深度学习(Deep Learning)

深度学习基础知识

张新峰
计算机科学与技术学院
中国科学院大学
邮箱: xfzhang@ucas.ac.cn

提纲

- 数学基础
- 机器学习基础
- 神经元模型
- 感知器及多层感知器
- BackPropagation算法
- 中英文术语对照

1 数学基础

线性代数

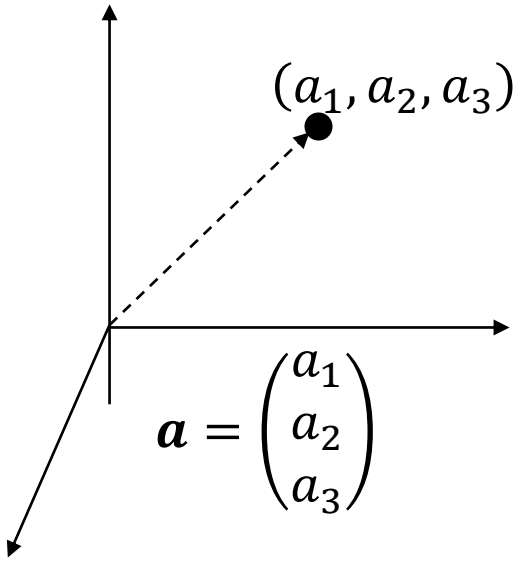
❑ 标量(Scalar)

- 只有大小没有方向的物理量，如时间，温度、质量等，$a_1 = 1$
- 如果该标量是实数，记做$a_1 \in R$

❑ 向量(Vector)

- 又称矢量，既包含大小又包含方向的物理量，如速度、位移等，一般用粗体变量表示
- 如果是n维实数矢量，记做$\boldsymbol{a} \in R^n$

$$\boldsymbol{a} = \begin{bmatrix} a_1 \\ \vdots \\ a_i \\ \vdots \\ a_n \end{bmatrix} \quad \text{或者} \quad \boldsymbol{a} = [a_1 \cdots a_i \cdots a_n]^T$$



矩阵

❑ 矩阵(Matrix)

- 矩阵是一个二维数组，其中的每一个元素一般由两个索引来确定，一般用大写变量表示
- m行n列的实数矩阵，记做$\mathrm{A} \in R^{m \times n}$

$$\mathrm{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

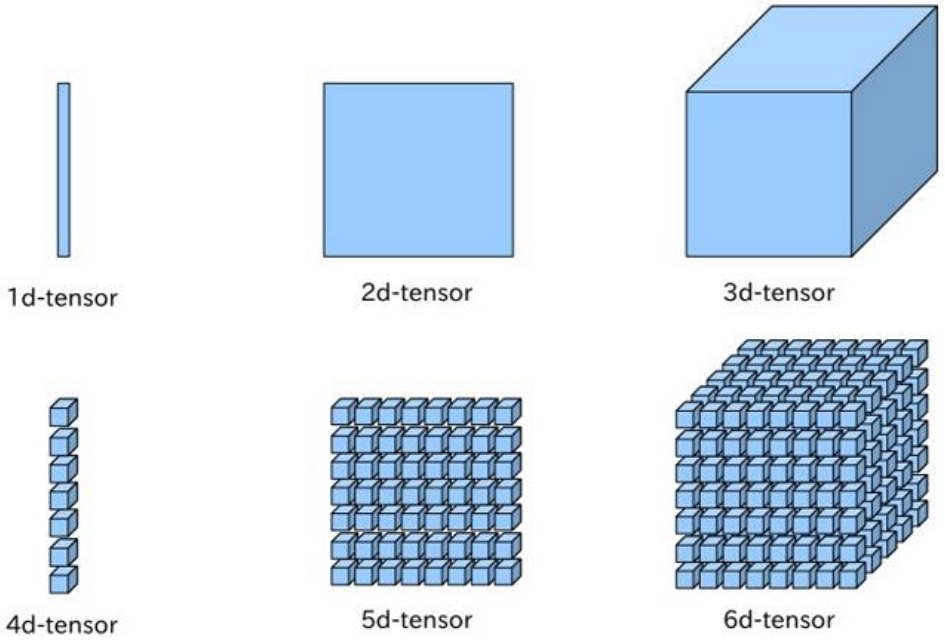
$$\mathrm{A} = [\boldsymbol{a}_1, \boldsymbol{a}_2, \boldsymbol{a}_3] \qquad \boldsymbol{a}_j = \begin{bmatrix} a_{1j} \\ a_{2j} \\ a_{3j} \end{bmatrix}$$

- $\mathrm{A}_{ij}=a_{ij}$矩阵A第i行第j列的元素

张量

❑ 张量(Tensor)

- 矢量概念的推广，可用来表示在一些矢量、标量和其他张量之间的线性关系的多线性函数
- 标量是0阶张量，矢量是一阶张量，矩阵是二阶张量
- 三维及以上数组一般称为张量



矩阵运算

❑ 矩阵加法

– 两个矩阵满足行数和列数相等时，两个矩阵可以相加

$$\mathrm{A}+\mathrm{B}=\mathrm{C}$$

$$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} + \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix} = \begin{bmatrix} a_{11}+b_{11} & a_{12}+b_{12} \\ a_{21}+b_{21} & a_{22}+b_{22} \end{bmatrix}$$

