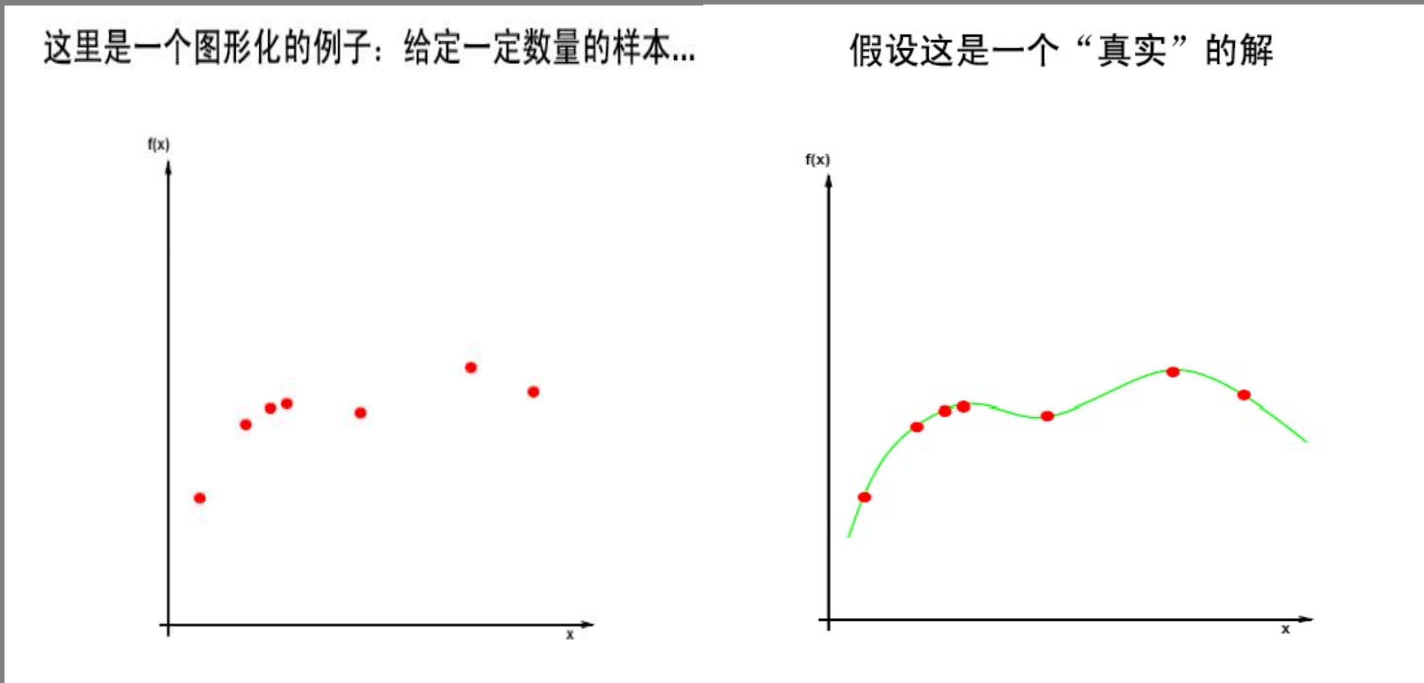
支持向量机

（Support Vector Machine）

高琰

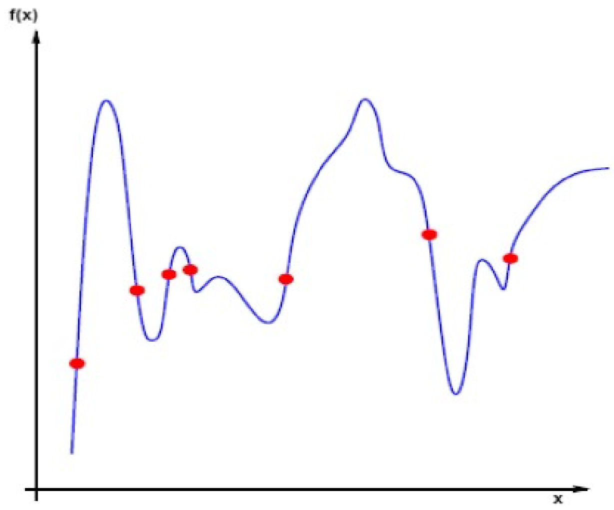
ZIXING AI
自兴人工智能

1.SVM的引入

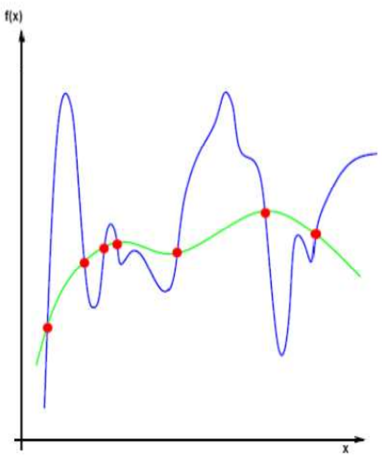


ZIXING AI
自兴人工智能

但经验风险最小化给出这样的解！



如何保证对足够数量的样本，经验风险最小化解会收敛到一个真实的解？



训练样本

过拟合，认为有锯齿的才是叶子

欠拟合，认为有绿色的都是叶子

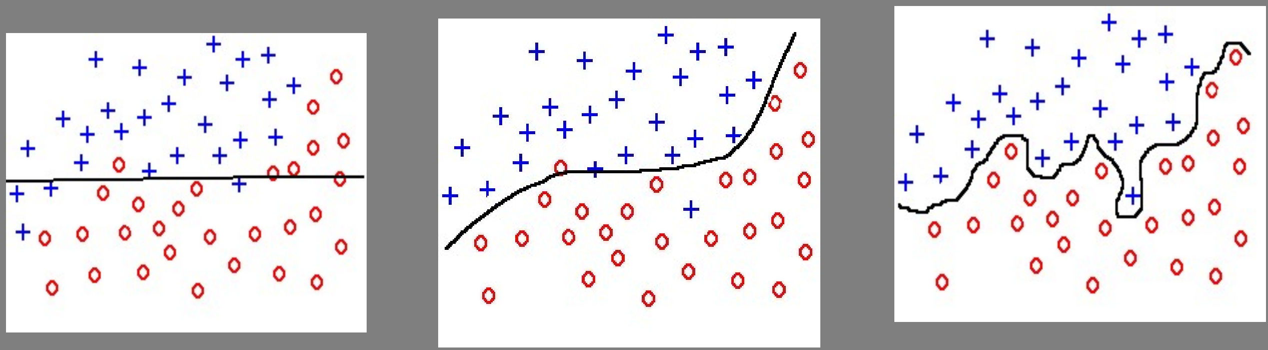
经验风险（emprical risk ）

R_{emp}：

$$L(y, f(\boldsymbol{x}, w)) = \begin{cases} 0 & y = f(\mathbf{x}, w) \\ 1 & y \neq f(\mathbf{x}, w) \end{cases}$$

