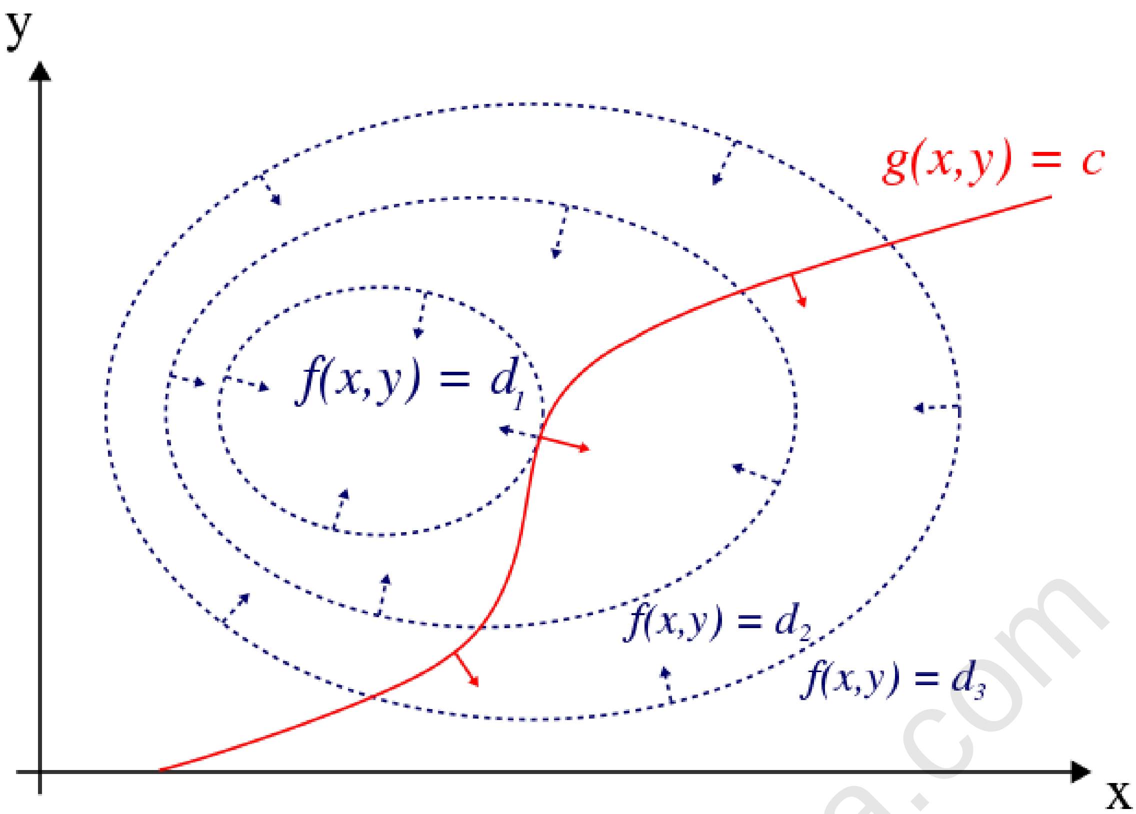
$$\begin{cases} x = \sqrt{6} \\ y = \sqrt{6} \\ z = \sqrt{6} \\ \lambda = -\frac{\sqrt{6}}{4} \end{cases}$$

因此，最大体积为：$6\sqrt{6}$。

拉格朗日原理解析

单个约束条件

我们为什么可以采用上述的方法来求解极值呢？首先，我们来看含有单个约束条件的情况。下图中给出了直观的解释。



在上图，蓝色为二元函数$f(x)$的等高线，红色为约束条件$g(x) = c$。

两个函数在相交点不会取得极值。因为一定还存在f的其他等高线与g相交或相切，使得f值更大或更小。因此，f会在切点处取得极值，由于梯度的方向是垂直于等高线的，因此，在切点（极值点）处，f与g的梯度平行（共线），二者的梯度向量成比例，有：

$$\nabla_x f(x) = \alpha \nabla_x g(x)$$

而这个条件，正是$L(x,\alpha)$对x求梯度后的结果，因此，我们可以得到如下的结论：

$$L(x,\alpha)=f(x)+\alpha g(x)$$

$$\begin{cases}\nabla_x f(x)=\alpha\nabla_x g(x)\\ g(x)=0\end{cases}\Leftrightarrow\begin{cases}\nabla_x L(x,\alpha)=0\\ \nabla_\alpha L(x,\alpha)=0\end{cases}$$