❑ 矩阵乘法

– 两个矩阵满足第一个矩阵的列数与第二个矩阵的行数相等时，两个矩阵可以相乘

$$\mathrm{AB}=\mathrm{C}$$

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \\ b_{31} & b_{32} \end{bmatrix} = \begin{bmatrix} \sum_{k=1}^{3} a_{1k}b_{k1} & \sum_{k=1}^{3} a_{1k}b_{k2} \\ \sum_{k=1}^{3} a_{2k}b_{k1} & \sum_{k=1}^{3} a_{2k}b_{k2} \\ \sum_{k=1}^{3} a_{3k}b_{k1} & \sum_{k=1}^{3} a_{3k}b_{k2} \end{bmatrix}$$

矩阵的转置

❑ 矩阵的转置(Transpose)

- 将矩阵的行列互换得到的新矩阵称为转置矩阵
- $m \times n$的矩阵A转置后为$n \times m$的矩阵A^{T}

$$\mathrm{A}_{ij}^{T} = \mathrm{A}_{ji}, \forall\, 1 \le i \le m, 1 \le j \le n$$

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{bmatrix}^{T} = \begin{bmatrix} 1 & 3 & 5 \\ 2 & 4 & 6 \end{bmatrix}$$

矩阵的秩

❑ 矩阵的秩(Rank)

– 矩阵列向量中的极大线性无关组的数目，记作矩阵的列秩，同样可以定义行秩。行秩=列秩=矩阵的秩，通常记作rank(A）

$$\mathrm{A} = [\boldsymbol{a}_1, \boldsymbol{a}_2, \cdots, \boldsymbol{a}_k, \cdots \boldsymbol{a}_n], \qquad \boldsymbol{a}_1 \in R^m, \quad \mathrm{A} \in R^{m \times n}$$

矩阵的秩

❑ 矩阵的秩(Rank)

– 矩阵列向量中的极大线性无关组的数目，记作矩阵的列秩，同样可以定义行秩。行秩=列秩=矩阵的秩，通常记作rank(A）

$$\mathrm{A} = [\boldsymbol{a}_1, \boldsymbol{a}_2, \cdots, \boldsymbol{a}_k, \cdots \boldsymbol{a}_n], \qquad \boldsymbol{a}_1 \in R^m, \quad \mathrm{A} \in R^{m \times n}$$

– 线性相关组

如果存在不全为零的数 $\alpha_1, \alpha_2, \cdots, \alpha_t$, 使：

$$\alpha_1 \boldsymbol{a}_1 + \alpha_2 \boldsymbol{a}_2 + \cdots \alpha_t \boldsymbol{a}_t = \boldsymbol{o},$$

其中$\boldsymbol{o}$是m维的全零向量，那么向量组$\boldsymbol{a}_1, \boldsymbol{a}_2, \cdots, \boldsymbol{a}_t$称为线性相关

$$\boldsymbol{a}_1 = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} \quad \boldsymbol{a}_2 = \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix} \quad \boldsymbol{a}_3 = \begin{bmatrix} 2 \\ 3 \\ 5 \end{bmatrix} \qquad 2\boldsymbol{a}_1 + 3\boldsymbol{a}_2 - \boldsymbol{a}_3 = \boldsymbol{o}$$

矩阵的逆

❑ 矩阵的逆

- 若矩阵A为方阵，当$rank(A_{n\times n})<n$时，称A为奇异矩阵或不可逆矩阵
- 若矩阵A为方阵，当$rank(A_{n\times n})=n$时，称A为非奇异矩阵或可逆矩阵
  - 其逆矩阵A^{-1}满足以下条件，则称A^{-1}为矩阵A的逆矩阵

$$AA^{-1}=A^{-1}A=I_n$$

  - 其中I_n是$n\times n$的单位阵

$$I_n=\begin{bmatrix}1 & 0 & \cdots & 0\\ 0 & 1 & \cdots & 0\\ 0 & 0 & \ddots & \vdots\\ 0 & 0 & \cdots & 1\end{bmatrix}$$

矩阵的逆

❑ 矩阵的广义逆矩阵

- 如果矩阵不为方阵或者是奇异矩阵，不存在逆矩阵，但是可以计算其广义逆矩阵或者伪逆矩阵
- 对于矩阵A，如果存在矩阵B使得ABA=A，则称B为A的广义逆矩阵
- 通过对矩阵A进行奇异值分解，来计算其广义逆矩阵

$$A = U\begin{pmatrix}\Sigma & \mathbf{0}\\ \mathbf{0} & \mathbf{0}\end{pmatrix}V^T \quad \Rightarrow \quad B = V\begin{pmatrix}\Sigma^{-1} & \mathbf{0}\\ \mathbf{0} & \mathbf{0}\end{pmatrix}U^T$$

- 相关计算都存在库函数
  - MATALB: inv() 和 pinv()函数
  - PyTorch: inverse()和pinverse()函数

矩阵分解

❑ 机器学习中常见的矩阵分解

– 特征分解和奇异值分解

❑ 矩阵的特征值和特征向量

– 若矩阵A为方阵，则存在非零向量$\boldsymbol{x}$和常数λ满足$\mathrm{A}\boldsymbol{x} = \lambda\boldsymbol{x}$，则称λ为矩阵A的一个特征值，$\boldsymbol{x}$为矩阵A关于λ的特征向量

