南京大学
NANJING UNIVERSITY

人工智能导论

统计机器学习算法
(Statistical Learning)

郭兰哲

南京大学 智能科学与技术学院

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

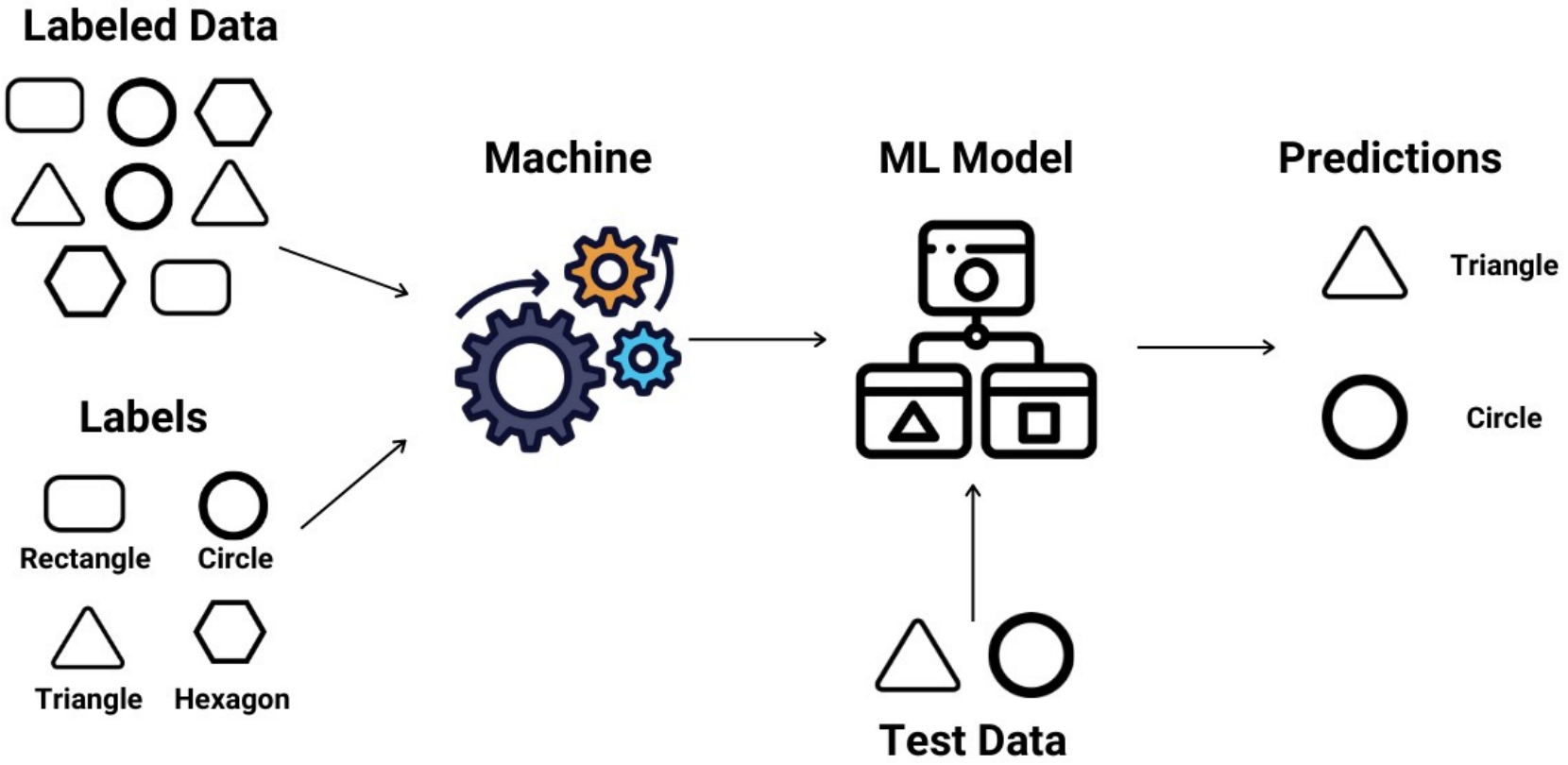
Email: guolz@nju.edu.cn

大纲

- 监督学习(Supervised Learning)
  - 近邻算法
  - 线性回归
  - 对数几率回归
- 无监督学习(Unsupervised Learning

监督学习(Supervised Learning)

监督学习：所有训练样本均有对应的标注



考虑一个二分类问题

- 给定训练数据集 $D = \{(x_1, y_1), (x_2, y_2), \cdots, (x_n, y_n)\}$
- 学习模型 $f: \mathcal{X} \to \mathcal{Y}$, $\mathcal{Y} = \{-1, +1\}$