课堂练习

如果存在极值条件，对于α，其取值范围为（ ）。

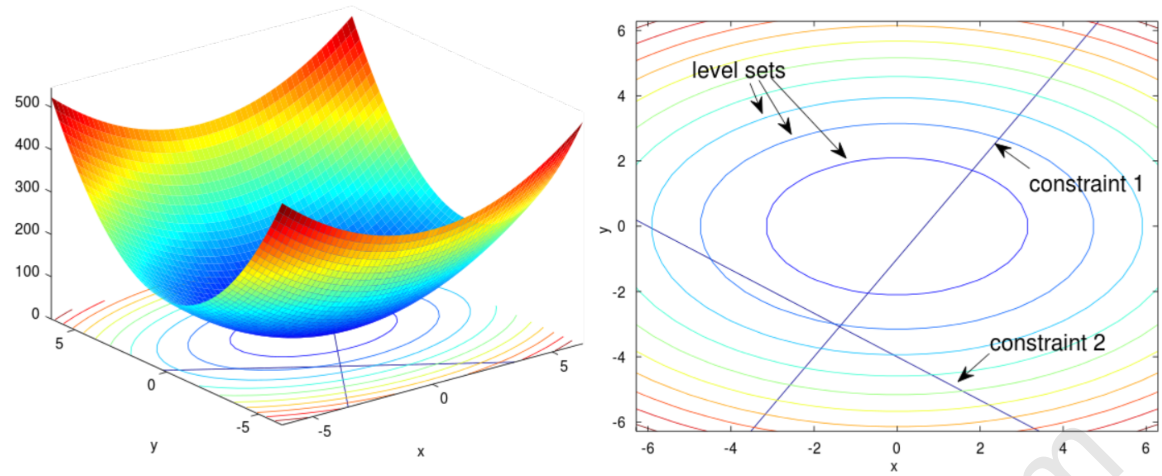
A $\alpha=0$

B $\alpha>0$

C $\alpha<0$

D $\alpha\neq 0$

多个约束条件



在多个约束条件下，相交点的位置上，函数f的梯度并要求与其中任何一个约束函数的梯度平行，而是要求是这些约束函数梯度的线性组合。

$$\nabla_x f(x)=\sum_{i=1}^m\alpha_i\nabla g_i(x)$$

因此，我们有如下的结论：

$$L(x,\alpha)=f(x)+\sum_{i=1}^m\alpha_i g_i(x)$$

$$\begin{cases}\nabla_x f(x)=\alpha_i\nabla_x g_i(x)\\ g_i(x)=0,i=1,2,\ldots,m\end{cases}\Leftrightarrow\begin{cases}\nabla_x L(x,\alpha)=0\\ \nabla_\alpha L(x,\alpha)=0\end{cases}$$

条件极值（二）

刚才我们讨论的条件极值，是等式约束的条件，实际上，条件极值也可以是不等式约束条件。

我们依然以极小值为例，在求解函数$f(x)$在附加条件$h(x)=0$下的极值点：

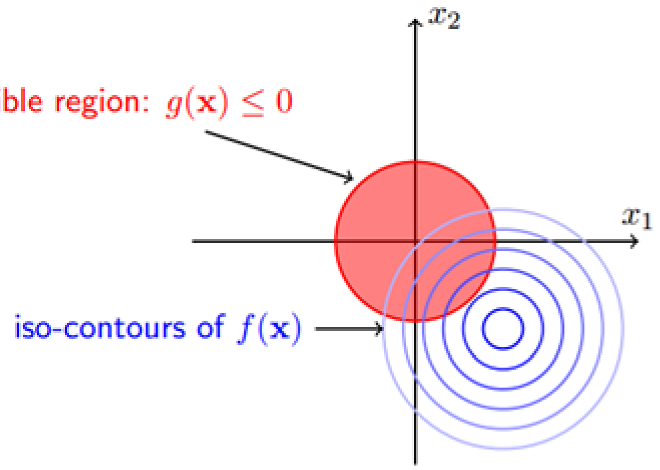
$$\min_{x} f(x)$$

$$s.t.\ g_i(x) \leqslant 0, i=1,2,\ldots,m$$

如果同样构造拉格朗日函数，格式为：

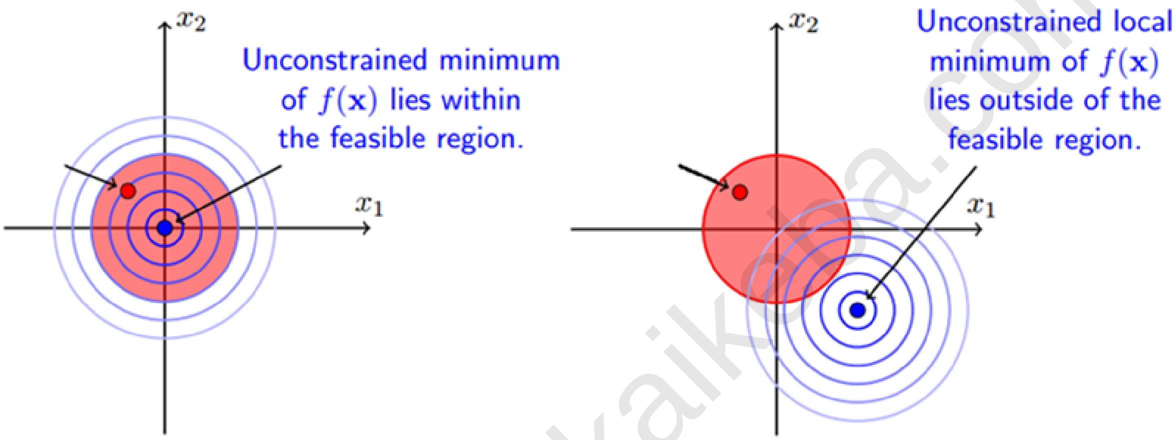
$$L(x,\alpha)=f(x)+\sum_{i=1}^{m}\beta_i g_i(x)$$

松弛互补条件



由上图可知，可行解需要限定在函数$g(x)\leqslant 0$的区域内，这包含两种情况：

1. 可行解落在函数$g(x)<0$的区域内时，此时相当于没有约束条件（$\beta=0$），直接最小化f即可。
2. 可行解落在函数$g(x)=0$的区域时，此时等价于等式约束（$\beta\neq 0$）。