– $\mathrm{A}_{n\times n}$的矩阵具有n个特征值，$\lambda_1 \le \lambda_2 \le \cdots \le \lambda_n$,其对应的$n$个特征向量为$\boldsymbol{u}_1$，$\boldsymbol{u}_2$，…，$\boldsymbol{u}_n$

– 矩阵的迹(trace)和行列式(determinant)的值

$$tr(\mathrm{A}) = \sum_{i=1}^{n} \lambda_i \qquad |\mathrm{A}| = \prod_{i=1}^{n} \lambda_i$$

矩阵分解

❑ 矩阵特征分解(Eigendecomposition)

– 若矩阵$A_{n\times n}$存在n个不同的特征值，那么矩阵A可以分解为

$$A = U\Sigma U^T \longrightarrow \text{矩阵A的特征分解}$$

$$\Sigma = \begin{bmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 \\ 0 & 0 & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_n \end{bmatrix} \qquad U = [\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots, \boldsymbol{u}_n]$$

其中$\boldsymbol{u}_i$是标准化的特征向量，即满足$\|\boldsymbol{u}_i\|_2 = 1$

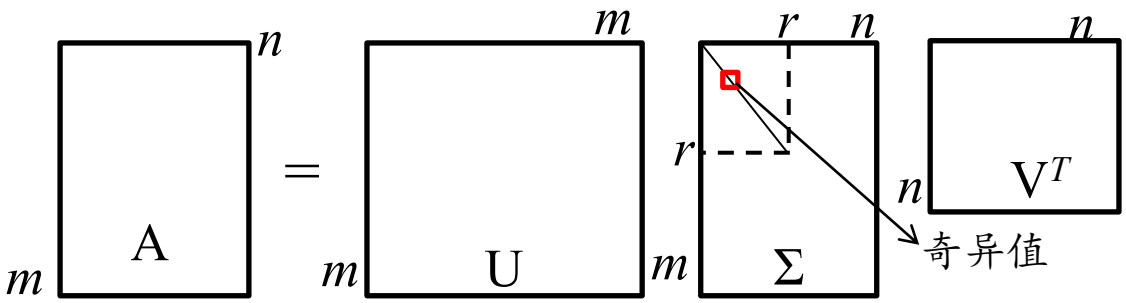
矩阵分解

❑ 奇异值分解(Singular value decomposition)

– 对于任意矩阵A_{mxn}，存在正交矩阵U_{mxm}，V_{nxn}，使得其满足

$$A=U\Sigma V^T \qquad U^TU= V^TV =I$$

则称上式为矩阵A的特征分解，其中Σ为mxn的矩阵



– 求解过程

- A^TA的特征值的$\{\lambda_i\}$ 和特征向量$\{\boldsymbol{v}_i\}$
- AA^T的特征向量$\{\boldsymbol{u}_i\}$
- $U=[\boldsymbol{u}_1,\ldots,\boldsymbol{u}_m]$, $V=[\boldsymbol{v}_1,\ldots,\boldsymbol{v}_m]$, $\Sigma=\mathrm{diag}(\sqrt{\lambda_i})$

概率和统计

❑ 随机变量(Random variable)

- 随机事件的数量表现，随机事件数量化的好处是可以用数学分析的方法来研究随机现象
- 随机变量可以是离散的或者连续的，离散随机变量是指拥有有限个或者可列无限多个状态的随机变量，连续随机变量是指变量值不可随机列举出来的随机变量，一般取实数值
- 随机变量通常用概率分布来指定它的每个状态的可能性。

例如：

1. 投掷一枚硬币为正面是离散型随机事件X，发生概率P(X=1)=0.5
2. 每次射箭距离靶心的距离X可以认为连续型随机变量，距离靶心小于1cm的概率P(X<1cm)

常见概率分布

❑ 伯努利分布(Bernoulli)

- 伯努利试验：只可能有两种结果的单次随机实验
- 又称0-1分布，单个二值型离散随机变量的分布
- 其概率分布：$P(X=1)=p, P(X=0)=1-p$

❑ 二项分布(Binomial)

- 二项分布即重复n次伯努利试验，各试验之间都相互独立
- 如果每次试验时，事件发生的概率为p，不发生的概率为1-p，则n次重复独立试验中事件发生k次的概率为

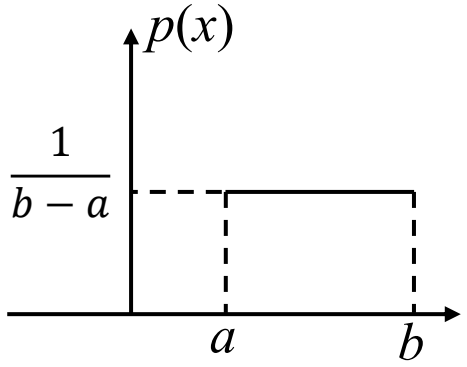
$$P(X=k)=C_n^k p^k(1-p)^{n-k}$$

常见概率分布

❑ 均匀分布(Uniform)