过学习 Overfitting and underfitting

Problem: how rich class of classifications q(**x**;**θ**) to use.



underfitting　　good fit　　overfitting

Problem of generalization: a small emprical risk R_{emp} does not imply small true expected risk R.

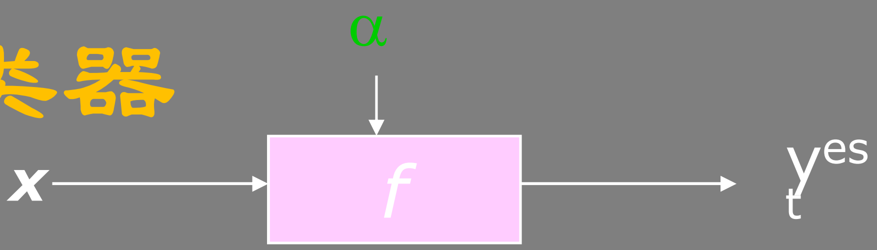
真实风险

- 学习机器的实际风险由两部分组成:
  - 训练样本的经验风险
  - 置信范围(同置信水平 $1-\eta$ 有关,而且同学习机器的VC维和训练样本数有关。

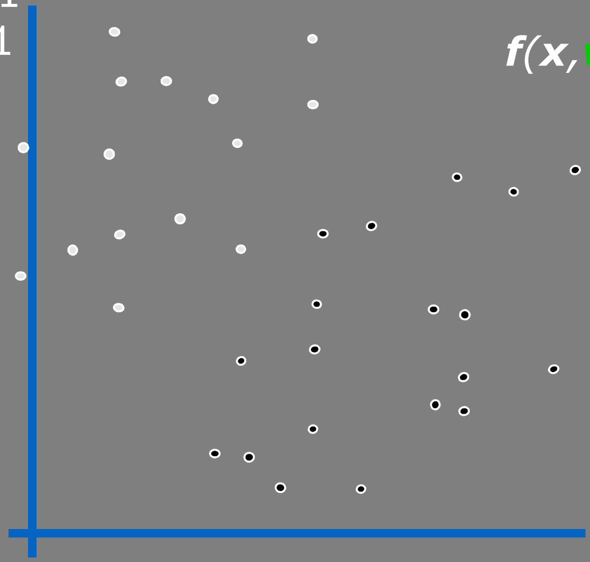
$$R(\alpha) \le R_{emp}(\alpha) + \sqrt{\frac{h(\ln(2n/h)+1)-\ln(\eta/4)}{n}}$$

