南京大學
NANJING UNIVERSITY

人工智能导论

监督学习
(supervised learning)

郭兰哲

南京大学 智能科学与技术学院

Homepage: www.lamda.nju.edu.cn/guolz

Email: guolz@nju.edu.cn

大纲

- 近邻算法
- 决策树
- 贝叶斯分类器
- 线性回归
- 对数几率回归
- 支持向量机

线性回归

- 思考：

- 闭式解为：$\hat{W} = (X^\top X)^{-1}X^\top Y$，回想一下矩阵求逆，什么时候没有唯一解？应该怎么办？

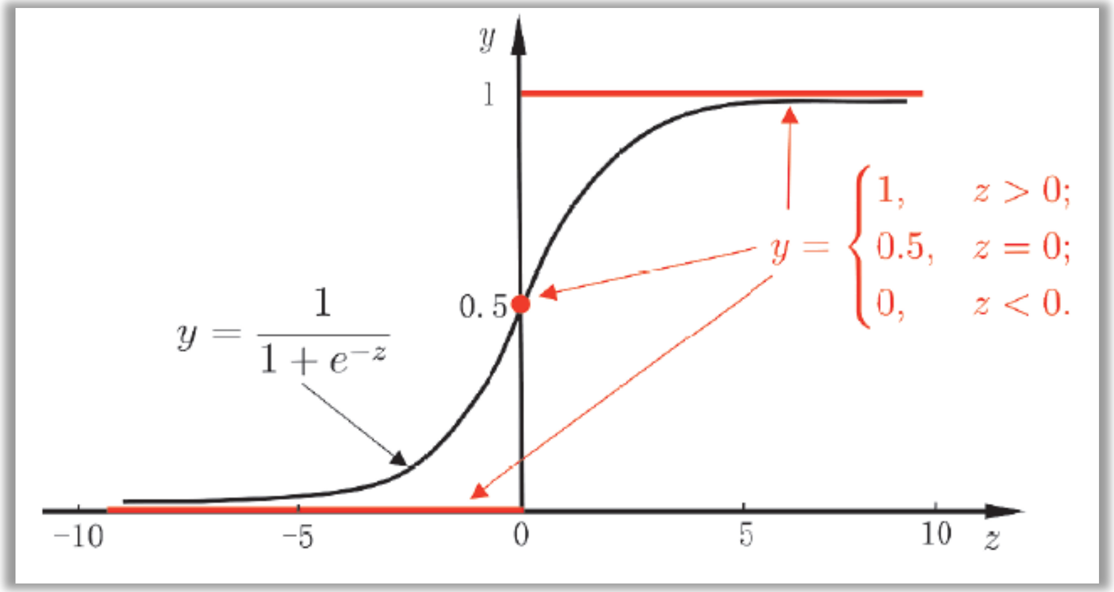
若$X^\top X$不满秩，则可解出多个$\hat{W}$

此时需求助于归纳偏好或引入正则化 (regularization)

考虑二分类任务

- 线性回归模型产生的实值输出 $z = w^{\mathrm{T}}x + b$
- 期望输出 $y \in \{0,1\}$

- 能不能找一个函数把z和y联系起来？



“单位阶跃函数”
(unit-step function)

$$y = \begin{cases} 0, z < 0 \\ 0.5, z = 0 \\ 1, z > 0 \end{cases}$$

性质不好，
需找“替代函数”
(surrogate function)

单调可微、任意阶可导
常用

$$y = \frac{1}{1 + e^{-z}}$$

Sigmoid 函数

对数几率回归(logistic regression)

- 对数几率回归(logistic regression)就是在回归模型中引入 sigmoid函数的一种模型

- Logistic回归模型可如下表示：

$$y = \frac{1}{1+e^{-z}} = \frac{1}{1+e^{-(\boldsymbol{w}^T\boldsymbol{x}+b)}}$$

其中$\frac{1}{1+e^{-z}}$是sigmoid函数、$x \in \mathbb{R}^d$是输入数据、$w \in \mathbb{R}^d$和$b \in \mathbb{R}$是待求解的模型参数