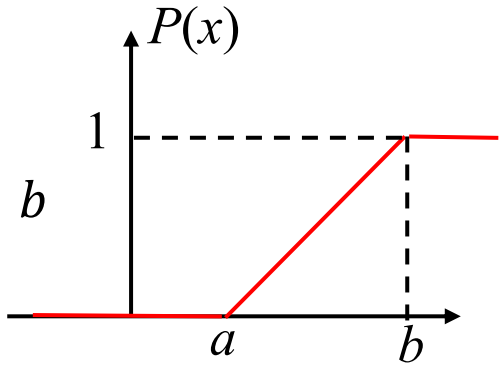
– 又称矩形分布，在给定长度间隔$[a,b]$内的分布概率是等可能的，均匀分布由参数a，b定义，概率密度函数为：

$$p(x)=\begin{cases}\dfrac{1}{b-a}, & \text{if } a\le x\le b\\ 0, & \text{else}\end{cases}$$



– 累积概率分布函数

$$P(X\le x)=\int_{-\infty}^{x}p(x)dx=\begin{cases}0, & \text{for } x<a\\ \dfrac{x-a}{b-a}, & \text{for } a\le x<b\\ 1, & \text{for } x\ge b\end{cases}$$



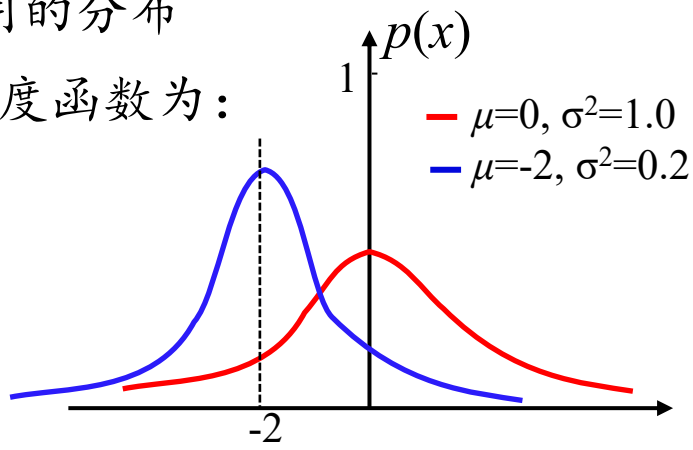
常见概率分布

❑ 高斯分布(Gaussian)

– 又称正态分布(normal)，是实数中最常用的分布

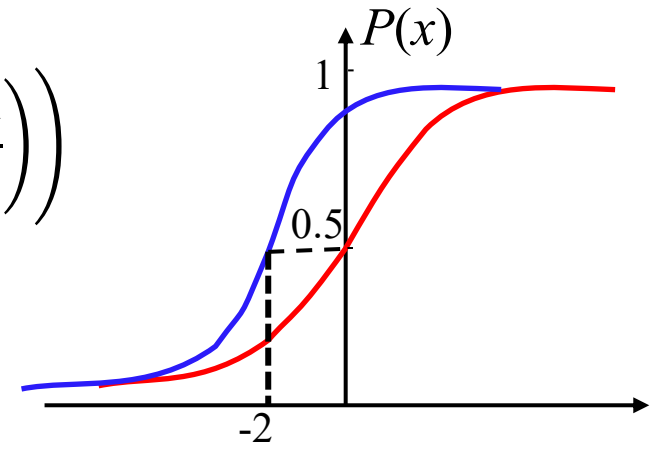
– 由均值μ和标准差σ决定其分布，概率密度函数为：

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



– 累积概率分布函数

$$P(X \le x) = \int_{-\infty}^{x} p(x)dx = \frac{1}{2}\left(1 + \text{erf}\left(\frac{x-\mu}{\sigma\sqrt{2}}\right)\right)$$

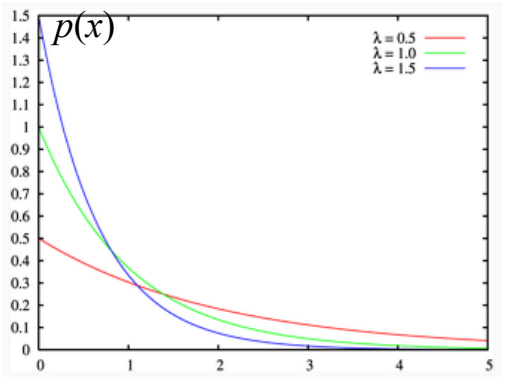


常见概率分布

❑ 指数分布(exponential)

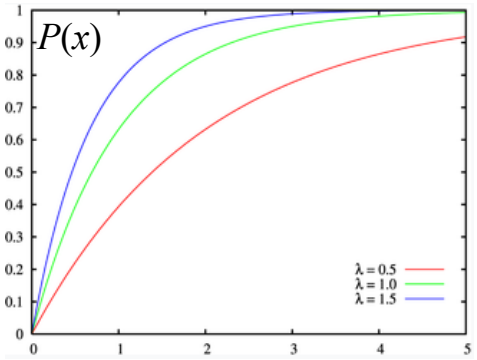
- 常用来表示独立随机事件发生的时间间隔
- 参数为λ>0的指数分布概率密度函数为：

$$p(x) = \begin{cases} \lambda e^{-\lambda x}, x \geq 0 \\ 0, \quad x < 0 \end{cases}$$