由以上两点，我们能够得出一个结论，可行解必须满足如下条件：

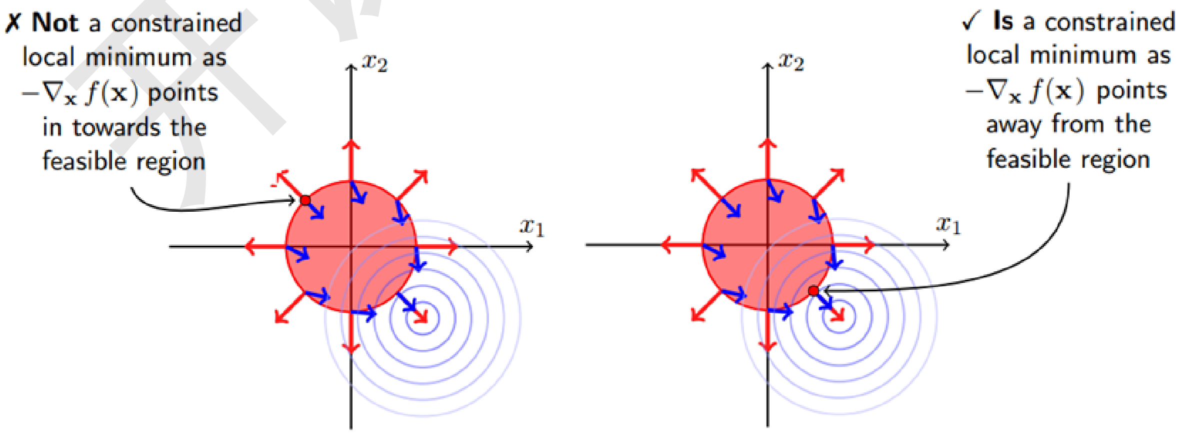
$$\beta g(x)=0$$

我们将该条件称为**松弛互补条件**。

梯度方向解析

当$g(x)=0$时，相当于等值约束条件，不过此时与等值约束条件还有一些不同。对于极值点处的目标函数f与约束函数g:

- 等值约束中，二者梯度只要平行即可。
    - 梯度同向还是反向都可以。
- 在不等值约束中，二者梯度依然平行。
    - f的负梯度方向与g的梯度方向相同。
    - f的负梯度方向应该远离约束区域，否则，如果朝向约束区域，就一定可以沿着该区域，使得f的值进一步减小。



根据上述的介绍，我们可以得到如下结论：

$$
\begin{aligned}
&-\nabla_x f(x)=\beta \nabla g_x(x) \\
&\beta>0
\end{aligned}
$$

KKT条件

实际上，对于不等式约束，只要满足一定的前提条件，依然可以通过拉格朗日乘数法来解决，我们将这些前提条件称为**KKT条件（Karush-Kuhn-Tucker Conditions）**。

综合之前的介绍，我们给出更综合的形式：

$$
\begin{aligned}
s.t. h_i(x)&=0, i=1,2,\ldots,m \\
g_j(x)&\leqslant 0, i=1,2,\ldots,n
\end{aligned}
$$

如果同样构造拉格朗日函数，格式为：

$$
L(x,\alpha,\beta)=f(x)+\sum_{i=1}^m \alpha_i h_i(x)+\sum_{j=1}^n \beta_j g_j(x)
$$

根据以上的分析，可行解x需要满足如下条件（KKT条件）：

$$\begin{cases} \nabla_x L(x,\alpha,\beta)=0 & (1) \\ \beta_j g_j(x)=0, j=1,2,\ldots,n & (2) \\ h_i(x)=0, i=1,2,\ldots,m & (3) \\ g_j(x)\leqslant 0, j=1,2,\ldots,n & (4) \\ \beta_j \geqslant 0, j=1,2,\ldots,n & (5) \end{cases}$$

条件解析

我们依次对KKT条件进行解析。

1. 在极值点，目标函数的梯度与约束函数的梯度平行（单约束条件）或与约束函数梯度的线性组合平行（多约束条件）。
2. 松弛互补条件。
3. 等式约束条件。
4. 不等式约束条件。
5. 在不等式约束下，目标函数的负梯度与约束函数梯度方向相同。

逻辑回归算法回顾

决策函数

之前，我们学习过逻辑回归等分类算法，也熟知该算法的分类原理。我们以二分类为例，对于逻辑回归来说，通过决策函数$F(X)$（返回值为z）来实现分类，如下：

$$F(X)=\vec{w}^T\vec{x}$$

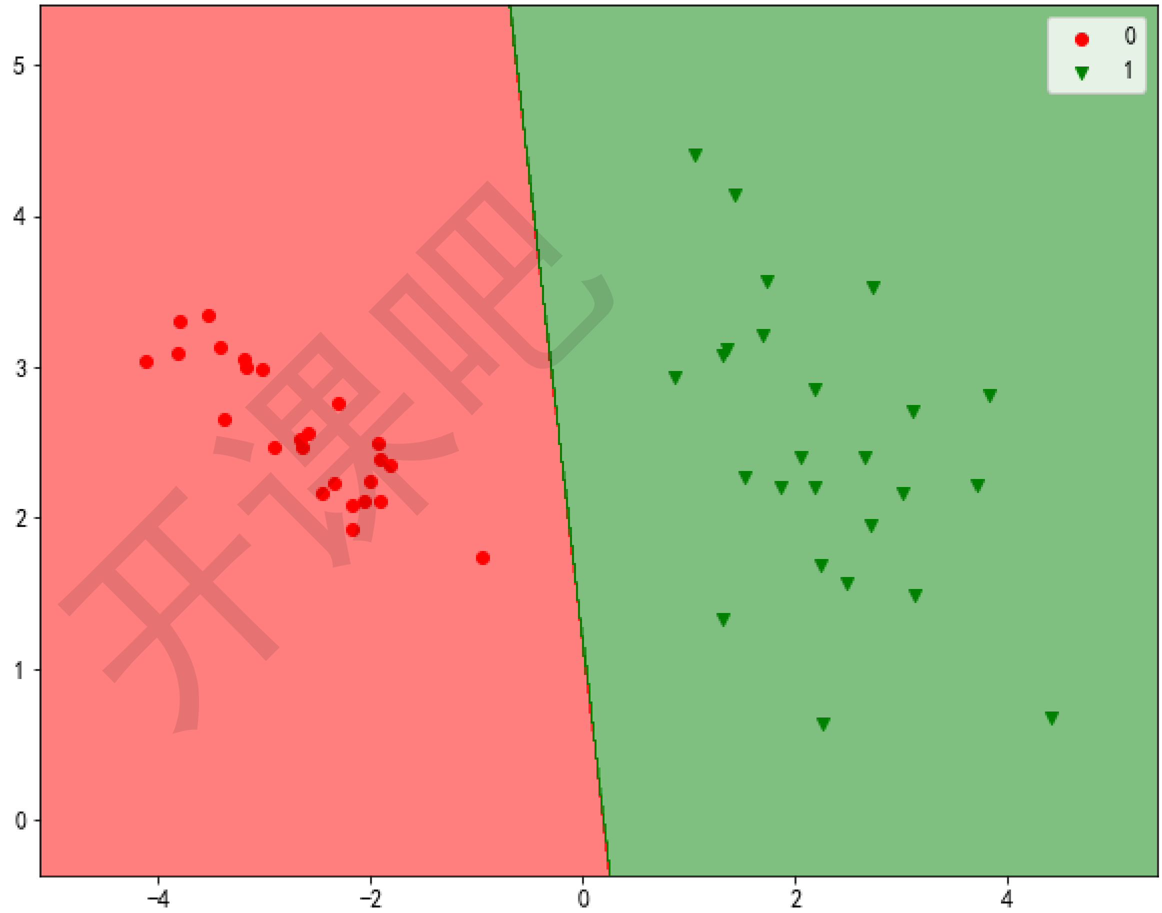
$$\begin{cases} F(X)>0 & 1 \\ F(X)\leqslant 0 & 0 \end{cases}$$