$$R(\alpha) \le R_{emp}(\alpha) + \phi(h/n)$$

线性分类器



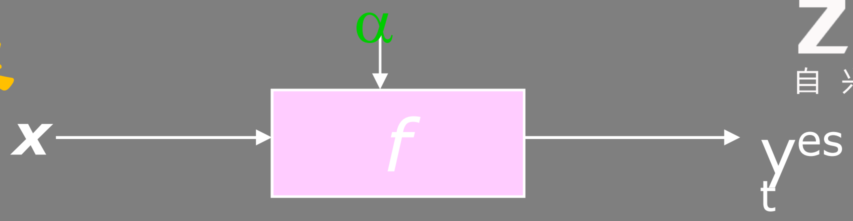
• denotes +1
• denotes -1



$f(\mathbf{x},\mathbf{w},b) = sign(\mathbf{w}.\ \mathbf{x} - b)$

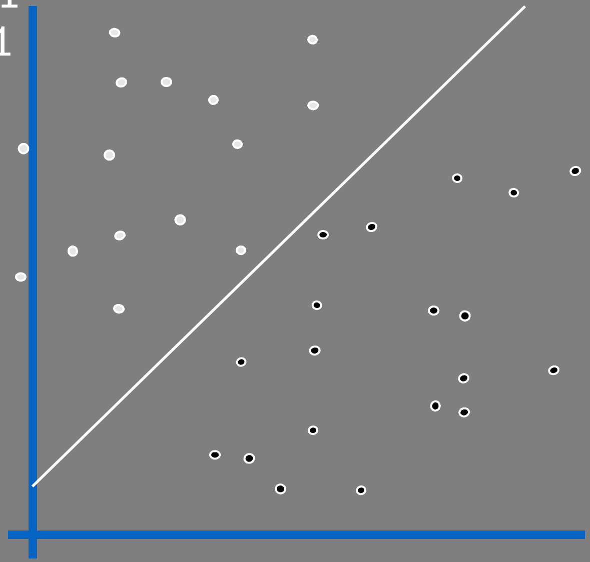
How would you classify this data?

线性分类器



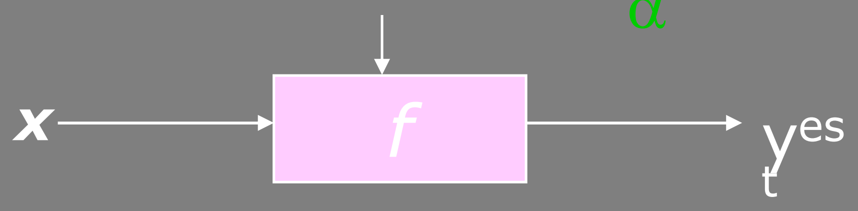
$f(\mathbf{x},\mathbf{w},b) = sign(\mathbf{w}.\ \mathbf{x} - b)$

- denotes +1
- denotes -1

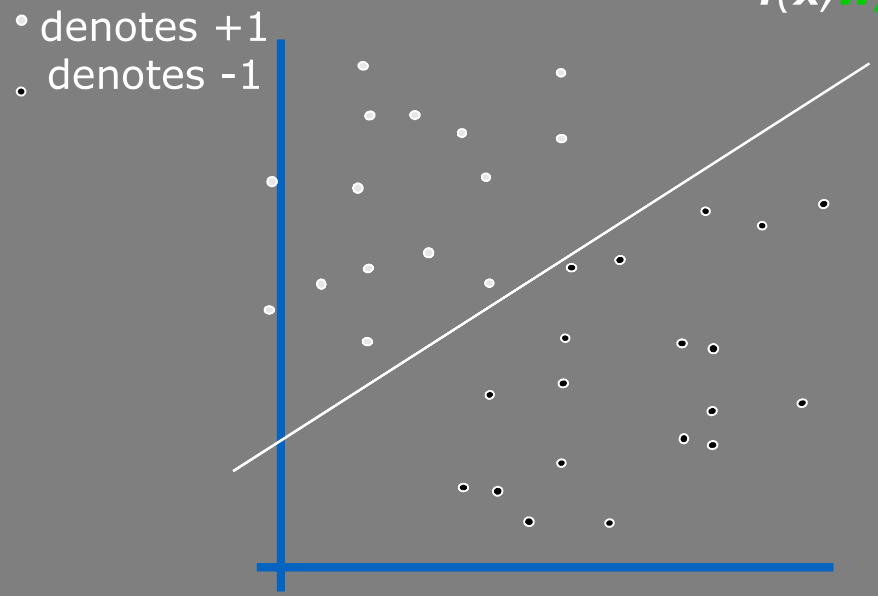


How would you classify this data?