- 累积概率分布函数

$$P(X \leq x) = \int_{-\infty}^{x} p(x)dx = \begin{cases} 1 - e^{-\lambda x}, x \geq 0 \\ 0, \quad x < 0 \end{cases}$$



常见概率分布

❑ 指数分布(exponential)

– 指数分布重要特征是无记忆性

- 例如某元件寿命为X，已知该元件已经使用了x小时，那么它总共至少使用$x+s$小时的条件概率，与从开始使用时算起它使用至少s小时的概率相等，即：

$$P(X > s + x | X > x) = P(X > s)$$

证明：因为$X \sim Exp(\lambda)$，所以$P(X>x)=e^{-\lambda x}$，又因为

$$\{X>s+x\} \subseteq \{X>x\}$$

因此

$$P(X > s + x | X > x) = \frac{P(X > s + x, X > x)}{P(X > x)} = \frac{P(X > s + x)}{P(X > x)} = \frac{e^{-\lambda(s+x)}}{e^{-\lambda x}} = P(X > s)$$

多个随机变量概率分布

- **条件概率(Conditional probability)**
  - 事件X在事件Y发生的条件下发生的概率，$P(X|Y)$
- **联合概率(Joint probability)**
  - 表示两个事件X和Y共同发生的概率, $P(X,Y)$
- **条件概率和联合概率的性质**

$$P(Y|X) = \frac{P(Y,X)}{P(X)} \qquad P(X) > 0$$

推广到n个事件，条件概率的链式法则

$$P(X_1, X_2, \dots, X_n) = P(X_1|X_2, \dots, X_n)P(X_2|X_3, X_4, \dots, X_n) \dots P(X_{n-1}|X_n)P(X_n)$$

$$= P(X_n)\prod_{i=1}^{n-1} P(X_i|X_{i+1}, \dots, X_n)$$

多个随机变量概率分布

❑ 先验概率(Prior probability)

– 根据以往经验和分析得到的概率，在事件发生前已知，它往往作为“由因求果”问题中的“因”出现

❑ 后验概率(Posterior probability)

– 指得到“结果”的信息后重新修正的概率，是“执果寻因”问题中的“因”，后验概率是基于新的信息，修正后来的先验概率所获得的更接近实际情况的概率估计

举一个简单的例子：一口袋里有3只红球、2只白球，采用不放回方式摸取，求：
(1) 第一次摸到红球(记作A)的概率；
(2) 第二次摸到红球(记作B)的概率；
(3) 已知第二次摸到了红球，求第一次摸到的是红球的概率？
解：
(1) P(A=1) = 3/5, 这就是先验概率；
(2) P(B=1) = P(A=1) P(B=1|A=1)+ P(A=0)P(B=1|A=0)=$\frac{3}{5}\frac{2}{4}+\frac{2}{5}\frac{3}{4}=\frac{3}{5}$；
(3) P(A=1|B=1) = $\frac{P(\mathrm{A}=1)P(\mathrm{B}=1|\mathrm{A}=1)}{P(\mathrm{B}=1)}=\frac{1}{2}$, 这就是后验概率；

多个随机变量概率分布

❑ 全概率公式

– 设事件$\{A_i\}$是样本空间Ω的一个划分，且$P(A_i)>0$ $(i=1,2,\ldots,n)$,那么：

$$P(B)=\sum_{i=1}^{n}P(A_i)\,P(B|A_i)$$

样本空间划分的定义：(1) $\Omega=A_1\cup A_2\cup\ldots\cup A_n$

(2) $A_i\cap A_j=\emptyset,\ \ \forall i,j,i\neq j$

❑ 贝叶斯公式

– 全概率公式给我们提供了计算后验概率的途径，即贝叶斯公式：

$$P(A_i|B)=\frac{P(B|A_i)P(A_i)}{P(B)}=\frac{P(B|A_i)P(A_i)}{\sum_{j=1}^{n}P(A_j)\,P(B|A_j)}$$

信息论

❑ 熵(Entropy)

- 信息熵，可以看作是样本集合纯度一种指标，也可以认为是样本集合包含的平均信息量
- 假定当前样本集合X中第i类样本x_i所占的比例为$P(x_i)(i=1,2,\ldots,n)$，则X的信息熵定义为：

$$H(\mathrm{X}) = -\sum_{i=1}^{n} P(x_i)\log_2 P(x_i)$$

$H(X)$的值越小，则X的纯度越高，蕴含的不确定性越少

❑ 联合熵

- 两个随机变量X和Y的联合分布可以形成联合熵，度量二维随机变量XY的不确定性

$$H(\mathrm{X},\mathrm{Y}) = -\sum_{i=1}^{n}\sum_{j=1}^{n} P(x_i, y_j)\log_2 P(x_i, y_j)$$