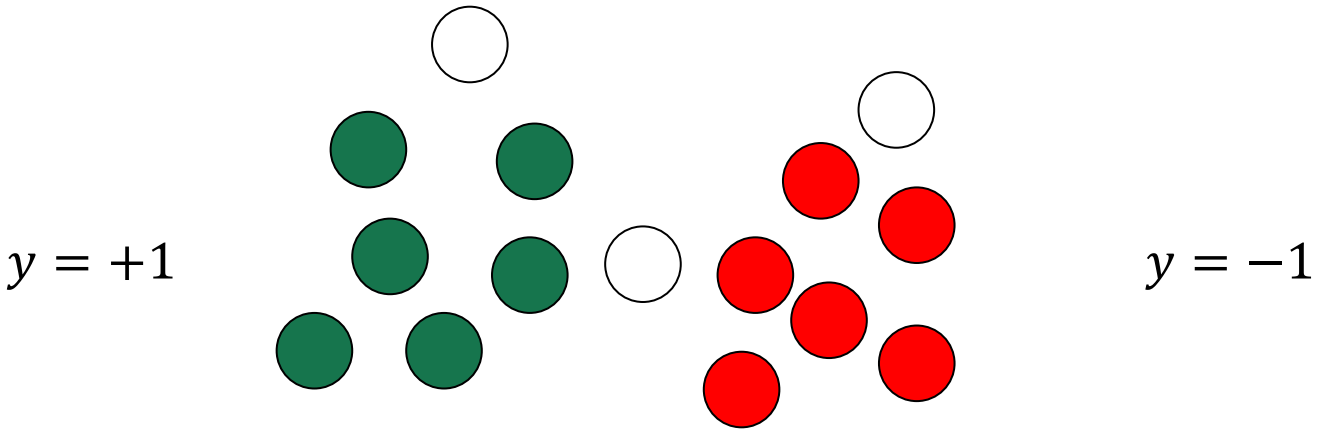
基本思路：

近朱者赤，近墨者黑

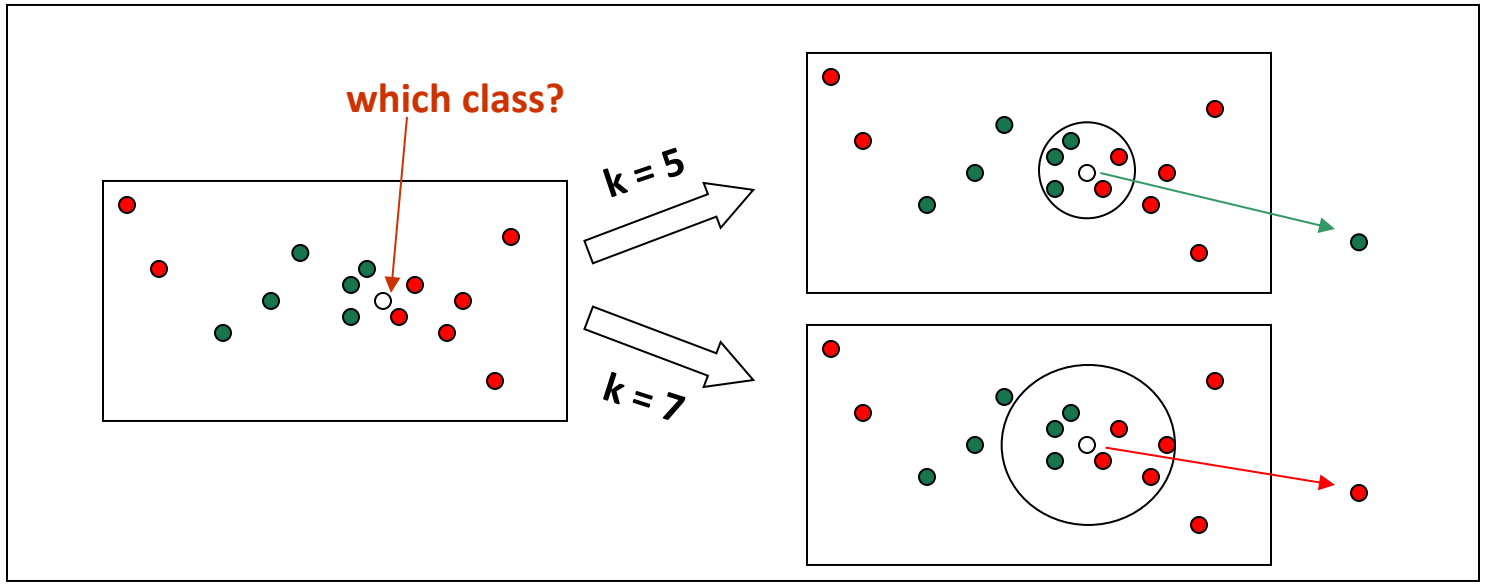
绿色代表正类训练样本，红色代表负类训练样本

懒惰学习(lazy learning)的代表

白色测试样本应该被预测为哪个类别？



k 近邻学习器(K Nearest Neighbors)