线性分类器

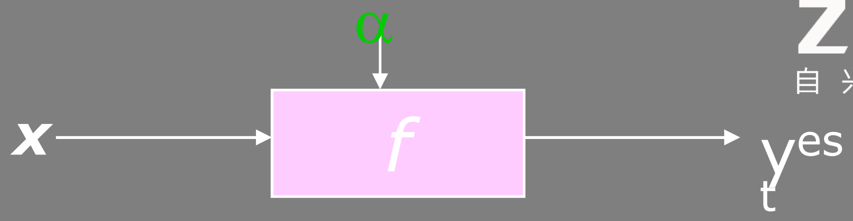


$$f(\mathbf{x},\mathbf{w},b) = sign(\mathbf{w}.\ \mathbf{x} - b)$$



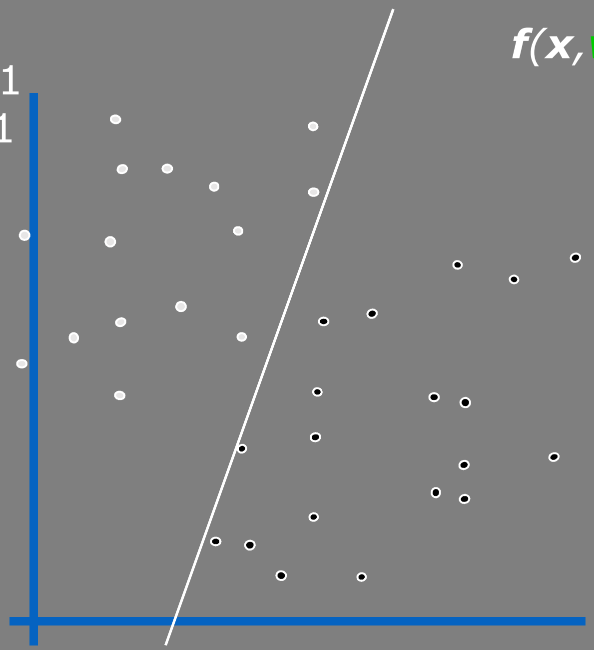
How would you classify this data?

线性分类器



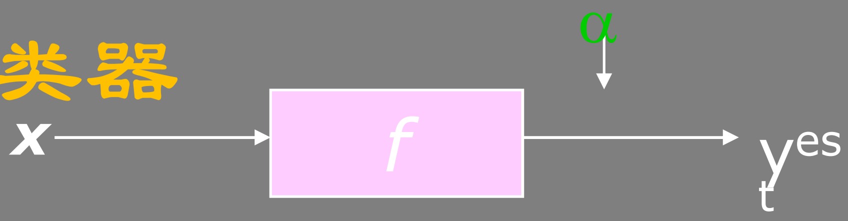
$f(\mathbf{x},\mathbf{w},b) = sign(\mathbf{w}.\ \mathbf{x} - b)$

• denotes +1
∘ denotes -1



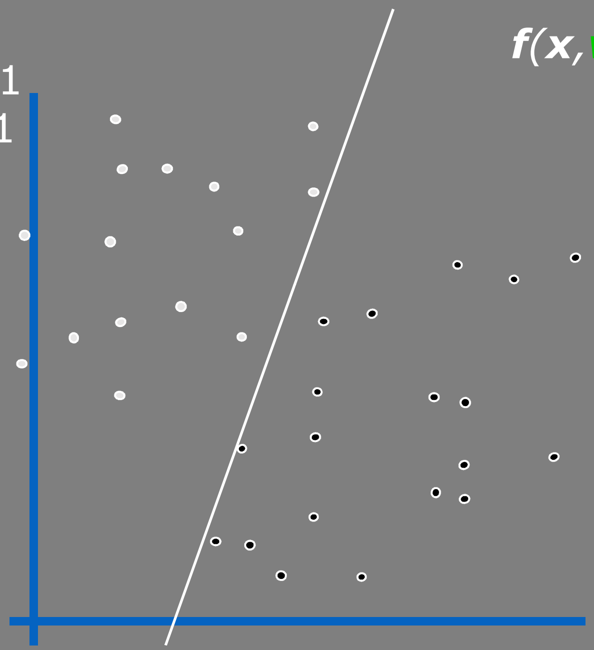
How would you classify this data?

线性分类器



$f(\mathbf{x},\mathbf{w},b) = sign(\mathbf{w}.\ \mathbf{x} - b)$

- denotes +1
- denotes -1



How would you classify this data?