❑ 条件熵

– 在随机变量X发生的前提下，随机变量Y发生带来的熵，定义为Y的条件熵，用H(Y|X)表示，定义为：

$$H(\mathrm{Y}|\mathrm{X}) = -\sum_{i=1}^{n}\sum_{j=1}^{n} P(x_i, y_j)\log_2 P(y_j|x_i)$$

$$H(\mathrm{Y}|\mathrm{X}) = \sum_{i=1}^{n} P(x_i)H(Y|X=x_i) = -\sum_{i=1}^{n} P(x_i)\sum_{j=1}^{n} P(y_j|x_i)log_2 P(y_j|x_i)$$

$$= -\sum_{i=1}^{n}\sum_{j=1}^{n} P(x_i, y_j)\log_2 P(y_j|x_i)$$

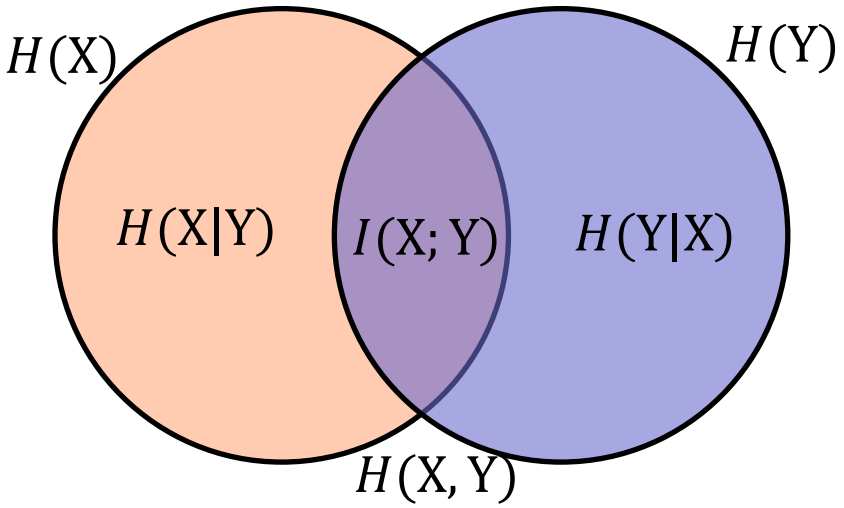
条件熵用来衡量在已知随机变量X的条件下，随机变量Y的不确定。熵、联合熵和条件熵之间的关系如下：

$$H(\mathrm{Y}|\mathrm{X}) = H(\mathrm{X},\mathrm{Y}) - H(\mathrm{X})$$

信息论

❑ 互信息的定义

$$I(X;Y) = H(X)+H(Y)-H(X,Y)$$



❑ 信息增益

$$G(Y, X) = H(Y)-H(Y|X)$$

信息论

❑ 相对熵

- 又称KL散度
- 描述两个概率分布P和Q差异的一种方法，记做D(P||Q)
- 在信息论中，D(P||Q)表示用概率分布Q来拟合真实分布P时，产生的信息表达的损耗，其中P表示信源的真实分布，Q表示P的近似分布
  - 也就是：使用基于Q的分布来编码服从P的分布的样本所需的额外的平均比特数

离散形式 $D(P||Q) = \sum P(x) \log \frac{P(x)}{Q(x)}$

连续形式 $D(P||Q) = \int P(x) \log \frac{P(x)}{Q(x)}$

信息论

❑ 相对熵

- 又称KL散度
- 比如分类问题：随机变量X~P，取值为1, 2, 3时概率为[1,0,0]，估计的变量Y~Q，取值为1, 2, 3时概率为[0.7, 0.2, 0.1]

$$D(P||Q) = 1 * log\left(\frac{1}{0.7}\right) + 0 * log\left(\frac{0}{0.2}\right) + 0 * log\left(\frac{0}{0.1}\right)$$

信息论

❑ 交叉熵(cross entropy)

– 一般用来求目标与预测值之间的差距，深度学习中经常用到的一类损失函数度量，比如在对抗生成网络（GAN）中

$$\mathrm{D(P||Q)} = \sum P(x)\log\frac{P(x)}{Q(x)} = \sum P(x)\log P(x) - \sum P(x)\log Q(x)$$

$$= -H\big(P(x)\big) - \sum P(x)\log Q(x)$$

交叉熵 $H(P,Q) = -\sum P(x)\log Q(x)$

常用统计量

❑ 期望(Expectation)

- 在概率和统计学中，数学期望是试验中每次可能结果的概率乘以其结果的总和，反映随机变量平均值的大小
- 假设X是离散随机变量，可能取值$(x_1,x_2,\ldots,x_n)$，各取值的概率为$P(x_k)$, 则期望的计算为：

$$E(X)=\sum_{k=1}^{n}x_kP(x_k)$$

- 假设X是连续随机变量，其概率密度函数为$p(x)$，则期望的计算为

$$E(X)=\int_{-\infty}^{+\infty}xp(x)dx$$

常用统计量

❑ 方差(Variance)

- 用来衡量随机变量与数学期望之间的偏离程度
- 统计中的方差则为样本方差，是各个样本数据分别与其平均数之差的平方和的平均数，计算过程为：

$$Var(X) = E\{[x - E(x)]^2\} = E(x^2) - [E(x)]^2$$

❑ 协方差(Covariance)