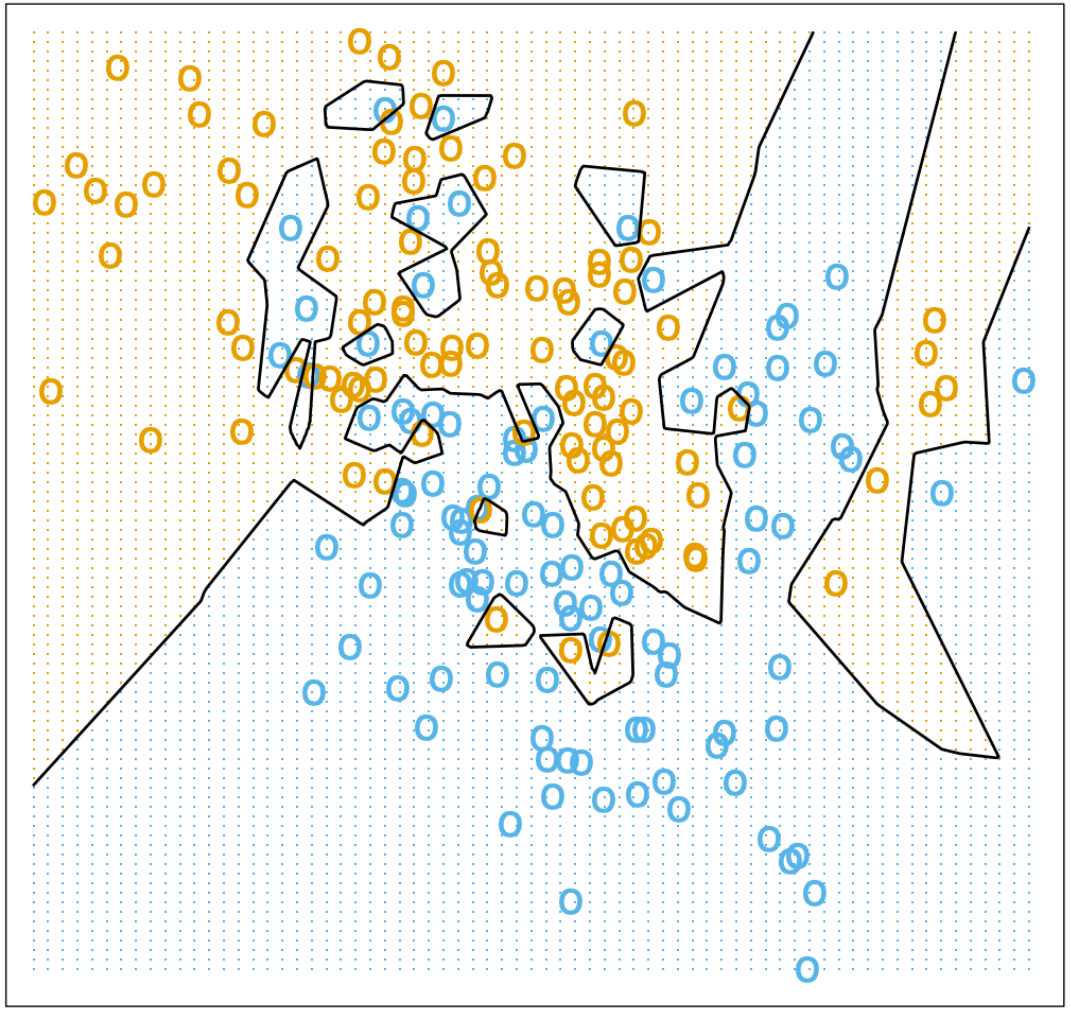


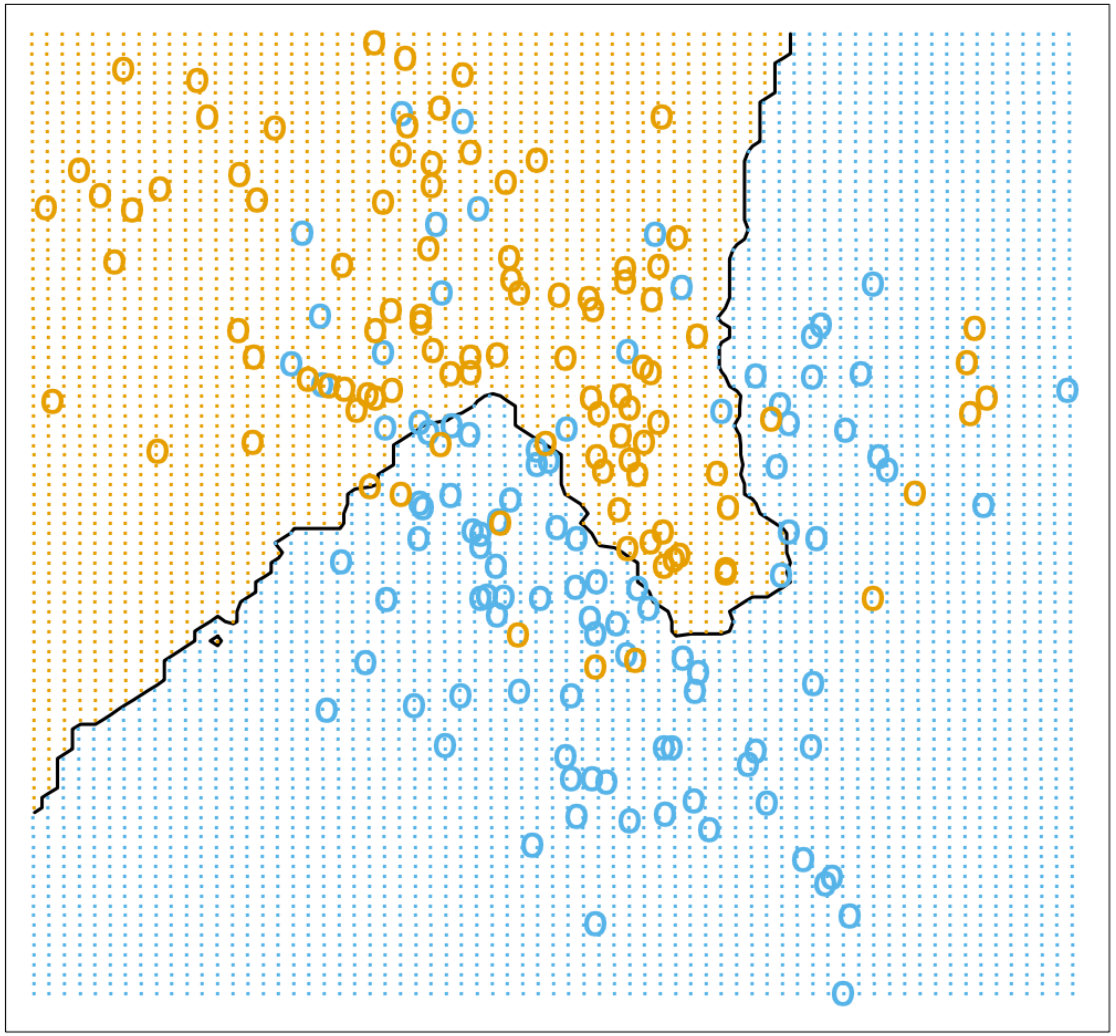
找到最近的k个样本，投票 或者 平均

关键问题： k值选取；距离计算

k值的选择

下图分别是K=1和K=15的情况，哪个是K更大的分类器？





k值的选择

k 值的选择会对k近邻学习器的结果产生重要影响

- 如果选择较小的 k，相当于用较小的邻域中的训练样本进行预测，训练误差会减小，泛化误差会增大，即k 的减小意味着整体模型变得更复杂，容易发生过拟合
- 如果选择较大的 k，相当于用较大的邻域中的训练样本进行预测，会让模型变得简单，一定程度上可以减小泛化误差
- 如果k=N(训练样本总数量）？
  - 无论输入样本是什么，都将预测它属于在训练样本中最多的类，模型过于简单
- 在应用中，k值一般取一个较小的数值，然后通过对性能进行评估来选取最优的k值

距离度量

距离度量 (distance metric) 需满足的基本性质：

非负性: $\mathrm{dist}(\boldsymbol{x}_i, \boldsymbol{x}_j) \geqslant 0$;

同一性: $\mathrm{dist}(\boldsymbol{x}_i, \boldsymbol{x}_j) = 0$ 当且仅当 $\boldsymbol{x}_i = \boldsymbol{x}_j$;

对称性: $\mathrm{dist}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \mathrm{dist}(\boldsymbol{x}_j, \boldsymbol{x}_i)$;

直递性: $\mathrm{dist}(\boldsymbol{x}_i, \boldsymbol{x}_j) \leqslant \mathrm{dist}(\boldsymbol{x}_i, \boldsymbol{x}_k) + \mathrm{dist}(\boldsymbol{x}_k, \boldsymbol{x}_j)$.

常用距离形式：闵可夫斯基距离 (Minkowski distance)

假设输入样本为d维向量：$x = (x_1, x_2, \cdots, x_d)$

$$dist_{mk}(x, x') = \left(\sum_{i=1}^{d} \left| x_i - x_i' \right|^p \right)^{\frac{1}{p}}$$

p = 2: 欧氏距离(Euclidean distance)

p = 1: 曼哈顿距离(Manhattan distance)

k近邻学习器

思考：

- 如何计算离散属性之间的距离？
- 有序属性和无序属性的区别？

色泽	根蒂	敲声
青绿	蜷缩	浊响
乌黑	蜷缩	浊响
青绿	硬挺	清脆
乌黑	稍蜷	沉闷

k近邻学习器

思考：

- 测试时的复杂度是多少？能否优化？
- 距离度量会严重影响近邻算法的性能，有没有可能针对具体的任务学习出来一个最适合的距离度量指标？

使用Sklearn实现

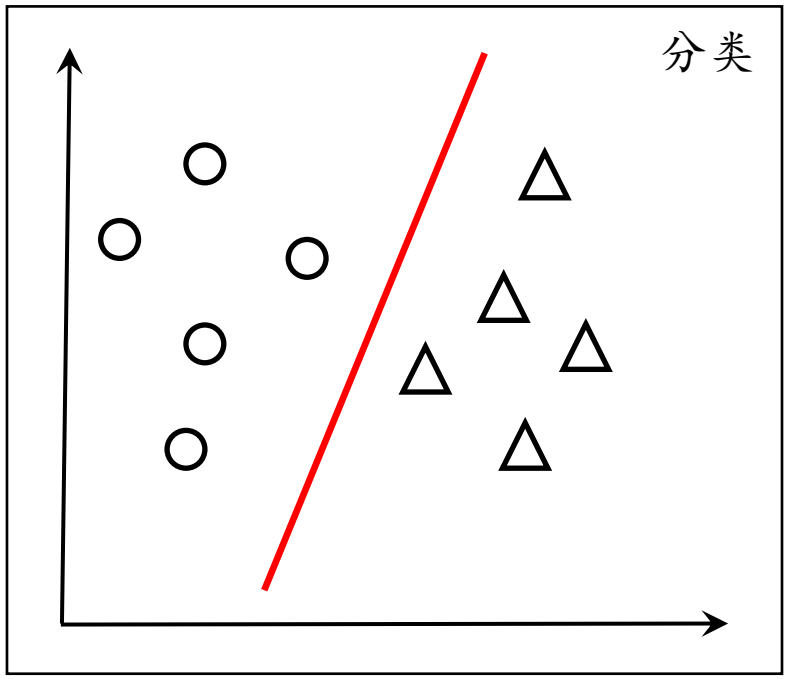
```
class sklearn.neighbors.KNeighborsClassifier(n_neighbors=5, *, weights='uniform', algorithm='
auto', leaf_size=30, p=2, metric='minkowski', metric_params=None, n_jobs=None)
```