决策边界

从空间几何的角度来讲，决策函数确定决策边界，将样本划分在决策边界的两侧，从而实现分类的效果。在逻辑回归算法中，决策边界的方程为：

$$F(X)=0 \quad \Rightarrow \quad \vec{w}^T\vec{x}=0$$

在n维空间中，对于线性分类器，决策边界就是$n-1$维的超平面。在本节中，为了方便理解，我们都是使用二维空间来举例，此时的超平面就是一维的直线。



SVM算法介绍

SVM分类思想

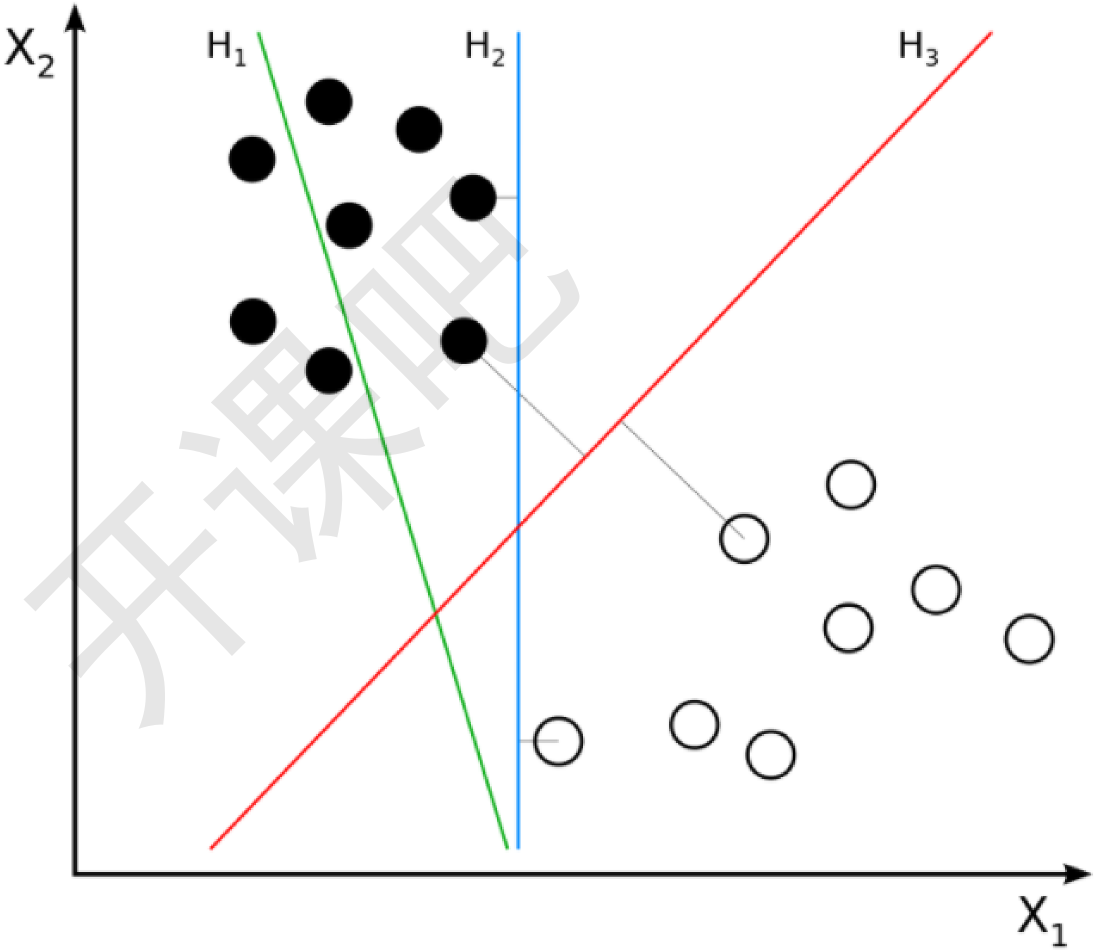
如果目的仅是能够正确分类，则决策边界（超平面）可能并不是唯一的，我们也许能画出很多种不同的决策边界。下图中，那个分类的效果最好？

A H1

B H2

C H3

D H2与H3



SVM（Support Vector Machine），即支持向量机，可用于分类，回归或异常值检测等任务。该算法最早用于解决二分类任务，其思想为在高维空间中构建一个（或一组）超平面（决策边界），使得在正确分类的同时，能够让离超平面最近的样本，到超平面的距离尽可能的远（算法优化目标）。而距离超平面最近的样本，我们称为**支持向量**。