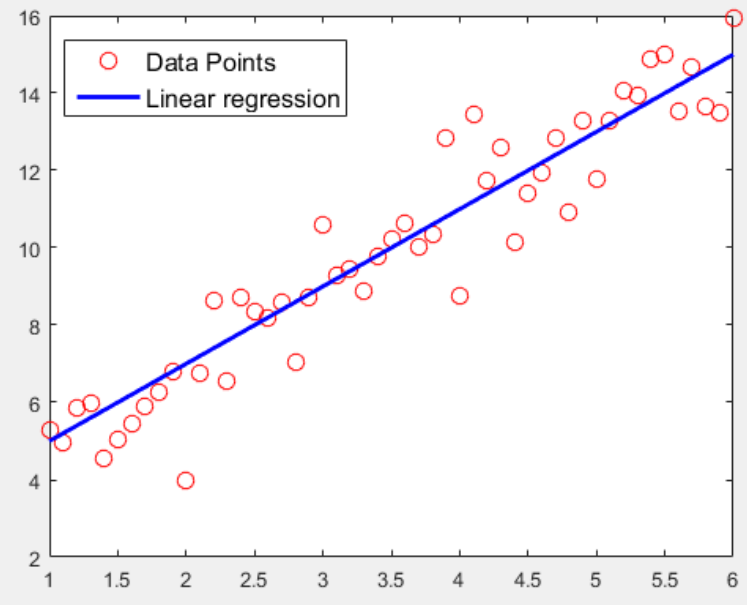
- 衡量两个随机变量X和Y直接的总体误差，计算过程为：

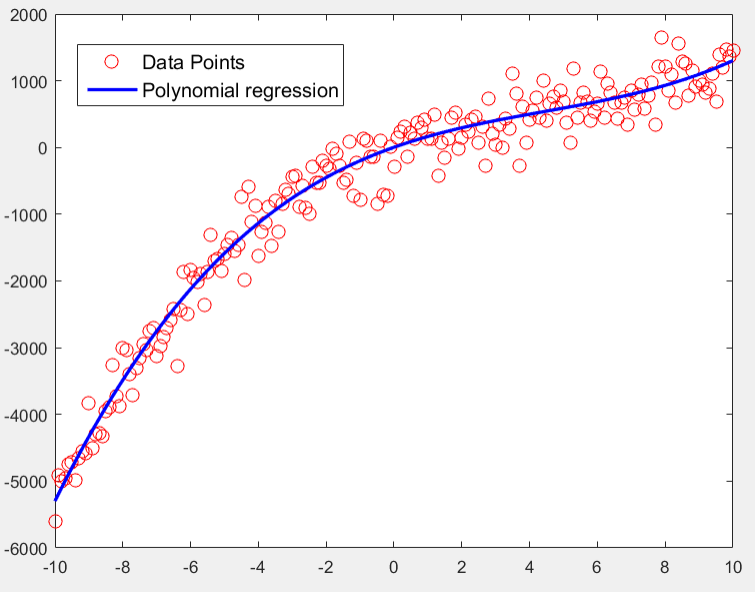
$$Cov(X, Y) = E\{[x - E(x)][y - E(y)]\} = E(xy) - E(x)\,E(y)$$

最优化估计方法

❑ 最小二乘法(Least Squares Method)

- 又称最小平方法，是一种数学优化方法。它通过最小化误差的平方和寻找数据的最佳函数匹配
- 最小二乘法经常应用于回归问题，可以方便地求得未知参数
  - 曲线拟合、最小化能量或者最大化熵等问题





最优化估计方法

❑ 最小二乘法(Least Squares Method)

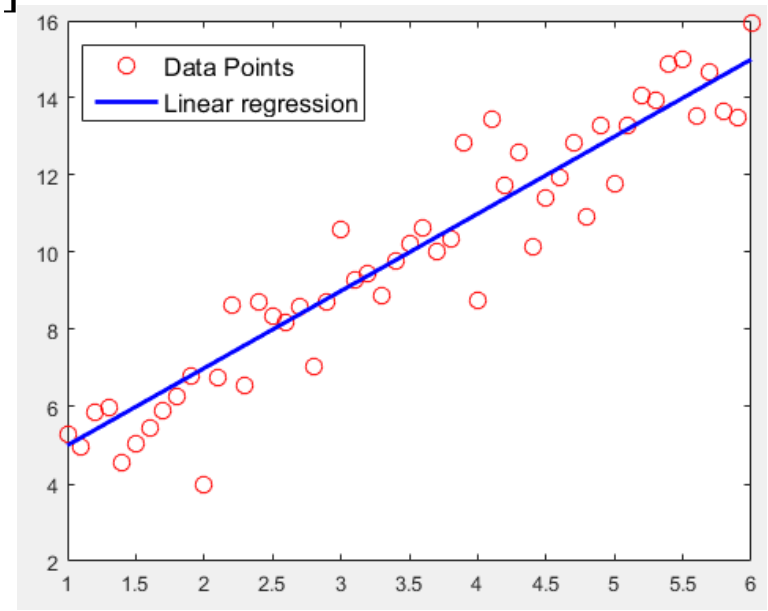
– 数学定义：给定函数$f(\boldsymbol{x};\alpha_0,\alpha_1,\alpha_2,\ldots,\alpha_m)$及其在$N$个不同点$\boldsymbol{x}_1,\boldsymbol{x}_2,\ldots,\boldsymbol{x}_n$的测量值$y_1,y_2,\ldots,y_n$, 即$f(\boldsymbol{x};\alpha_0,\alpha_1,\alpha_2,\ldots,\alpha_m)$的值，那么确定未知参数集,可以通过最小化下式获得($n>m$时有解),

$$\min_{\boldsymbol{\alpha}} J(\boldsymbol{\alpha}) = \frac{1}{n}\sum_{i=1}^{n}[f(\boldsymbol{x};\alpha_0,\alpha_1,\alpha_2,\ldots,\alpha_m) - y_i]^2$$