2.线性可分SVM

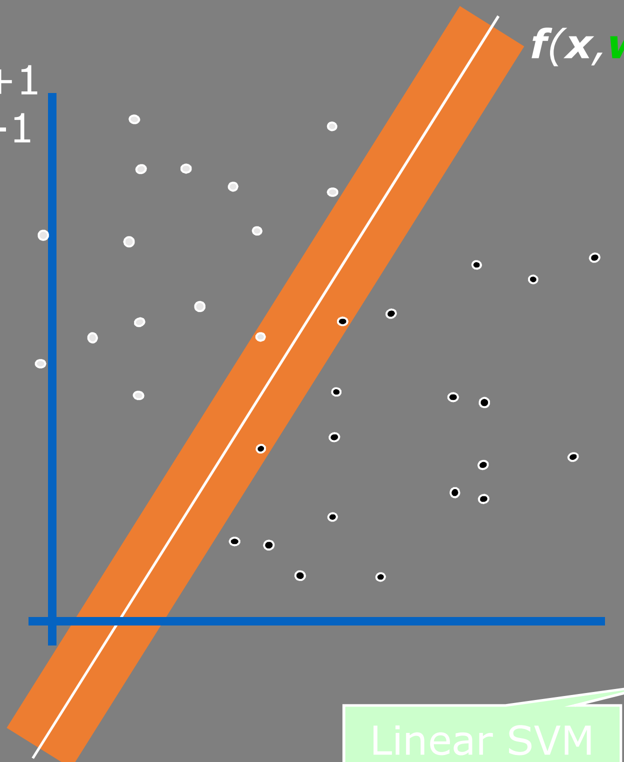
α

$\rightarrow \boxed{fx} \rightarrow y^{es}_{t}$

$f(\mathbf{x}, \mathbf{w}, b) = sign(\mathbf{w}.\ \mathbf{x} - b)$

• denotes +1

∘ denotes -1



Linear SVM

最大间隔

目的：寻找一个超平面来对样本进行分割，把样本中的正例和反例用超平面分开，但是不是简单地分看，其原则是使正例和反例之间的间隔最大。

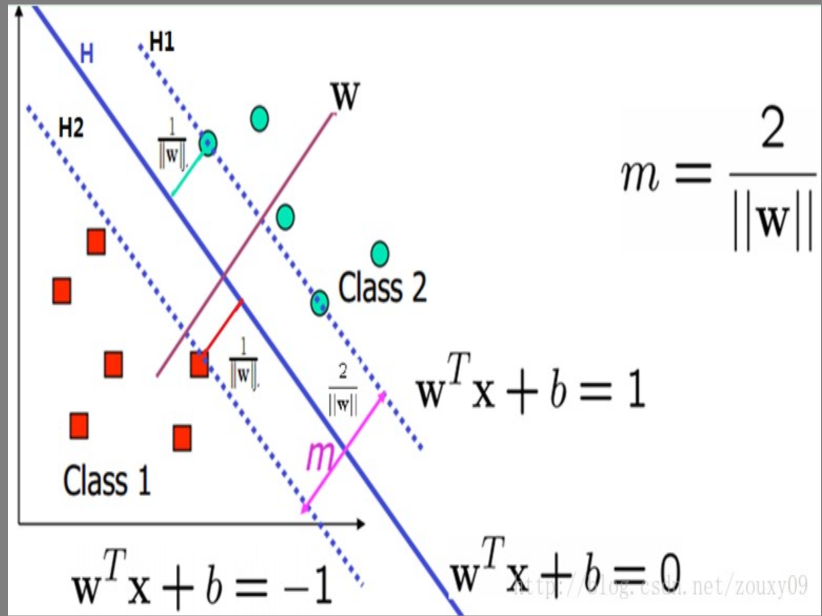
2.线性可分SVM

ZIXING AI 自兴人工智能

- 找到两个和这个超平面平行和距离相等的超平面：H_1：$y = w^T x + b = +1$ 和 H_2：$y = w^T x + b = -1$。
- 两个条件：

（1）没有任何样本在这两个平面之间；

（2）这两个平面的距离需要最大。



ZIXING AI
自兴人工智能

2.线性可分SVM

- 知识点：两条平行线的距离的求法

例如ax+by=c1和ax+by=c2，那他们的距离是|c2-c1|/sqrt(a²+b²)

- H_1和H_2的距离：|1+1|/sqrt(w²) = 2/||w||

$$y_i(w^T x_i + b) >= 1$$

- 约束条件：

任何一个正样本，它都要处于H_1的右边，也就是：$y = w^T x + b >= +1$

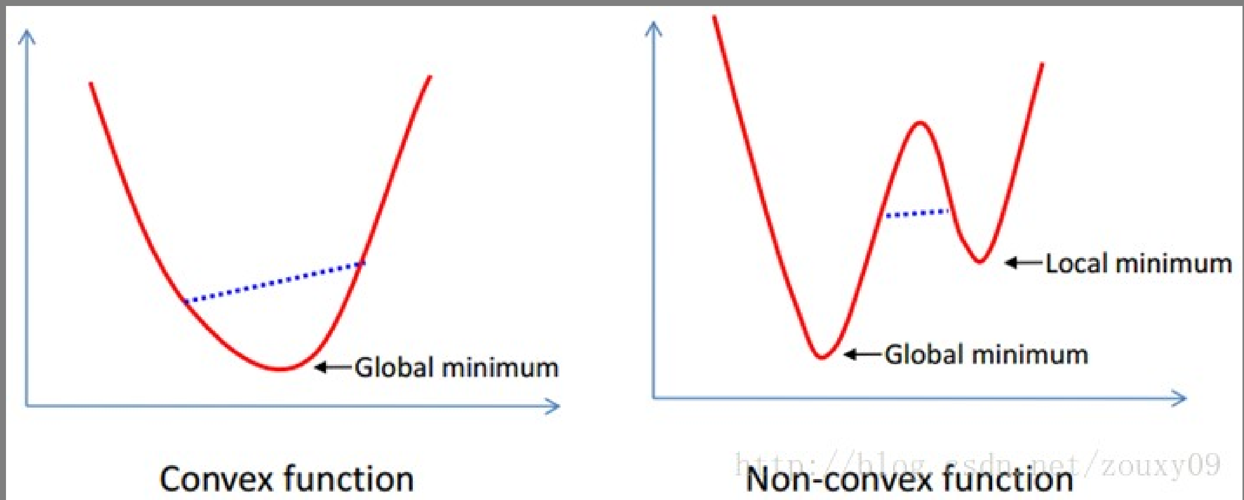
任何一个负样本，它都要处于H_2的左边，也就是：$y = w^T x + b <= -1$