支持向量到超平面的距离较远，可以使得模型具有更好的泛化能力，降低过拟合的风险。

课堂练习

正例的支持向量与负例的支持向量，到超平面（决策边界）的距离一定是相等的。这种说法正确吗?
A 正确
B 不正确
C 不确定

损失函数

点到直线的距离

在二维空间中，点(x_0, y_0)到直线$Ax+By+C=0$的距离公式为：

$$d = \frac{|Ax_0+By_0+C|}{\sqrt{A^2+B^2}}$$

- d：点到直线的距离。

如果将其扩展到n维空间，样本点$(x_1, x_2, \ldots, x_n)$到超平面$\vec{w}^T\vec{x}+b=0$的距离为：

$$d = \frac{|\vec{w}^T\overrightarrow{x_0}+b|}{||\vec{w}||}$$

$$\overrightarrow{x_0} = (x_1, x_2, \ldots, x_n)^T$$

$$||\vec{w}|| = \sqrt{w_1^2 + w_2^2 + \cdots + w_n^2}$$

- $\overrightarrow{x_0}$：样本点空间坐标的向量表示。
- $||\vec{w}||$：向量$\vec{w}$的二范数。

损失函数确定

在SVM算法中，我们的优化目标，就是让支持向量到决策边界的距离，尽可能的远。我们设决策函数为：

$$\vec{w}^T\vec{x}+b=0$$

而支持向量到该决策边界（超平面）的距离为d，则对于任意的样本$x^{(i)}$，有：

$$\frac{|\vec{w}^T\vec{x}^{(i)}+b|}{||\vec{w}||} \geqslant d \qquad \Rightarrow$$

$$\begin{cases} \frac{\vec{w}^T\vec{x}^{(i)}+b}{||\vec{w}||} \geqslant d & \forall y^{(i)} = 1 \\ \frac{\vec{w}^T\vec{x}^{(i)}+b}{||\vec{w}||} \leqslant -d & \forall y^{(i)} = -1 \end{cases}$$

在方程两侧同时除以d，得到：

$$\begin{cases} \frac{\vec{w}^T\vec{x}^{(i)}+b}{||\vec{w}||d} \geqslant 1 & \forall y^{(i)} = 1 \\ \frac{\vec{w}^T\vec{x}^{(i)}+b}{||\vec{w}||d} \leqslant -1 & \forall y^{(i)} = -1 \end{cases}$$

分子与分母同时除以$||\vec{w}||d$，得到：

$$\begin{cases} \overrightarrow{w'}^T \vec{x}^{(i)} + b' \geqslant 1 & \forall y^{(i)} = 1 \\ \overrightarrow{w'}^T \vec{x}^{(i)} + b' \leqslant -1 & \forall y^{(i)} = -1 \end{cases}$$

对于之前提到的决策函数：

$$\vec{w}^T \vec{x} + b = 0$$

我们同样可以令方程两侧同时除以$||\vec{w}||d$，得到：

$$\overrightarrow{w'}^T \vec{x} + b' = 0$$

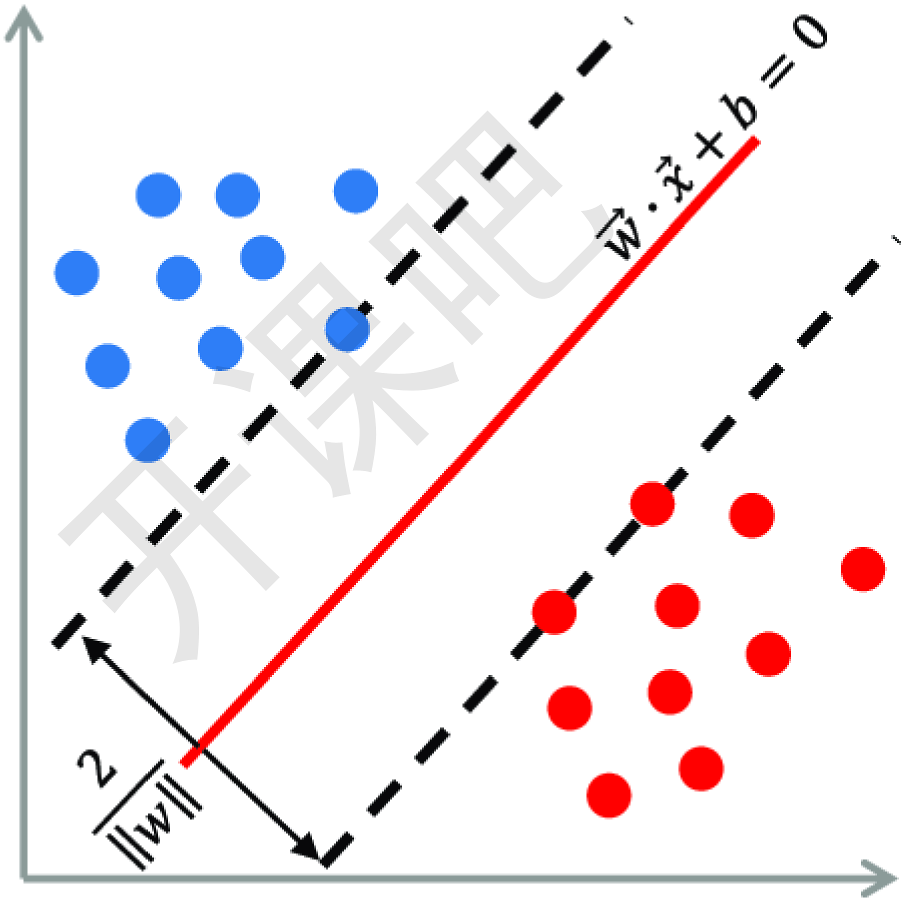
由于w'与b'仅仅是一个符号而已，因此，我们将其改回w与b，得到：

$$\begin{cases} \vec{w}^T \vec{x}^{(i)} + b \geqslant 1 & \forall y^{(i)} = 1 & （1） \\ \vec{w}^T \vec{x} + b = 0 \\ \vec{w}^T \vec{x}^{(i)} + b \leqslant -1 & \forall y^{(i)} = -1 & （2） \end{cases}$$