对数几率回归(logistic regression)

$$y = \frac{1}{1 + e^{-(\boldsymbol{w}^T\boldsymbol{x}+b)}} \quad \Longrightarrow \quad \ln\frac{y}{1-y} = w^{\mathrm{T}}x + b$$

“对数几率”
(log odds, 亦称 logit)

几率(odds), 反映了 $\boldsymbol{x}$ 作为正例的相对可能性

- 如果输入数据x属于正例的概率大于其属于负例的概率，即$p(y=1|x) > 0.5$，则输入数据x可被判断属于正例，这一结果等价于$\frac{p(y=1|x)}{p(y=0|x)} > 1$，即$\ln\left(\frac{p(y=1|x)}{p(y=0|x)}\right) > \ln 1 = 0$，也就是$w^T x + b > 0$成立

“对数几率回归”(logistic regression)
简称“对率回归”，跟逻辑没有关系

注意：它是
分类学习算法！

对数几率回归的求解

- 给定数据集$D=\{(x_1,y_1),(x_2,y_2),\cdots,(x_n,y_n)\}, x_i\in\mathbb{R}^d, y_i\in\{0,1\}$

- 若将y看作类后验概率估计，则有

$$\ln\frac{p(y=1|x)}{p(y=0|x)}=w^{\top}x+b$$

- 显然有，

$$p(y=1|x)=\frac{e^{w^{\top}x+b}}{1+e^{w^{\top}x+b}}$$

$$p(y=0|x)=\frac{1}{1+e^{w^{\top}x+b}}$$

对数几率回归的求解

- 我们希望每个样本属于其真实标记的概率越大越好

$$\max \sum_{i=1}^{n} \boxed{\ln p(y_i|x_i; w, b)}$$

似然项

极大似然估计
(maximum likelihood method)

- 根据

$$p(y=1|\boldsymbol{x}) = \frac{e^{w^\top x+b}}{1+e^{w^\top x+b}} = f(x)$$

$$p(y=0|\boldsymbol{x}) = \frac{1}{1+e^{w^\top x+b}} = 1-f(x)$$

- 等价于最大化 $\sum_{i=1}^{n} y_i \ln(f(x_i)) + (1-y_i)\ln(1-f(x_i))$

- 即最小化 $\mathrm{L}(f) = \sum_{i=1}^{n} -y_i \ln\big(f(x_i)\big) - (1-y_i)\ln(1-f(x_i))$

交叉熵损失

梯度下降法/牛顿法 [Boyd and Vandenberghe, 2004]

使用sklearn实现

```
class sklearn.linear_model.LogisticRegression(penalty='l2', *, dual=False, tol=0.0001, C=1.0, fit_intercept=True, intercept_scaling=1, class_weight=None, random_state=None, solver='lbfgs', max_iter=100, multi_class='auto', verbose=0, warm_start=False, n_jobs=None, l1_ratio=None)
```

```
In [19]: data
Out[19]:
```

	色泽	根蒂	敲声	纹理	脐部	触感	好瓜
0	2	2	1	1	0	0	1
1	0	2	0	1	0	0	1
2	0	2	1	1	0	0	1
3	2	2	0	1	0	0	1
4	1	2	1	1	0	0	1
5	2	1	1	1	2	1	1
6	0	1	1	2	2	1	1
7	0	1	1	1	2	0	1
8	0	1	0	2	2	0	0
9	2	0	2	1	1	1	0
10	1	0	2	0	1	0	0
11	1	2	1	0	1	1	0
12	2	1	1	2	0	0	0
13	1	1	0	2	0	0	0
14	0	1	1	1	2	1	0
15	1	2	1	0	1	0	0
16	2	2	0	2	2	0	0

```
In [6]: from sklearn.linear_model import LogisticRegression
        from sklearn.tree import DecisionTreeClassifier
        LR = LogisticRegression()
        clf = DecisionTreeClassifier()

        LR.fit(data.iloc[:,:-1],data.iloc[:,-1])
        clf.fit(data.iloc[:,:-1],data.iloc[:,-1])
```

```
In [7]: print(clf.predict([[1,1,0,1,1,0]]))
        print(LR.predict([[1,1,0,1,1,0]]))

        [1]
        [0]
```

```
In [8]: from sklearn.metrics import accuracy_score
        print("Training Accuracy of Logistic Regression: ", accuracy_score(LR.predict(data.iloc[:,:-1]),data.iloc[:,-1]))
        print("Training Accuracy of Decision Tree: ", accuracy_score(clf.predict(data.iloc[:,:-1]),data.iloc[:,-1]))

        Training Accuracy of Logistic Regression:  0.7058823529411765
        Training Accuracy of Decision Tree:  1.0
```