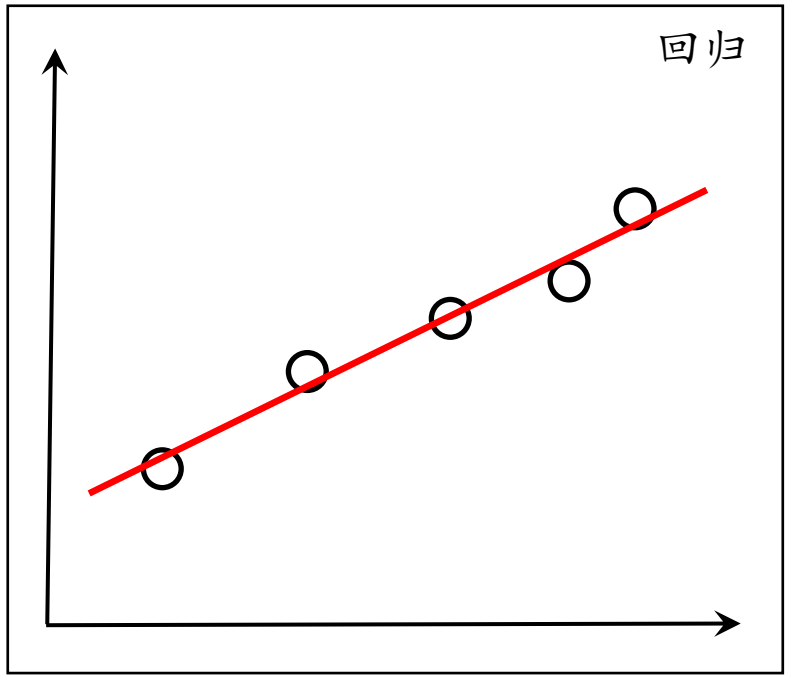
```
>>> X = [[0], [1], [2], [3]]
>>> y = [0, 0, 1, 1]
>>> from sklearn.neighbors import KNeighborsClassifier
>>> neigh = KNeighborsClassifier(n_neighbors=3)
>>> neigh.fit(X, y)
>>> print(neigh.predict([[1.1]]))
[0]
```

大纲

- 近邻算法
- 线性回归
- 对数几率回归

线性模型





线性回归(linear regression)

- 在现实生活中，往往需要分析若干变量之间的关系，如碳排放量与气候变暖之间的关系、某一商品广告投入量与该商品销售量之间的关系等

- 这种分析不同变量之间存在关系的研究叫回归分析，刻画不同变量之间关系的模型被称为回归模型，如果这个模型是线性的，则称为线性回归模型

$$y = 33.73(英寸) + 0.516x$$

y: 子女平均身高　　x:父母平均身高

- 父母平均身高每增加一个单位，其成年子女平均身高只增加0.516个单位，它反映了这种“衰退(regression)”效应（“回归”到正常人平均身高）

- 虽然x和y之间并不总是具有“衰退”（回归）关系，但是“线性回归”这一名称就保留了下来了

一元线性回归(linear regression)

- 例子：下表给出了某地区发生森林火灾的部分历史数据，表中列举了每次发生森林火灾时的气温温度取值x和受到火灾影响的森林面积y

气温温度x	5.1	8.2	11.5	13.9	15.1	16.2	19.6	23.3
火灾影响面积y	2.14	4.62	8.24	11.24	13.99	16.33	19.23	28.74

如何对气温温度与火灾所影响的森林面积之间关系进行建模？

$$f(x_i) = wx_i + b \text{ 使得 } f(x_i) \cong y_i$$

- 需要学习的参数：w, b
- 机器学习的任务：基于训练数据确定w, b的取值

一元线性回归(linear regression)

模型：$f(x_i) = wx_i + b$，目标：$f(x_i) \cong y_i$

- 模型在每个样本x_i的预测值$f(x_i)$与真实标注（实际值）y_i之差记为

$$(f(x_i) - y_i)^2$$

- 训练集中n个样本所产生误差总和为：

$$L(f) = \sum_{i=1}^{n} (y_i - f(\mathrm{x_i}))^2$$

均方误差(mean squared error)

- 目标：$(\mathrm{w}^*, \mathrm{b}^*) = \arg\min_{(w,b)} \sum_{i=1}^{n} (y_i - f(\mathrm{x_i}))^2$

最小二乘法(least square method)

一元线性回归(linear regression)

$$(\mathrm{w}^*, \mathrm{b}^*) = \arg\min_{(w,b)} \sum_{i=1}^{n} (y_i - f(\mathrm{x_i}))^2$$

$$= \arg\min_{(w,b)} \sum_{i=1}^{n} (y_i - \mathrm{w}x_i - b)^2$$

将$L(w,b)$分别对w和b求导：

$$\frac{\partial L(w,b)}{\partial w} = \sum_{i=1}^{n} 2(y_i - wx_i - b)(-x_i) = 2\left(w\sum_{i=1}^{n} x_i^2 - \sum_{i=1}^{n}(y_i - b)x_i\right)$$

$$\frac{\partial L(w,b)}{\partial b} = \sum_{i=1}^{n} 2(y_i - wx_i - b)(-1) = 2\left(nb - \sum_{i=1}^{n}(y_i - wx_i)\right)$$

一元线性回归(linear regression)

令导数为0，得到闭式(closed-form)解：

$$b = \frac{1}{n}\sum_{i=1}^{n}(y_i - wx_i) = \bar{y} - w\bar{x} \quad (\bar{x}, \bar{y}\text{表示均值})$$

$$w = \frac{\sum_{i=1}^{n} x_i y_i - n\bar{x}\,\bar{y}}{\sum_{i=1}^{n} x_i x_i - n\bar{x}^2}$$

$$\frac{\partial L(w,b)}{\partial w} = 2\left(w\sum_{i=1}^{n} x_i^2 - \sum_{i=1}^{n}(y_i - b)x_i\right)$$

$$\frac{\partial L(w,b)}{\partial b} = 2\left(nb - \sum_{i=1}^{n}(y_i - wx_i)\right)$$

一元线性回归(linear regression)

- 例子：下表给出了某地区发生森林火灾的部分历史数据，表中列举了每次发生森林火灾时的气温温度取值x和受到火灾影响的森林面积y

气温温度x	5.1	8.2	11.5	13.9	15.1	16.2	19.6	23.3
火灾影响面积y	2.14	4.62	8.24	11.24	13.99	16.33	19.23	28.74

$$b = \frac{1}{n}\sum_{i=1}^{n}(y_i - wx_i) = \bar{y} - w\bar{x} \qquad w = \frac{\sum_{i=1}^{n} x_i y_i - n\bar{x}\,\bar{y}}{\sum_{i=1}^{n} x_i x_i - n\bar{x}^2}$$

基于训练样本可得：$w = \frac{x_1y_1+x_2y_2+\cdots+x_8y_8-8\bar{x}\bar{y}}{x_1^2+x_2^2+\cdots+x_8^2-8\bar{x}^2} = 1.428, b = \bar{y} - a\bar{x} = -7.09$

即预测芒提兹尼欧地区火灾所影响森林面积与气温温度之间的一元线性回归模型为“火灾所影响的森林面积$= 1.428\times$气温温度$- 7.09$”，即$y = 1.428x - 7.09$

多元线性回归(linear regression)

- 例子：接下来扩展到数据特征的维度是多维的情况，在上述数据中增加一个影响火灾影响面积的潜在因素—风力

气温x	5.1	8.2	11.5	13.9	15.1	16.2	19.6	23.3
风力z	4.5	5.8	4	6.3	4	7.2	6.3	8.5
火灾影响面积y	2.14	4.62	8.24	11.24	13.99	16.33	19.23	28.74

假设总共有n个训练样本$\{(x_i, y_i)\}_{i=1}^{n}$，其中$x_i = [x_{i,1}, x_{i,2}, \dots, x_{i,d}] \in \mathbb{R}^d$，$d$为数据特征的维度