线性SVM:分划直线表达式为 $(w \cdot x)+b=0$ “间隔” 为 $\frac{2}{\|w\|}$

极大化“间隔”的思想导致求解下列对变量w和 b的最优化问题

$$\min \frac{1}{2}\|w\|^2$$
$$s.t.\ y_i(w \cdot x_i + b) \geq 1, \forall x_i$$

线性硬间隔支持向量分类机



3、拉格朗日方法和对偶问题

求解问题为：

$$\min \frac{1}{2}\|w\|^2$$
$$s.t.\ y_i(w \cdot x_i + b) \geq 1, \forall x_i$$

引入拉格朗日乘子，得到以下拉格朗日函数：

$$L(w, b, \alpha) = \frac{1}{2}w^T w + \sum_{i=1}^{N} \alpha_i\left(1 - y_i(wx_i + b)\right)$$

$$\min_{\alpha} \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i\alpha_j y_i y_j(\boldsymbol{x}_i \cdot \boldsymbol{x}_j) - \sum_{i=}^{N}$$

$$s.t.\ \sum_{i=1}^{N} \alpha_i y_i = 0,$$
$$\alpha_i \geq 0, i = 1,2, \ldots N$$

3、拉格朗日方法和对偶问题

ZIXING AI 自兴人工智能

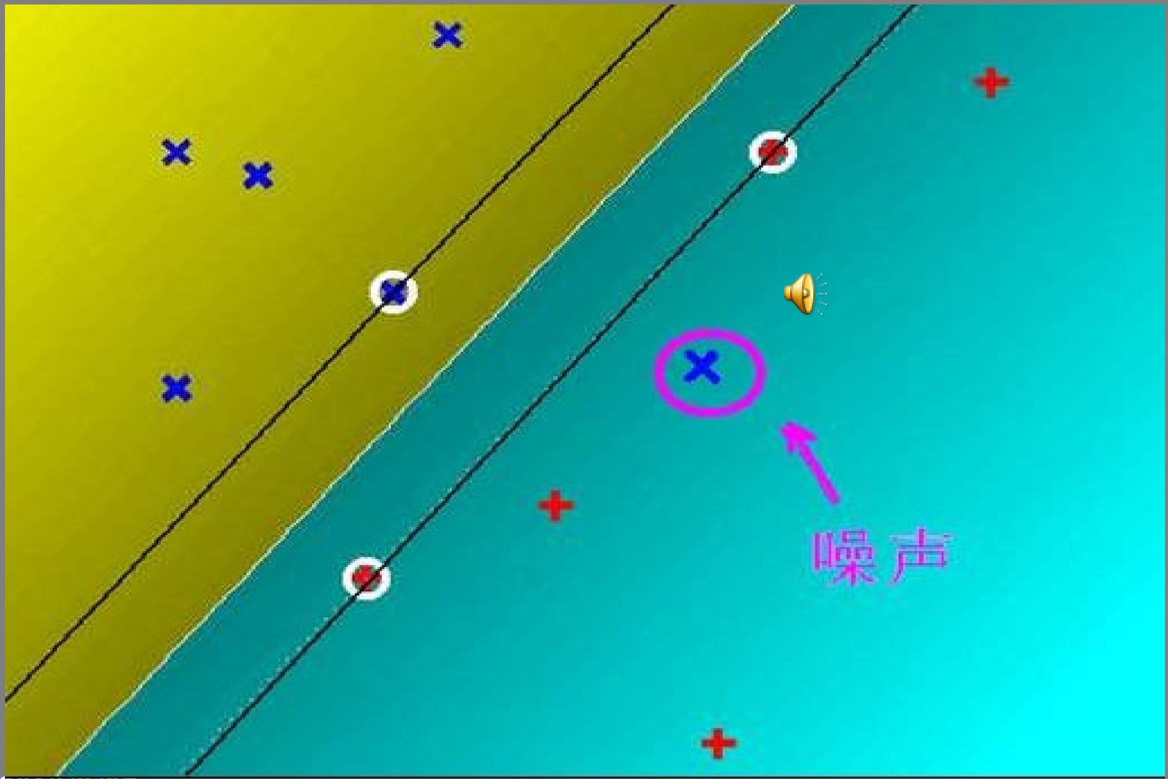
- 如果解出最优解$\alpha^*=(\alpha_1^*, \alpha_2^*, \ldots, \alpha_N^*)$，则可得到$w^*$和$b^*$ ，和相应的模型：

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

$$b^* = y_i - \sum_{i=1}^{N} \alpha_i^* y_i (\boldsymbol{x}_i \ * \boldsymbol{x}_i\ )$$

$$f(x) = w^* x + b^*$$

4.近似线性可分问题(软间隔)



近似线性可分问题

ZIXING AI 自兴人工智能

因此，引入一个惩罚参数 $C>0$ ，新的目标函数变为:

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

$$s.t \quad y_i((w\cdot x_i)+b) \ge 1-\xi_i, i=1,\ldots l$$

$$\xi_i \ge 0, i=1,\ldots l$$

$\sum_{i=1}^{l}\xi_i$ 体现了经验风险，而 $\|w\|$ 则体现了表达能力。所以惩罚参数 C 实质上是对经验风险和表达能力匹配一个裁决。当 $C\to\infty$ 时，近似线性可分SVC的原始问题退化为线性可分SVC的原始问题。