对数几率回归

思考：

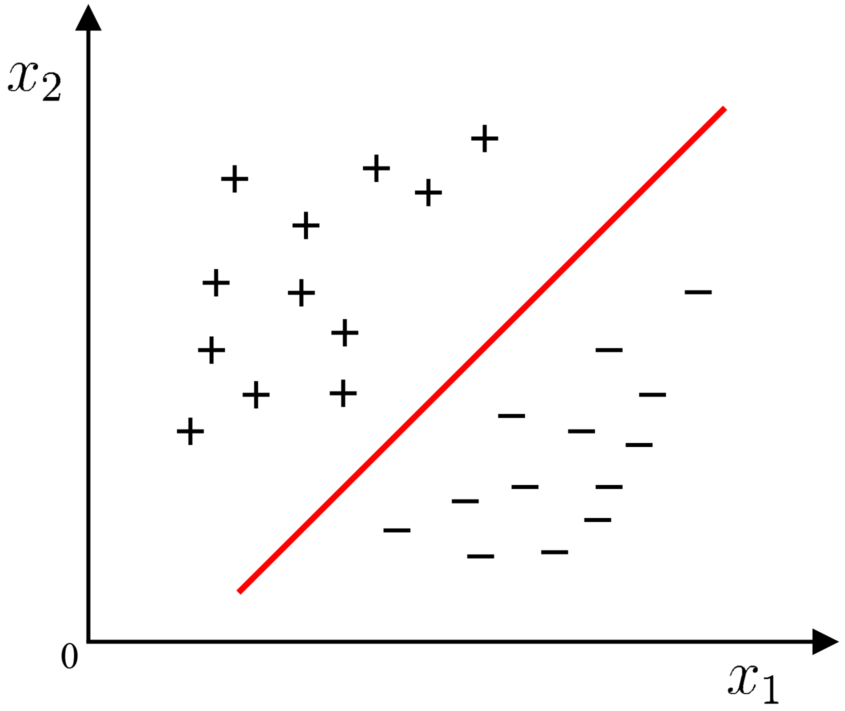
- 如何扩展logistic regression，使其能够处理多分类(multi-class)问题