线性模型(linear model)试图学得一个通过属性的线性组合来进行预测的函数

$$f(x_i) = w_1 x_{i1} + w_2 x_{i2} + \cdots + w_d x_{id} + b$$

向量形式：$f(x) = w^{\mathsf{T}} x + b$

多元线性回归-矩阵形式

- 矩阵形式：给定数据集$D=\{X,Y\}, X\in\mathbb{R}^{n\times d}, Y\in\mathbb{R}^{n\times 1}$

$$X=\begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1d} \\ x_{21} & x_{22} & \cdots & x_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nd} \end{pmatrix} \qquad \mathrm{Y}=\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

- $f(X)=XW+b$

其中$W=(w_1;w_2;\ldots;w_d)\in\mathbb{R}^{d\times 1}$

$$f(X)=XW+b \text{ 使得 } f(X)\cong Y$$

多元线性回归-齐次表达

- 把W和b吸收入矩阵$\hat{W}$

$$\hat{W} = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_d \\ b \end{bmatrix}$$

- 在特征矩阵后添加一列，元素全部置为1

$$X = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1d} & 1 \\ x_{21} & x_{22} & \cdots & x_{2d} & 1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nd} & 1 \end{pmatrix}$$

- 可得：

$$f(X) = X\hat{W}$$

多元线性回归

- 同样采用最小二乘法求解，有

$$\widehat{W} = \arg\min_{\widehat{W}} \left|\left|X\widehat{W} - Y\right|\right|^2$$

- 对$\widehat{W}$求导可得：

$$\frac{\partial L(\widehat{W})}{\partial \widehat{W}} = 2X^{\mathrm{T}}(X\widehat{W} - Y)$$

- 令导数为0可得：

$$\widehat{W} = (X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}}Y$$

均方误差函数是一个二次的凸函数，所以函数只存在一个极小值点，也同样是最小值点

多元线性回归

- 例子：接下来扩展到数据特征的维度是多维的情况，在上述数据中增加一个影响火灾影响面积的潜在因素—风力

气温x	5.1	8.2	11.5	13.9	15.1	16.2	19.6	23.3
风力z	4.5	5.8	4	6.3	4	7.2	6.3	8.5
火灾影响面积y	2.14	4.62	8.24	11.24	13.99	16.33	19.23	28.74

对于上面的例子，转化为矩阵的表示形式为：

$$X = \begin{bmatrix} 5.1 & 8.2 & 11.5 & 13.9 & 15.1 & 16.2 & 19.6 & 23.3 \\ 4.5 & 5.8 & 4. & 6.3 & 4. & 7.2 & 6.3 & 8.5 \\ 1. & 1. & 1. & 1. & 1. & 1. & 1. & 1. \end{bmatrix}^{\mathrm{T}}$$

$$\mathrm{Y} = [2.14 \quad 4.62 \quad 8.24 \quad 11.24 \quad 13.99 \quad 16.33 \quad 19.23 \quad 28.74]^{T}$$

计算可得

$$\mathrm{W} = [1.312 \quad 0.626 \quad -9.103]$$

$$Y = 1.312x + 0.626z - 9.103$$

使用sklearn实现

In [10]:
```
import numpy as np
from sklearn.linear_model import LinearRegression
x = np.array([[5.1],[8.2],[11.5],[13.9],[15.1],[16.2],[19.6],[23.3]])
y = np.array([2.14,4.62,8.24,11.24,13.99,16.33,19.23,28.74])
reg = LinearRegression().fit(x,y)
```

In [11]:
```
print(reg.coef_)
print(reg.intercept_)
```

```
[1.42835273]
-7.09137783221065
```

In [14]:
```
reg.predict([[18.1]])
```

Out[14]:
```
array([18.76180649])
```

In [15]:
```
x = np.array([[5.1,4.5],[8.2,5.8],[11.5,4],[13.9,6.3],[15.1,4],[16.2,7.2],[19.6,6.3],[23.3,8.5]])
y = np.array([2.14,4.62,8.24,11.24,13.99,16.33,19.23,28.74])
```

In [16]:
```
reg = LinearRegression().fit(x,y)
```

In [17]:
```
print(reg.coef_)
print(reg.intercept_)
```

```
[1.31227219 0.62649508]
-9.102525137839692
```

线性回归

- 思考：

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

若$X^{\mathrm{T}}X$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)

正则化(Regularization)

- 限制假设空间复杂度：提升泛化能力
- 更稳定的数值优化

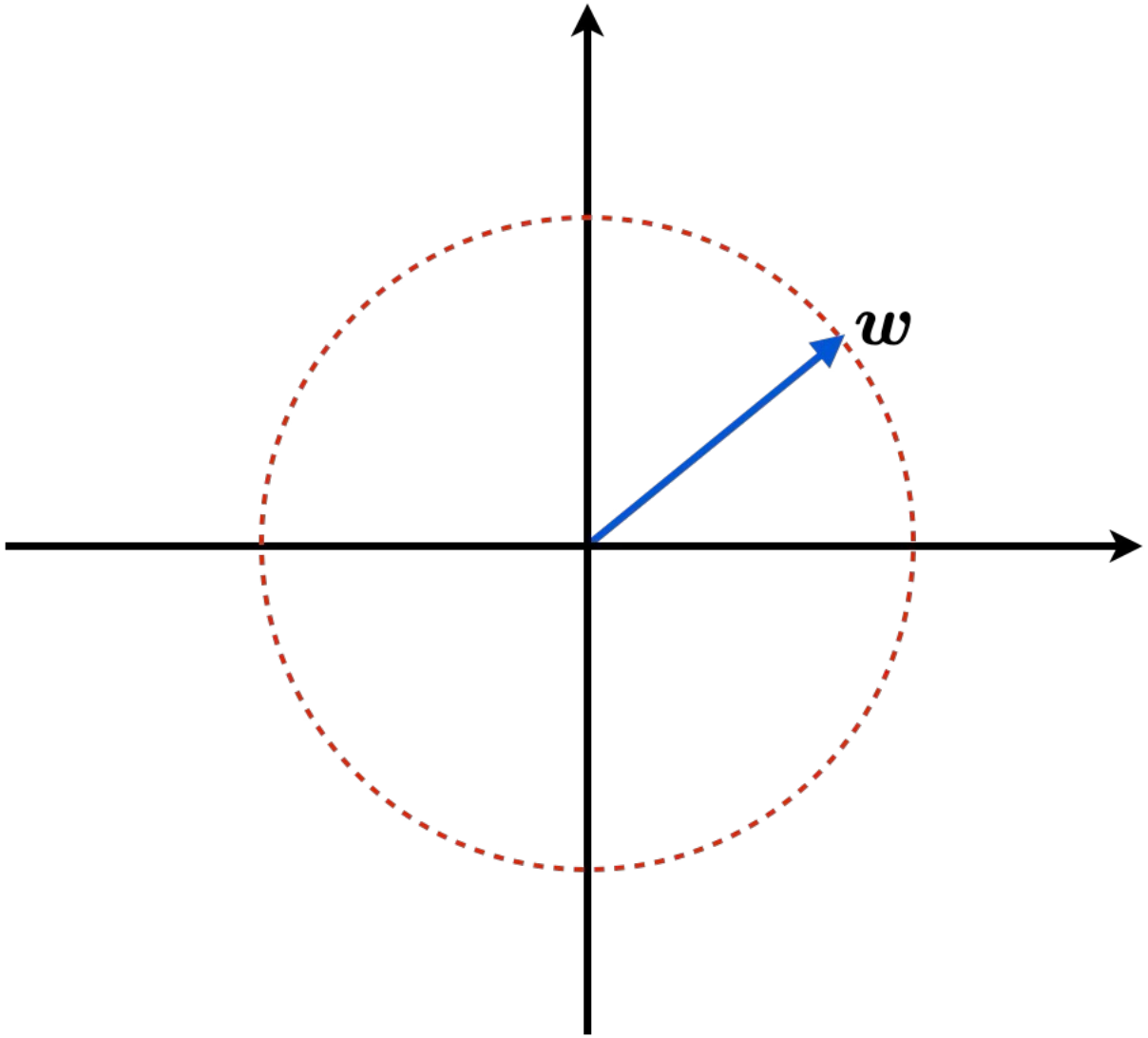
一般做法：限制参数w

$$\|\boldsymbol{w}\|_p = \left(\sum_{i=1}^{n} |w_i|^p \right)^{1/p}$$

$$\|\boldsymbol{w}\|_2 = \sqrt{\sum_{i=1}^{n} w_i^2}$$

$$\|\boldsymbol{w}\|_1 = \sum_{i=1}^{n} |w_i|$$

$$\|\boldsymbol{w}\|_\infty = \max_{i=1,\ldots,n} |w_i|$$



正则化(Regularization)