上面的方程具有相同的权重，因此：

$$\begin{cases} \vec{w}^T \vec{x} + b = 1 & （3） \\ \vec{w}^T \vec{x} + b = 0 & （4） \\ \vec{w}^T \vec{x} + b = -1 & （5） \end{cases}$$

（3），（4）与（5）是3条平行的超平面，且正例（$y=1$）的支持向量$x^{(i)}$满足方程（3），负例（$y=-1$）的支持向量$x^{(i)}$满足方程（5）。



对于任意的支持向量$x^{(i)}$，样本到决策边界的距离：

$$\frac{|\vec{w}^T\vec{x}^{(i)}+b|}{||\vec{w}||}=\frac{1}{||\vec{w}||}$$

我们要使得这个距离最大，只需要使得其倒数最小即可。

$$max\frac{1}{||\vec{w}||}\Rightarrow min||\vec{w}||\Rightarrow min\frac{1}{2}||\vec{w}||^2$$

$$s.t.\quad y^{(i)}(\vec{w}^T\vec{x}^{(i)}+b)\geqslant 1\quad\text{【根据（1）与（2）】}$$

课堂练习

我们为什么会有这个受限条件?
A 没有也可以。
B 为了避免模型过拟合。
C 为了避免模型欠拟合。
D 为了确保分类正确性。

参数求解

很明显，SVM的损失函数具有额外的约束条件，是一个不等式条件极值问题，我们通过构造拉格朗日函数来实现求解。

$$L(w,b,\beta)=\frac{1}{2}\|w\|_2^2+\sum_{i=1}^{m}\beta_i\left[1-y^{(i)}\left(w^Tx^{(i)}+b\right)\right],\ \beta_i\geq 0$$

$$\min_{w,b}\max_{\beta\geq 0}L(w,b,\beta)\ \Rightarrow \max_{\beta\geq 0}\min_{w,b}L(w,b,\beta)$$

$$\frac{\partial L}{\partial w}=0\Rightarrow w-\sum_{i=1}^{m}\beta_i y^{(i)}x^{(i)}=0\Rightarrow w=\sum_{i=1}^{m}\beta_i y^{(i)}x^{(i)}$$

$$\frac{\partial L}{\partial b}=0\Rightarrow -\sum_{i=1}^{m}\beta_i y^{(i)}=0\Rightarrow \sum_{i=1}^{m}\beta_i y^{(i)}=0$$

$$\begin{aligned}
\ell(\beta)&=\frac{1}{2}\|w\|_2^2+\sum_{i=1}^{m}\beta_i\left[1-y^{(i)}\left(w^Tx^{(i)}+b\right)\right]\\
&=\frac{1}{2}\|w\|_2^2-\sum_{i=1}^{m}\beta_i\left[y^{(i)}\left(w^Tx^{(i)}+b\right)-1\right]\\
&=\frac{1}{2}w^Tw-\sum_{i=1}^{m}\beta_i y^{(i)}w^Tx^{(i)}-\sum_{i=1}^{m}\beta_i y^{(i)}b+\sum_{i=1}^{m}\beta_i\\
&=\frac{1}{2}w^T\sum_{i=1}^{m}\beta_i y^{(i)}x^{(i)}-w^T\sum_{i=1}^{m}\beta_i y^{(i)}x^{(i)}-b\sum_{i=1}^{m}\beta_i y^{(i)}+\sum_{i=1}^{m}\beta_i\\
&=-\frac{1}{2}w^T\sum_{i=1}^{m}\beta_i y^{(i)}x^{(i)}-b\sum_{i=1}^{m}\beta_i y^{(i)}+\sum_{i=1}^{m}\beta_i\\
&=-\frac{1}{2}\left(\sum_{i=1}^{m}\beta_i y^{(i)}x^{(i)}\right)^T\left(\sum_{i=1}^{m}\beta_i y^{(i)}x^{(i)}\right)+\sum_{i=1}^{m}\beta_i\\
&=-\frac{1}{2}\sum_{i=1}^{m}\beta_i y^{(i)}x^{(i)^T}\sum_{i=1}^{m}\beta_i y^{(i)}x^{(i)}+\sum_{i=1}^{m}\beta_i\\
&=\sum_{i=1}^{m}\beta_i-\frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\beta_i\beta_j y^{(i)}y^{(j)}x^{(i)^T}x^{(j)}
\end{aligned}$$

$$\ell(\beta)=\sum_{i=1}^{m}\beta_i-\frac{1}{2}\sum_{i=1,j=1}^{m}\beta_i\beta_j y^{(i)}y^{(j)}x^{(i)^T}x^{(j)}$$

$$s.t:\sum_{i=1}^{m}\beta_i y^{(i)}=0$$

$$\beta_i\geq 0, i=1,2,\ldots,m$$

$$\begin{aligned}&\max_{\beta\geq 0}\ell(\beta)\\ &s.t:\sum_{i=1}^{m}\beta_i y^{(i)}=0\end{aligned}\quad\Rightarrow\quad\begin{aligned}&\min_{\beta\geq 0}-\ell(\beta)\\ &s.t:\sum_{i=1}^{m}\beta_i y^{(i)}=0\end{aligned}$$

$$\min_{\beta\geq 0}\frac{1}{2}\sum_{i=1,j=1}^{m}\beta_i\beta_j y^{(i)}y^{(j)}x^{(i)^T}x^{(j)}-\sum_{i=1}^{m}\beta_i$$

$$s.t:\sum_{i=1}^{m}\beta_i y^{(i)}=0$$

$$w^{*}=\sum_{i=1}^{m}\beta_{i}^{*}y^{(i)}x^{(i)}$$

$$y\left(w^{T}x+b\right)=y\left(\sum_{i=1}^{m}\beta_{i}^{*}y^{(i)}x^{(i)T}x+b\right)=1$$