大纲

- 近邻算法
- 决策树
- 贝叶斯分类器
- 线性回归
- 对数几率回归
- 支持向量机

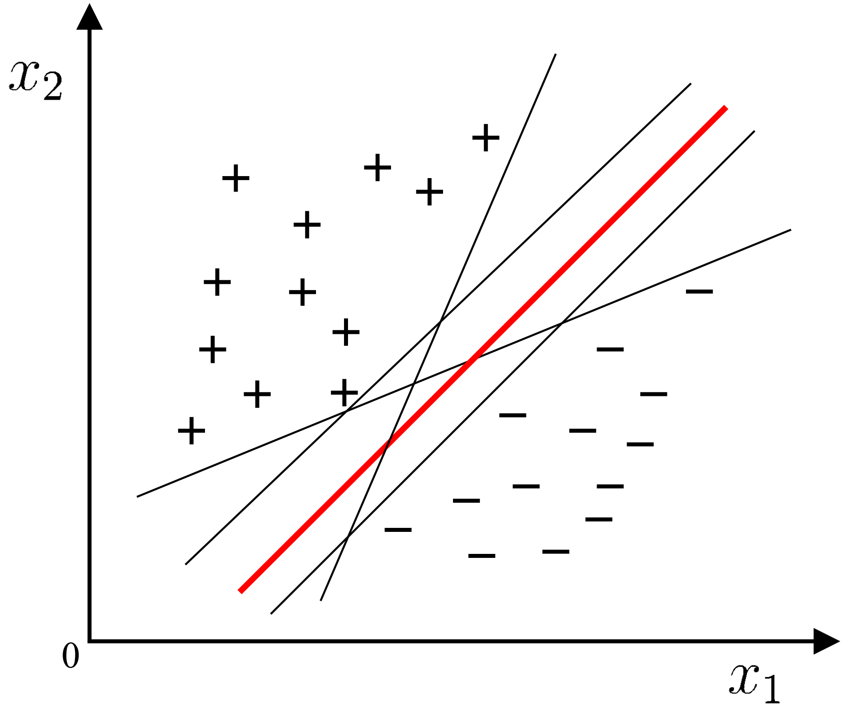
线性分类器

在样本空间中寻找一个超平面，将不同类别的样本分开



线性分类器

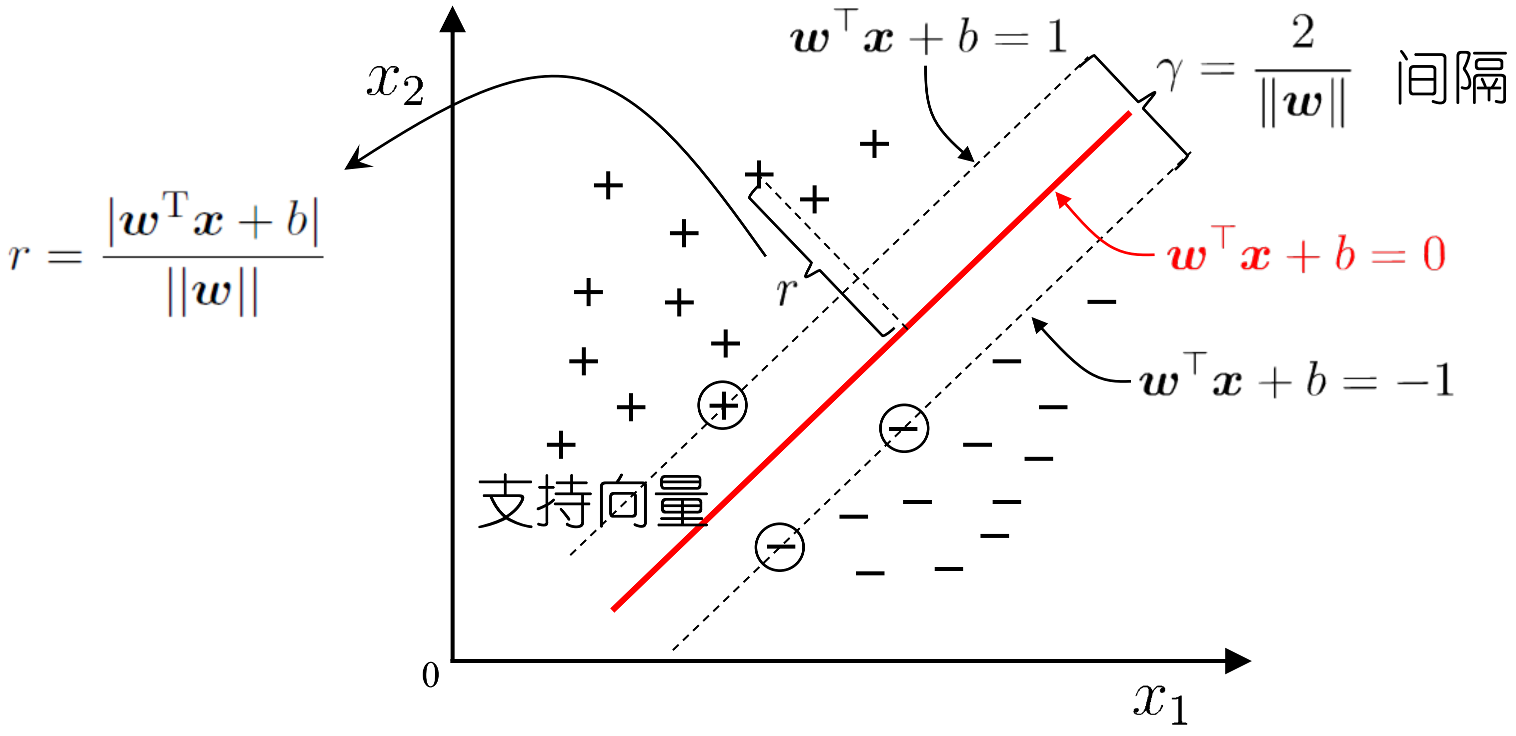
将训练样本分开的超平面可能有很多，哪一个更好呢？



"正中间"的：鲁棒性最好，泛化能力最强

间隔(margin)与支持向量(support vector)