近似可分问题的对偶问题

ZIXING AI 自兴人工智能

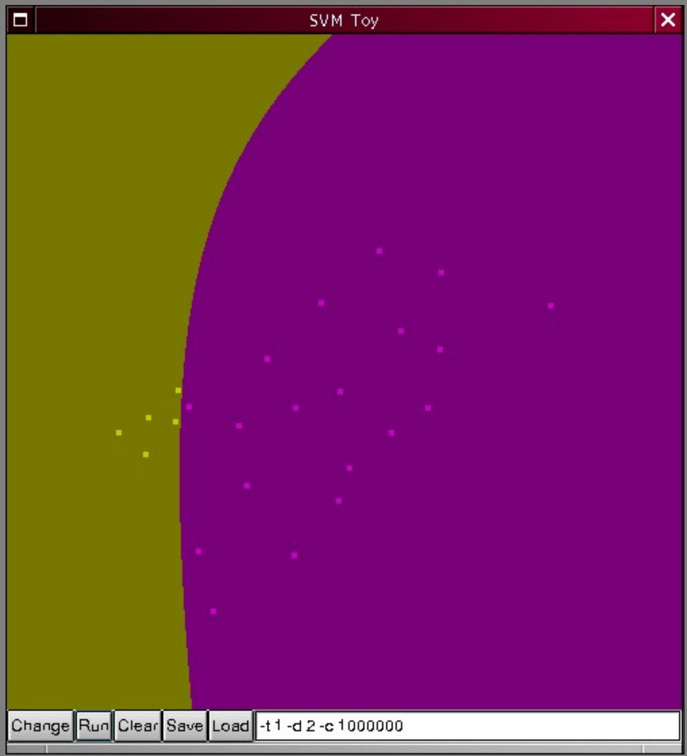
为求解原始问题，根据最优化理论，我们转化为对偶问题来求解

$$\max \; W(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{N} \alpha_i \alpha_j y_i y_j (x_i \cdot x_j)$$

$$s.t. \; \sum_{i=1}^{N} \alpha_i y_i = 0; \quad C \geq \alpha_i \geq 0, \quad i = 1, \ldots, N.$$