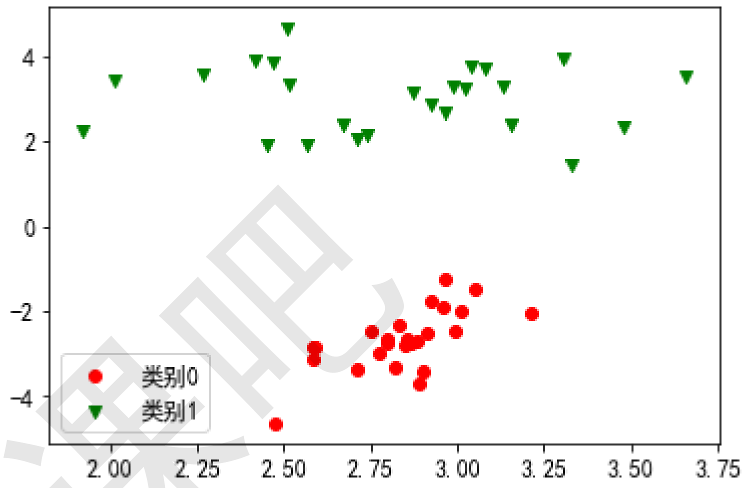
$$b^{*}=y-\sum_{i=1}^{m}\beta_{i}^{*}y^{(i)}x^{(i)T}x$$

程序示例

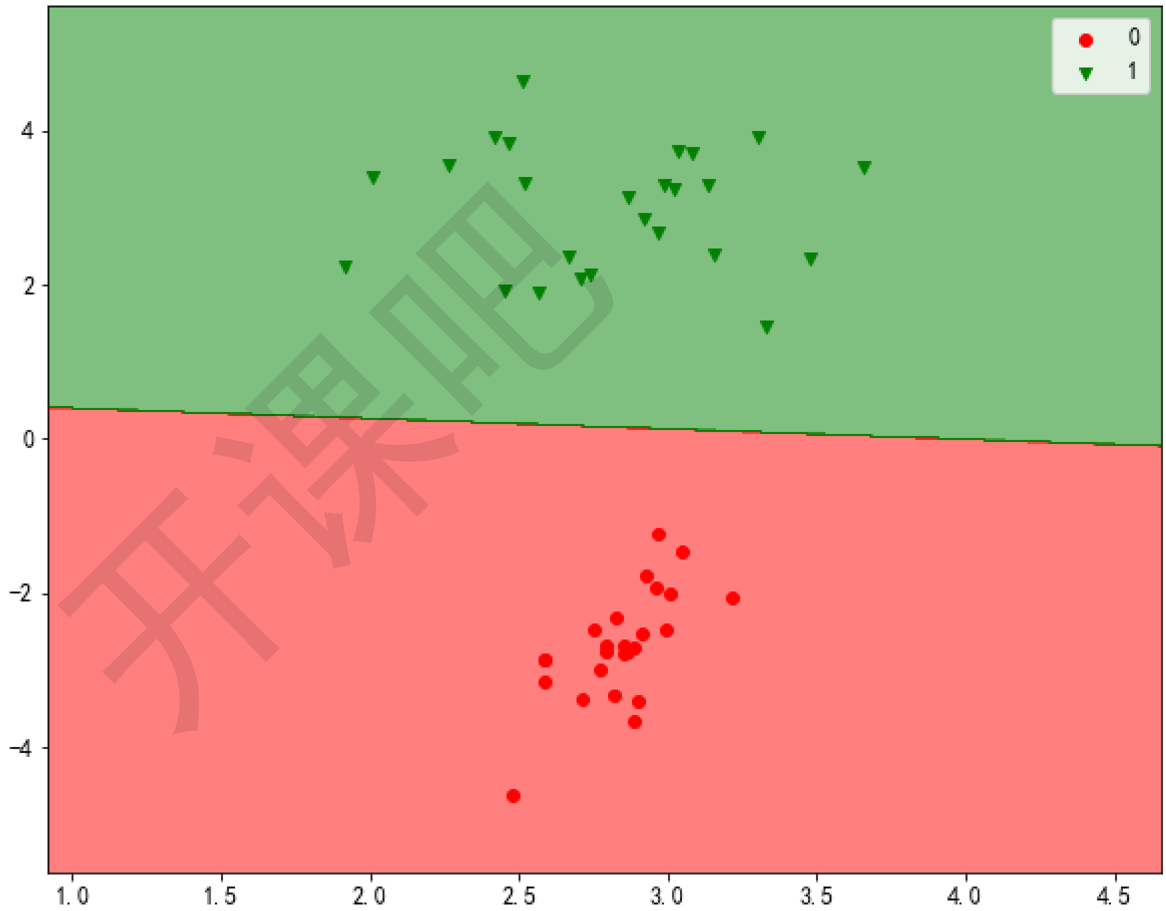
```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

plt.rcParams["font.family"] = "SimHei"
plt.rcParams["axes.unicode_minus"] = False
plt.rcParams["font.size"] = 12
```

```python
from sklearn.datasets import make_classification

X, y = make_classification(n_samples=50, n_features=2, n_redundant=0,
        n_classes=2, class_sep=2.8, n_clusters_per_class=1, flip_y=0,
        random_state=11)


plt.scatter(X[y == 0, 0], X[y == 0, 1], c="r", marker="o", label="类别0")
plt.scatter(X[y == 1, 0], X[y == 1, 1], c="g", marker="v", label="类别1")
plt.legend()
```