$$\boldsymbol{w}^\top \boldsymbol{x} + b = 0$$



支持向量机-基本型

最大间隔：寻找参数 $\boldsymbol{w}$ 和 b，使得 γ 最大

$$\begin{aligned} \arg\max_{\boldsymbol{w},b} \quad & \frac{2}{\|\boldsymbol{w}\|} \\ \text{s.t.} \quad & y_i(\boldsymbol{w}^\top \boldsymbol{x}_i + b) \geq 1,\ i = 1, 2, \ldots, m. \end{aligned}$$

$$\Downarrow$$

$$\begin{aligned} \arg\min_{\boldsymbol{w},b} \quad & \frac{1}{2}\|\boldsymbol{w}\|^2 \\ \text{s.t.} \quad & y_i(\boldsymbol{w}^\top \boldsymbol{x}_i + b) \geq 1,\ i = 1, 2, \ldots, m. \end{aligned}$$

凸二次规划问题，能用优化计算包求解，但可以有更高效的办法

支持向量机-对偶型

拉格朗日乘子法

- 第一步：引入拉格朗日乘子 $\alpha_i \geq 0$ 得到拉格朗日函数

$$L(\boldsymbol{w}, b, \boldsymbol{\alpha}) = \frac{1}{2}\|\boldsymbol{w}\|^2 + \sum_{i=1}^{m} \alpha_i \left(1 - y_i(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i + b)\right)$$

- 第二步：令 $L(\boldsymbol{w}, b, \boldsymbol{\alpha})$ 对 $\boldsymbol{w}$ 和 b 的偏导为零可得

$$\boldsymbol{w} = \sum_{i=1}^{m} \alpha_i y_i \boldsymbol{x}_i \ , \quad 0 = \sum_{i=1}^{m} \alpha_i y_i$$

- 第三步：回代可得

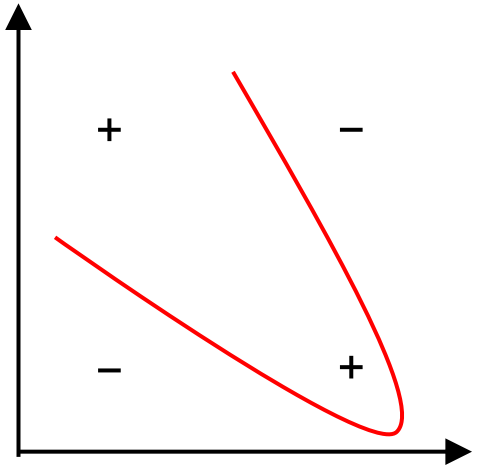
$$\max_{\boldsymbol{\alpha}} \quad \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j \boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{x}_j$$

$$\text{s.t.} \quad \sum_{i=1}^{m} \alpha_i y_i = 0 \ , \quad \alpha_i \geqslant 0 \ , \quad i = 1, 2, \ldots, m$$

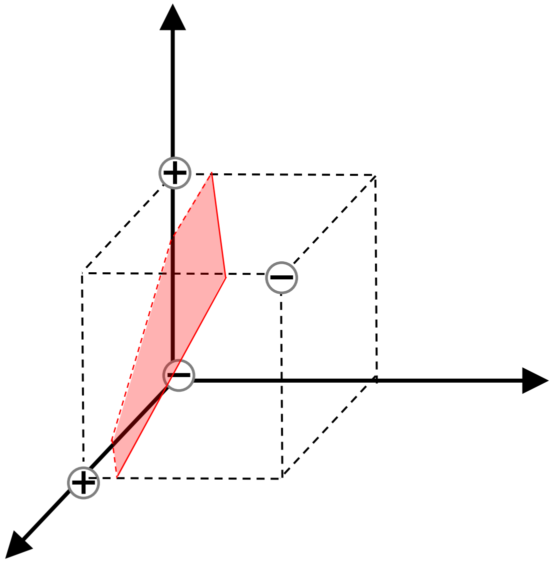
特征空间映射

若不存在一个能正确划分两类样本的超平面，怎么办?