- 岭回归(Ridge Regression)

$$\arg\min_{\boldsymbol{w},b} \frac{1}{m}\sum_{i=1}^{m}(\boldsymbol{w}^\top \boldsymbol{x}_i + b - y_i)^2 + \lambda\|\boldsymbol{w}\|_2$$

- LASSO Regression

$$\arg\min_{\boldsymbol{w},b} \frac{1}{m}\sum_{i=1}^{m}(\boldsymbol{w}^\top \boldsymbol{x}_i + b - y_i)^2 + \lambda\|\boldsymbol{w}\|_1$$

结构风险最小化(Structural Risk Minimization, SRM)

$$\arg\min_{\boldsymbol{w},b} L(\boldsymbol{w},b) + \|\boldsymbol{w}\|_p$$

经验风险　　结构风险

大纲

- 近邻算法
- 线性回归
- 对数几率回归

广义线性模型

对于样例 $(\boldsymbol{x}, y)$, $y \in \mathbb{R}$,望线性模型的预测值逼近真实标记，则得到线性回归模型 $y = \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} + b$

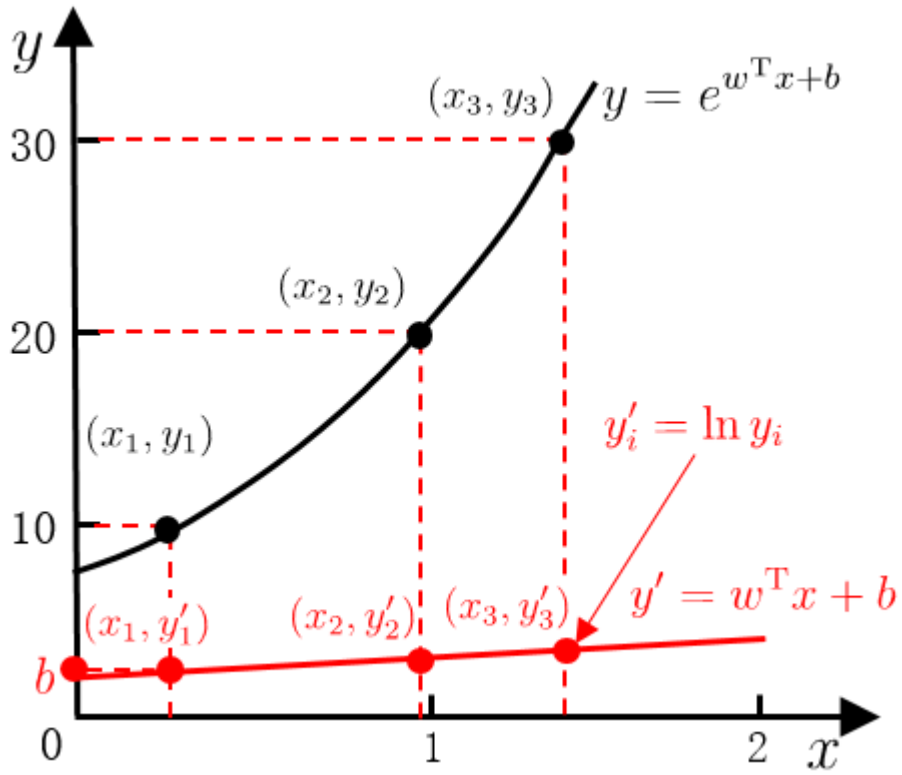
令预测值逼近 y 的衍生物？

若令 $\ln y = \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} + b$

则得到对数线性回归

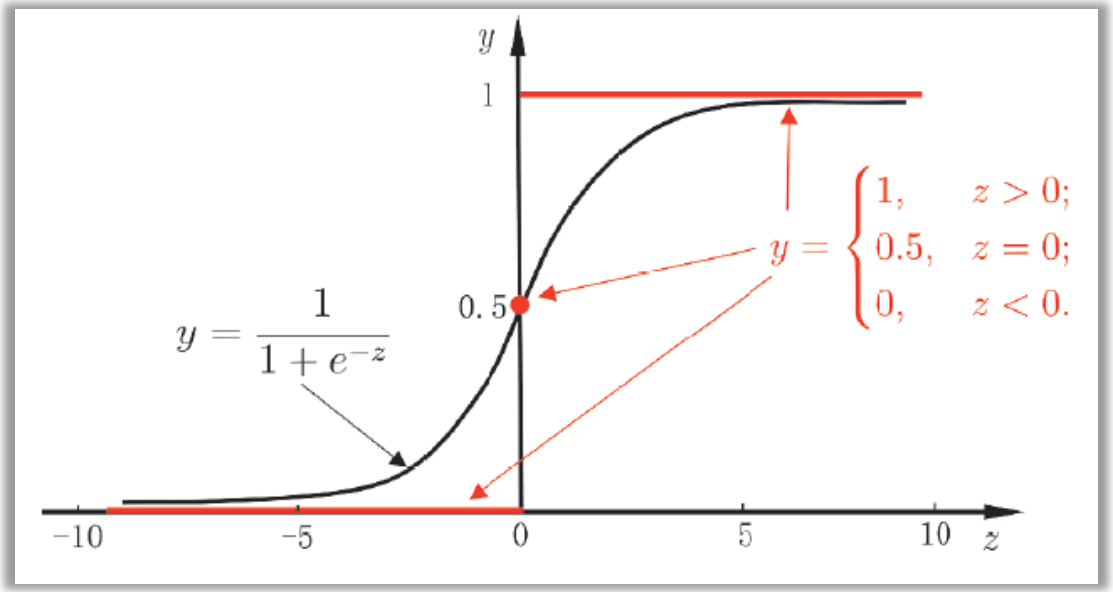
(log-linear regression)

实际是在用 $e^{\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+b}$ 逼近 y



考虑二分类任务

- 线性回归模型产生的实值输出 $z = w^{\mathsf{T}}x + b$
- 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)

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

“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

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

似然项

极大似然估计
(maximum likelihood method)

- 根据

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

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

- 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]

梯度下降(Gradient Descent)

- 梯度下降算法是一种使得函数最小化的迭代式优化方法

- 函数f在w处的取值为$f(w)$，梯度为$\nabla f(w)$

- 参数更新方法为：

$$w_i = w_{i-1} - \eta \nabla f(w_{i-1})$$

η一般被称为步长(step size)或者学习率(learning rate)