对偶问题

支持向量机实现



2018/3/27

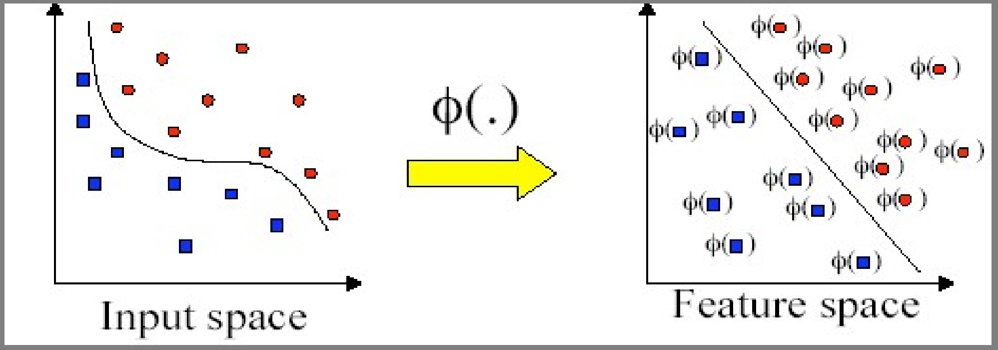
SVM的核函数

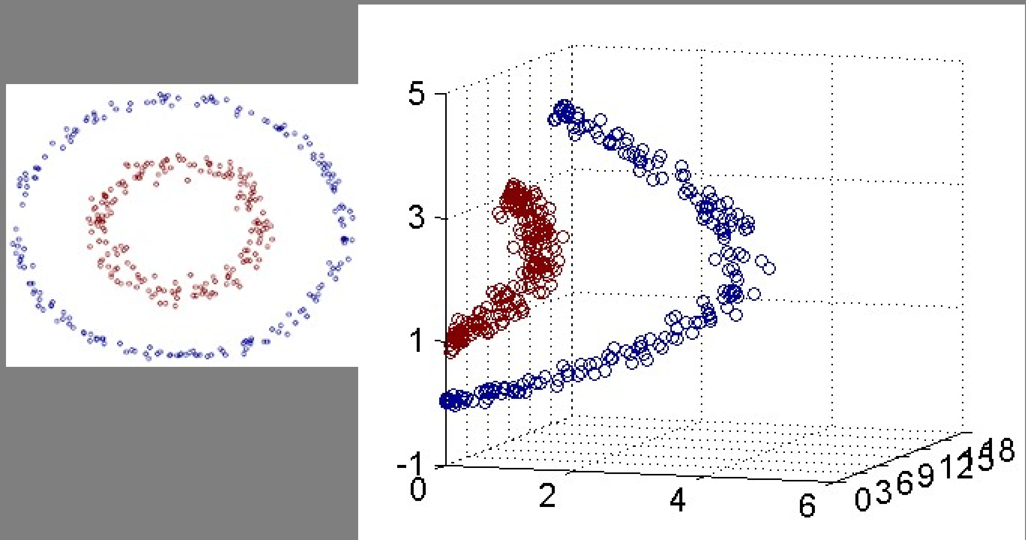
ZIXING AI 自兴人工智能

SVM 的基本思想2：

首先通过非线性变换将输入空间变换到一个高维空间；

然后在这个新空间中求取最优线性分类面，而这种非线性变换是通过定义适当的内积函数来实现的。




5
3
1
-1
0
2
4
6
0 3 6 9 12

ZIXING AI
自兴人工智能

常用的核函数

（1）多项式核函数

$$K(x,x_i)=\left[(x\cdot x_i)+1\right]^q$$

（2）径向基核函数

$$K(x,x_i)=\exp\left\{-\frac{|x-x_i|^2}{\sigma^2}\right\}$$

7、实例演示

ZIXING AI 自兴人工智能

```
import numpy as np
import matplotlib.pyplot as plt
 from sklearn import svm, datasets
 # 导入数据集
iris = datasets.load_iris()
X = iris.data[:, :2] # 只取前两维特征
 y = iris.target
h = .02 # 网格中的步长

# 创建支持向量机实例，并拟合出数据
C = 1.0 # SVM正则化参数
svc = svm.SVC(kernel='linear', C=C).fit(X, y) # 线性核
rbf_svc = svm.SVC(kernel='rbf', gamma=0.7, C=C).fit(X, y) # 径向基核
poly_svc = svm.SVC(kernel='poly', degree=3, C=C).fit(X, y) # 多项式核
lin_svc = svm.LinearSVC(C=C).fit(X, y) #线性核
```

```
for i, clf in enumerate((svc, lin_svc, rbf_svc, poly_svc)): # 绘出决策边界，不同的区域分配不同的颜色
    plt.subplot(2, 2, i + 1) # 创建一个2行2列的图，并以第i个图为当前图
    plt.subplots_adjust(wspace=0.4, hspace=0.4) # 设置子图间隔
    Z=clf.predict(np.c_[xx.ravel(), yy.ravel()]) #将xx和yy中的元素组成一对对坐标，作为支持向量机的输入，返回一个array
    # 把分类结果绘制出来
    Z = Z.reshape(xx.shape) #(220, 280)
    plt.contourf(xx, yy, Z, cmap=plt.cm.Paired, alpha=0.8) #使用等高线的函数将不同的区域绘制出来

    # 将训练数据以离散点的形式绘制出来
    plt.scatter(X[:, 0], X[:, 1], c=y, cmap=plt.cm.Paired)
    plt.xlabel('Sepal length’)
    plt.ylabel('Sepal width’)
    plt.xlim(xx.min(), xx.max())
    plt.ylim(yy.min(), yy.max())
    plt.xticks(())
```

7、实例演示

ZIXING AI
自 兴 人 工 智 能

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn import svm

# we create 40 separable points
np.random.seed(0)
X = np.r_[np.random.randn(20, 2) - [2, 2],
np.random.randn(20, 2) + [2, 2]]
Y = [0] * 20 + [1] * 20
```

7、实例演示

```
# fit the model
clf = svm.SVC(kernel='linear')
clf.fit(X, Y)

# get the separating hyperplane
w = clf.coef_[0]
a = -w[0] / w[1]
xx = np.linspace(-5, 5)
yy = a * xx - (clf.intercept_[0]) / w[1]
```

7、实例演示

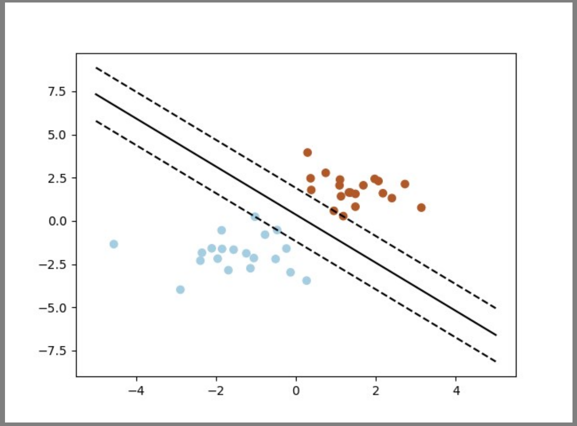
```
# plot the parallels to the separating hyperplane that pass through the support vectors
b = clf.support_vectors_[0]
yy_down = a * xx + (b[1] - a * b[0])
b = clf.support_vectors_[-1]
yy_up = a * xx + (b[1] - a * b[0])
```

7、实例演示

```
# plot the line, the points, and the nearest vectors to the plane
plt.plot(xx, yy, 'k-')
plt.plot(xx, yy_down, 'k--')
plt.plot(xx, yy_up, 'k--')

plt.scatter(clf.support_vectors_[:, 0], clf.support_vectors_[:, 1],
            s=80, facecolors='none')
plt.scatter(X[:, 0], X[:, 1], c=Y, cmap=plt.cm.Paired)

plt.axis('tight')
plt.show()
```



ZIXING AI
自 兴 人 工 智 能

练习

- 乳腺癌数据集的文件名： wdbc.data，wdbc.names，网址：http://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/
- 将数据集60%作为训练集，40%作为测试集，使用scikit-learn中SVM分类器对训练集进行训练，并利用测试数据集测试结果
- 调节svm中的参数，做出测试报告