将样本从原始空间映射到一个更高维的特征空间，使样本在这个特征空间内线性可分



$$\boldsymbol{x} \mapsto \phi(\boldsymbol{x})$$



如果原始空间是有限维(属性数有限)，那么一定存在一个高维特征空间使样本可分

特征空间映射

设样本 $\boldsymbol{x}$ 映射后的向量为 $\phi(\boldsymbol{x})$，划分超平面为 $f(\boldsymbol{x}) = \boldsymbol{w}^\top \phi(\boldsymbol{x}) + b$

原始问题

$$\min_{\boldsymbol{w},b} \quad \frac{1}{2}\|\boldsymbol{w}\|^2$$
$$\text{s.t.} \quad y_i(\boldsymbol{w}^\top \phi(\boldsymbol{x}_i) + b) \geq 1, \; i = 1, 2, \ldots, m.$$

对偶问题

$$\max_{\boldsymbol{\alpha}} \quad \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j \boxed{\phi(\boldsymbol{x}_i)^{\mathrm{T}} \phi(\boldsymbol{x}_j)}$$
$$\text{s.t.} \quad \sum_{i=1}^{m} \alpha_i y_i = 0 \,, \quad \alpha_i \geqslant 0 \,, \quad i = 1, 2, \ldots, m$$

预测

$$f(\boldsymbol{x}) = \boldsymbol{w}^{\mathrm{T}} \phi(\boldsymbol{x}) + b = \sum_{i=1}^{m} \alpha_i y_i \boxed{\phi(\boldsymbol{x}_i)^{\mathrm{T}} \phi(\boldsymbol{x})} + b$$

只以内积
形式出现

特征空间映射

基本思路：设计核函数

$$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \phi(\boldsymbol{x}_i)^{\mathrm{T}} \phi(\boldsymbol{x}_j)$$

绕过显式考虑特征映射、以及计算高维内积的困难

Mercer 定理：若一个对称函数所对应的核矩阵半正定，则它就能作为核函数来使用

任何一个核函数，都隐式地定义了一个RKHS (Reproducing Kernel Hilbert Space, 再生核希尔伯特空间)

“核函数选择”成为决定支持向量机性能的关键！

核函数

常用核函数

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	$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-\frac{\Vert\boldsymbol{x}_i - \boldsymbol{x}_j\Vert^2}{2\sigma^2}\right)$	$\sigma > 0$ 为高斯核的带宽(width)
拉普拉斯核	$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-\frac{\Vert\boldsymbol{x}_i - \boldsymbol{x}_j\Vert}{\sigma}\right)$	$\sigma > 0$
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, $\beta > 0$, $\theta < 0$

使用sklearn实现

```
class sklearn.svm.SVC(*, C=1.0, kernel='rbf', degree=3, gamma='scale'
, coef0=0.0, shrinking=True, probability=False, tol=0.001, cache_size
=200, class_weight=None, verbose=False, max_iter=-
1, decision_function_shape='ovr', break_ties=False, random_state=None
```

```
>>> import numpy as np
>>> from sklearn.pipeline import make_pipeline
>>> from sklearn.preprocessing import StandardScaler
>>> X = np.array([[-1, -1], [-2, -1], [1, 1], [2, 1]])
>>> y = np.array([1, 1, 2, 2])
>>> from sklearn.svm import SVC
>>> clf = make_pipeline(StandardScaler(), SVC(gamma='auto'))
>>> clf.fit(X, y)
Pipeline(steps=[('standardscaler', StandardScaler()),
                ('svc', SVC(gamma='auto'))])
```

```
>>> print(clf.predict([[-0.8, -1]]))
[1]
```

小结

- 常用评价指标：分类正确率，回归均方误差
- 近邻学习器：懒惰学习的代表，无需训练
- 决策树：不断选择属性划分节点建树
- 贝叶斯分类器：生成式方法的代表，估计联合分布再计算后验分布
- 线性回归：最小二乘法的闭式解
- 对数几率回归：引入sigmoid函数解决分类问题
- 支持向量机：大间隔、核函数