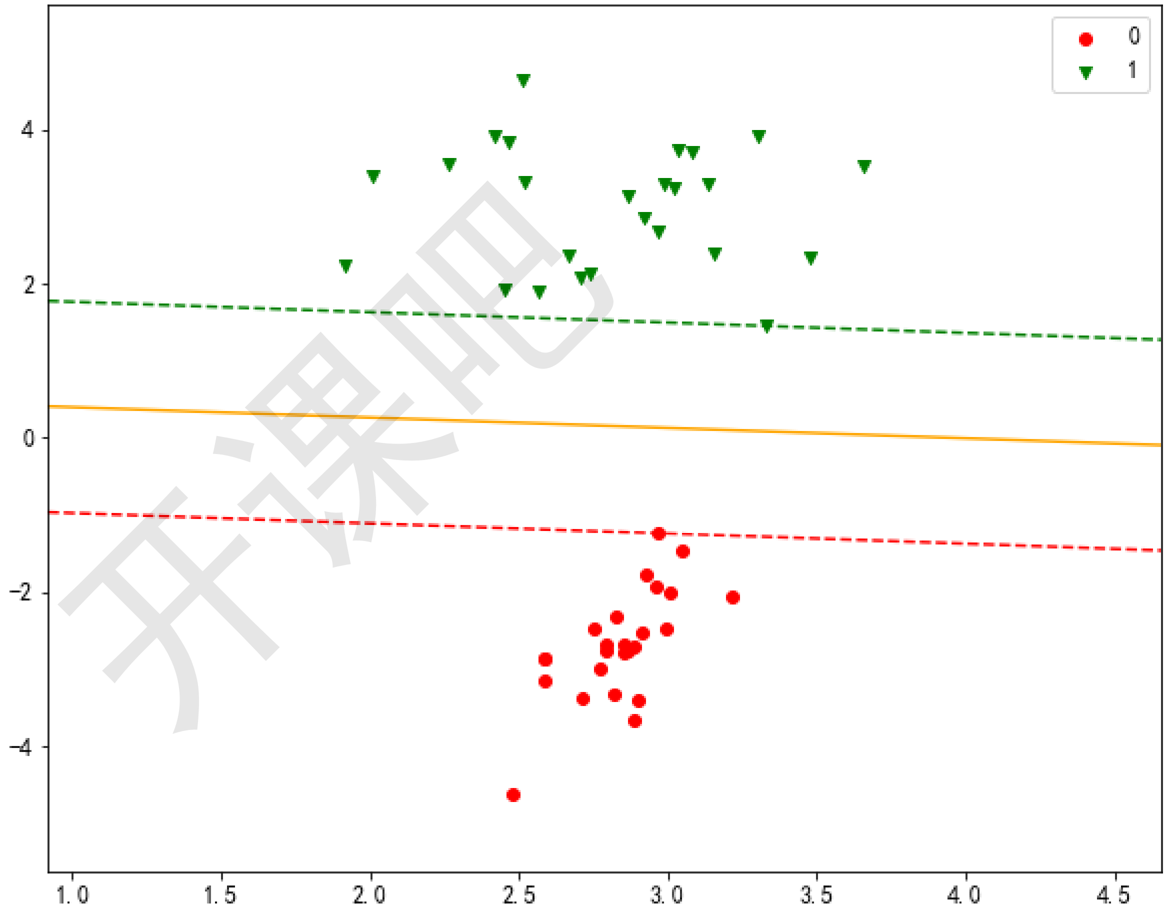
```
<matplotlib.legend.Legend at 0x200bdb00108>
```




```python
from matplotlib.colors import ListedColormap
from sklearn.svm import SVC

def plot_decision_boundary(model, X, y):
    color = ["r", "g", "b"]
    marker = ["o", "v", "x"]
    class_label = np.unique(y)
    cmap = ListedColormap(color[: len(class_label)])
    x1_min, x2_min = np.min(X, axis=0)
    x1_max, x2_max = np.max(X, axis=0)
    x1 = np.arange(x1_min - 1, x1_max + 1, 0.01)
    x2 = np.arange(x2_min - 1, x2_max + 1, 0.01)
    X1, X2 = np.meshgrid(x1, x2)
    Z = model.predict(np.c_[X1.ravel(), X2.ravel()])
    Z = Z.reshape(X1.shape)
    plt.contourf(X1, X2, Z, cmap=cmap, alpha=0.5)
    for i, class_ in enumerate(class_label):
        plt.scatter(x=X[y == class_, 0], y=X[y == class_, 1],
                    c=cmap.colors[i], label=class_, marker=marker[i])
    plt.legend()

svc = SVC(kernel="linear")
svc.fit(X, y)

plt.figure(figsize=(10, 8))
plot_decision_boundary(svc, X, y)
```




```python
def plot_decision_boundary2(model, X, y):
    color = ["r", "g", "b"]
    marker = ["o", "v", "x"]
    class_label = np.unique(y)
    cmap = ListedColormap(color[: len(class_label)])
    x1_min, x2_min = np.min(X, axis=0)
    x1_max, x2_max = np.max(X, axis=0)
    x1 = np.arange(x1_min - 1, x1_max + 1, 0.01)
    x2 = np.arange(x2_min - 1, x2_max + 1, 0.01)
    X1, X2 = np.meshgrid(x1, x2)
    v = model.decision_function(np.c_[X1.ravel(), X2.ravel()])
    v = v.reshape(X1.shape)
    plt.contour(X1, X2, v, colors=[cmap.colors[0], "orange",
cmap.colors[1]],
                levels=[-1, 0, 1], alpha=1, linestyles=["--", "-", "--"])
    for i, class_ in enumerate(class_label):
        plt.scatter(x=X[y == class_, 0], y=X[y == class_, 1],
                    c=cmap.colors[i], label=class_, marker=marker[i])
    plt.legend()


plt.figure(figsize=(10, 8))
plot_decision_boundary2(svc, X, y)
```



拓展点

- 待补充。

作业

1. 自行编写程序，来绘制出与决策边界平行的两条直线。

结束语——梁老师赋诗

海纳百川，凝聚四海的循回，（包容）
持之以恒，筑造宏伟的丰碑。（坚持）
闻鸡起舞，谱写青春的无悔，（刻苦）
同舟共济，共度岁月的相随。（团结）