学习率$\eta \in [0,1]$，不能太大、不能太小

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
```

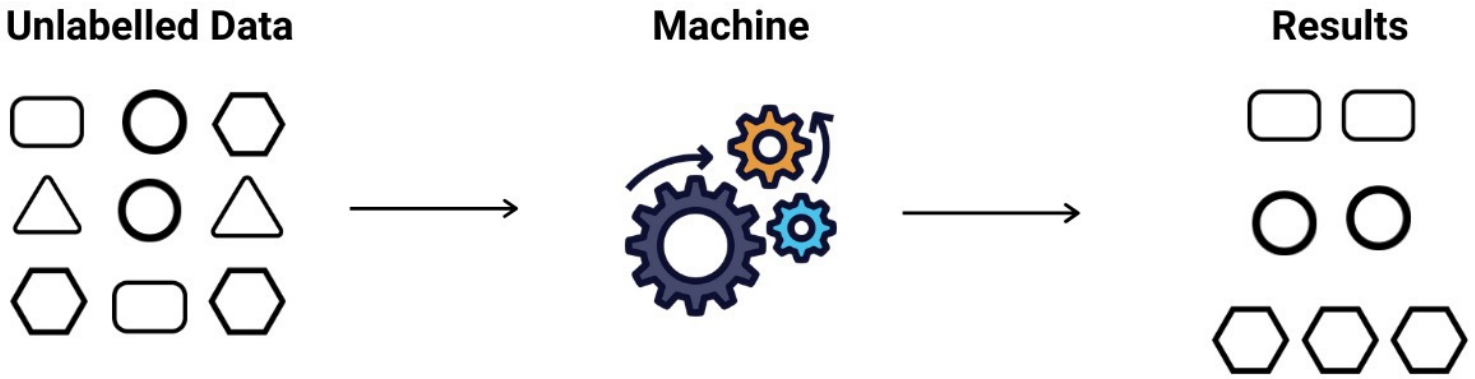
对数几率回归

思考：

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

无监督学习

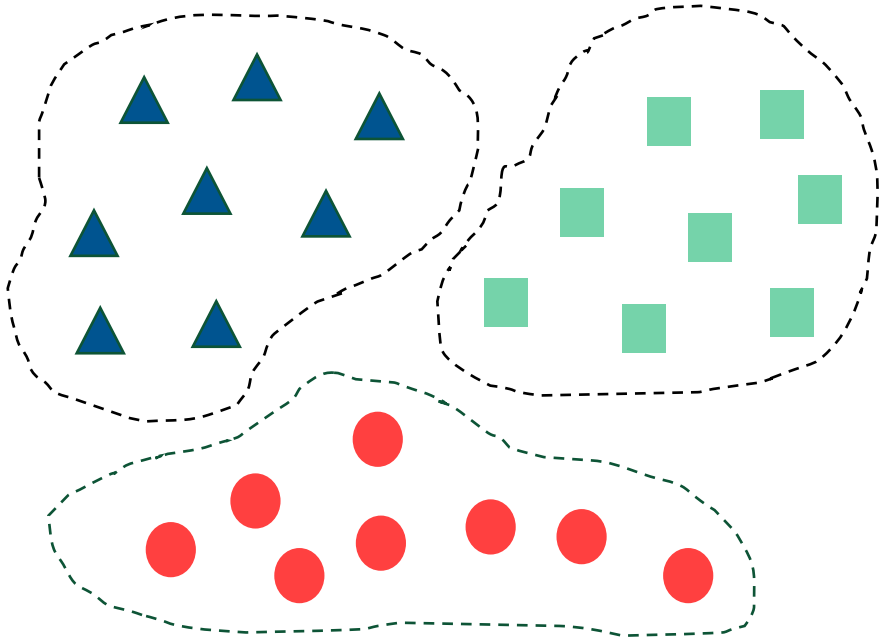
无监督学习：所有训练样本均没有标注



聚类(clustering)

在“无监督学习”任务中研究最多、应用最广

目标：将数据样本划分为若干个通常不相交的“簇”(cluster)



既可以作为一个单独过程用于找寻数据内在的分布结构，也可作为分类等其他学习任务的前驱过程

聚类的评价指标

聚类的“好坏”不存在绝对标准

the goodness of clustering depends on the opinion of the user

基本原则：

- “簇内相似度”（intra-cluster similarity）高，且
- “簇间相似度”（inter-cluster similarity）低

故事一则

聚类的故事：

老师拿来苹果和梨，让小朋友分成两份。

小明把大苹果大梨放一起，小个头的放一起，老师点头，恩，体量感。

小芳把红苹果挑出来，剩下的放一起，老师点头，颜色感。

小武的结果？不明白。小武掏出眼镜：最新款，能看到水果里有几个籽，左边这堆单数，右边双数。

老师很高兴：新的聚类算法诞生了

聚类也许是机器学习中“新算法”出现最多、最快的领域
总能找到一个新的“标准”，使以往算法对它无能为力

K均值聚类(K-Means)

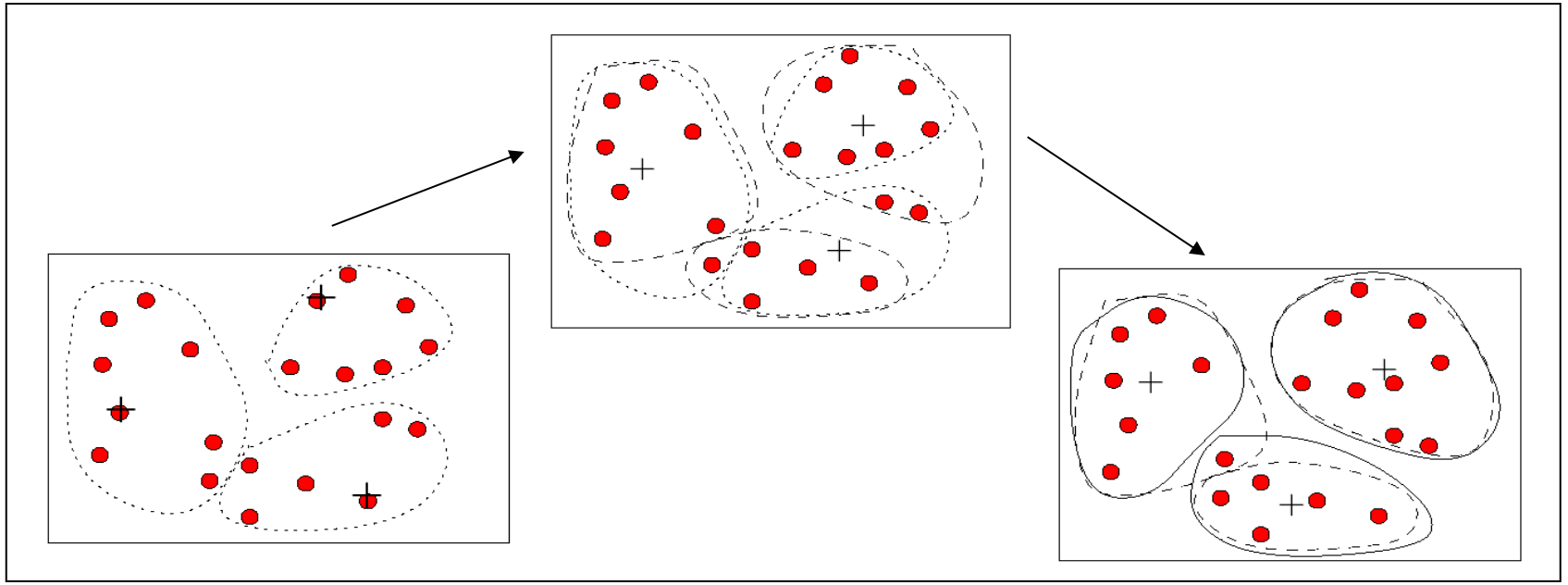
每个簇以该簇中所有样本点的“均值”表示

Step1: 随机选取k个样本点作为簇中心

Step2: 将其他样本点根据其与簇中心的距离，划分给最近的簇

Step3: 更新各簇的均值向量，将其作为新的簇中心

Step4: 若所有簇中心未发生改变，则停止；否则执行 Step 2



K均值聚类示例

例子：给定含有5个样本的集合，用K均值聚类将其聚成2类

$$X = \begin{bmatrix} 0 & 0 & 1 & 5 & 5 \\ 2 & 0 & 0 & 0 & 2 \end{bmatrix}$$

（1）选择两个样本点作为类中心，(0,2), (0, 0)

（2）通过计算距离，第一个簇为{1,5}、第二个簇为{2,3,4}

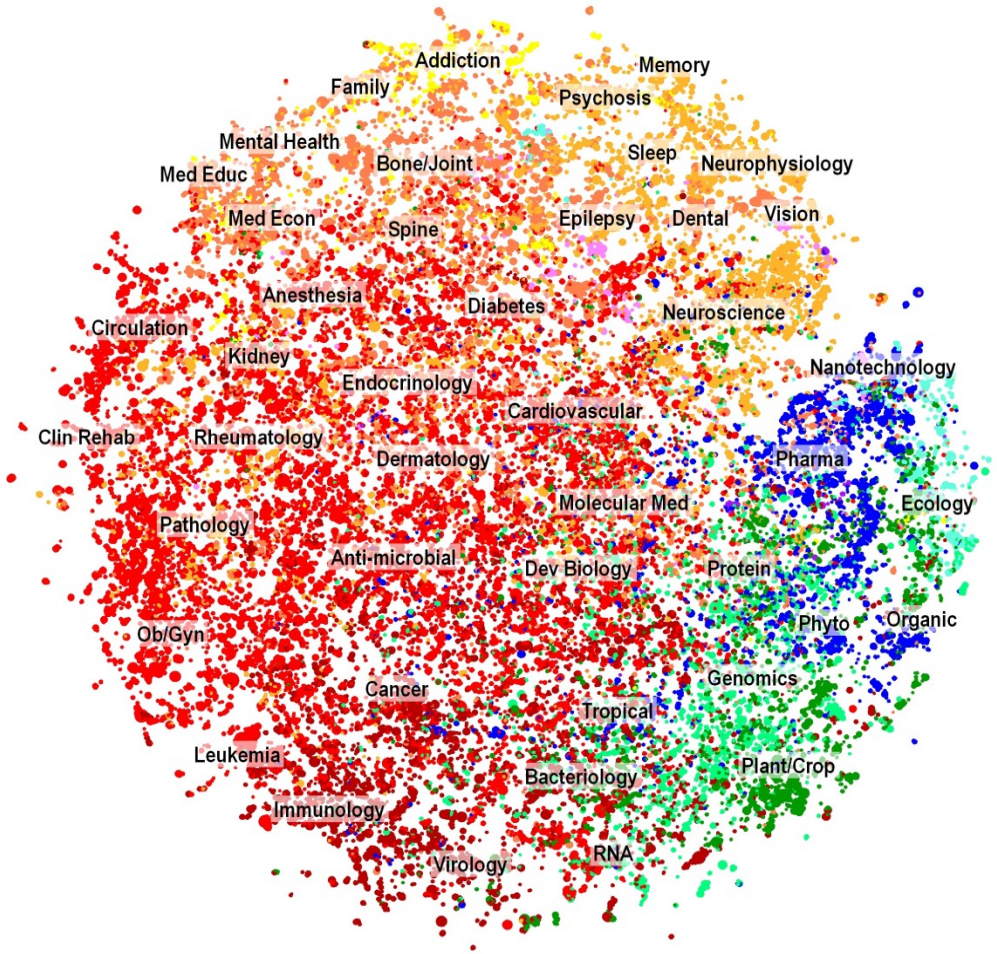
（3）计算新的类中心：(2.5,2), (2, 0)

（4）新的簇：第一个簇为{1,5}、第二个簇为{2,3,4}

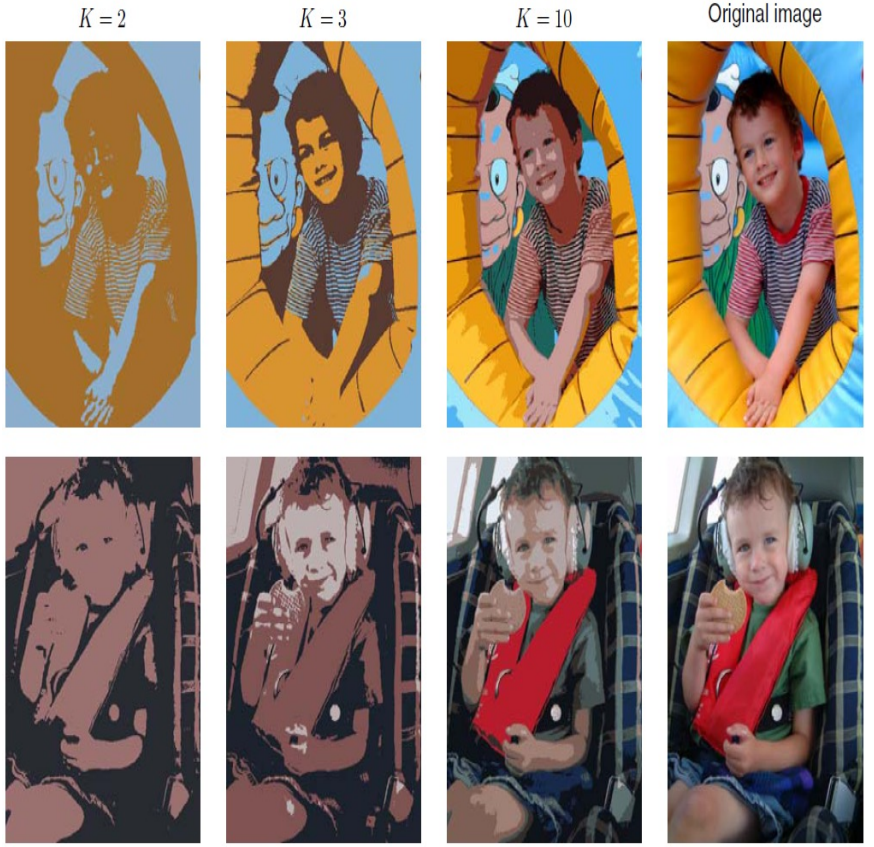
K均值聚类的不足

- 需要事先确定聚类数目，很多时候我们并不知道数据应被聚类的数目
- 需要初始化聚类质心，初始化聚类中心对聚类结果有较大的影响
- 算法是迭代执行，时间开销非常大
- 距离计算往往假设数据每个维度之间的重要性是一样的

K-Means的应用



文本分类：将200多万篇论文聚类到29,000个类别，包括化学、工程、生物、传染疾病、生物信息、脑科学、社会科学、计算机科学等及给出了每个类别中的代表单词



色彩压缩：每个簇的颜色变为同一种