损失函数/代价函数/目标函数

$$f(\boldsymbol{x};\alpha_0,\alpha_1,\alpha_2,\ldots,\alpha_m) = \alpha_0 + \sum_{j=1}^{m}\alpha_j\,\varphi_j(x_j)$$

如果$\varphi_j(x_j) = x_j$，那么f是线性函数



最优化估计方法

❑ 最小二乘法(Least Squares Method)

– 如果$f(\boldsymbol{x};\alpha_0, \alpha_1, \alpha_2,..., \alpha_m) = \alpha_0 + \sum_{j=1}^{m} \alpha_j x_j$,

$$\min_{\boldsymbol{\alpha}} J(\boldsymbol{\alpha}) = \frac{1}{n}\sum_{i=1}^{n}\left(\alpha_0 + \sum_{j=1}^{m} \alpha_j x_{ij} - y_i\right)^2$$

$$\begin{aligned}
&\alpha_0 + \alpha_1 x_{11} + \cdots + \alpha_j x_{1j} + \cdots + \alpha_m x_{1m} = y_1 \\
&\alpha_0 + \alpha_1 x_{21} + \cdots + \alpha_j x_{2j} + \cdots + \alpha_m x_{2m} = y_2 \\
&\qquad\qquad \vdots \\
&\alpha_0 + \alpha_1 x_{i1} + \cdots + \alpha_j x_{ij} + \cdots + \alpha_m x_{im} = y_i \\
&\qquad\qquad \vdots \\
&\alpha_0 + \alpha_1 x_{n1} + \cdots + \alpha_j x_{nj} + \cdots + \alpha_m x_{nm} = y_n
\end{aligned}$$

$$A = \begin{bmatrix} 1 & x_{11} & \cdots & x_{1m} \\ 1 & x_{21} & \cdots & x_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & \cdots & x_{nm} \end{bmatrix}$$

$$\alpha = [\alpha_0, \alpha_1, \cdots, \alpha_m]^T$$

$$\boldsymbol{y} = [y_1, \cdots, y_m]^T$$

$$\mathrm{A}\boldsymbol{\alpha} = \boldsymbol{y}$$

最优化估计方法

❑ 最小二乘法(Least Squares Method)

– 如果$f(\boldsymbol{x};\alpha_0, \alpha_1, \alpha_2,..., \alpha_m) = \alpha_0 + \sum_{j=1}^{m} \alpha_j x_j$,

$$\min_{\boldsymbol{\alpha}} J(\boldsymbol{\alpha}) = \frac{1}{n}\sum_{i=1}^{n}\left(\alpha_0 + \sum_{j=1}^{m} \alpha_j x_{ij} - y_i\right)^2$$

$$A = \begin{bmatrix} 1 & x_{11} & \cdots & x_{1m} \\ 1 & x_{21} & \cdots & x_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & \cdots & x_{nm} \end{bmatrix}$$

$$\alpha = [\alpha_0, \alpha_1, \cdots, \alpha_m]^T$$

$$\boldsymbol{y} = [y_1, \cdots, y_m]^T$$

– 非线性函数的最小二乘

$$f(\boldsymbol{x};\alpha_0, \alpha_1, \alpha_2,..., \alpha_m) = \alpha_0 + \sum_{j=1}^{m} \alpha_j \varphi_j(x_j)$$

例如： $\varphi_j(x_j) = (x_j)^j$

最优化估计方法

❑ 最小二乘法(Least Squares Method)

– 方程求解

$$A\boldsymbol{\alpha} = \boldsymbol{y} \quad \Longleftrightarrow \quad \min_{\boldsymbol{\alpha}} \sum_{i=1}^{n} [f(\boldsymbol{x}; \alpha_0, \alpha_1, \alpha_2, \ldots, \alpha_m) - y_i]^2$$

$$\downarrow$$

$$A^T A\boldsymbol{\alpha} = A^T \boldsymbol{y} \quad \longrightarrow \quad \boldsymbol{\alpha} = (A^T A)^{-1} A^T \boldsymbol{y}$$

– 病态矩阵

- 如果对数据进行较小的扰动，则得出的结果具有很大波动，这样的矩阵称为**病态矩阵**

$$A = \begin{bmatrix} 1 & x_{11} & \cdots & x_{1m} \\ 1 & x_{21} & \cdots & x_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & \cdots & x_{nm} \end{bmatrix}$$

$$\begin{bmatrix} 400 & -201 \\ -800 & 401 \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \end{bmatrix} = \begin{bmatrix} 200 \\ -200 \end{bmatrix} \quad \begin{bmatrix} \alpha_1 \\ \alpha_2 \end{bmatrix} = \begin{bmatrix} -100 \\ -200 \end{bmatrix}$$

$$\begin{bmatrix} 401 & -201 \\ -800 & 401 \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \end{bmatrix} = \begin{bmatrix} 200 \\ -200 \end{bmatrix} \quad \begin{bmatrix} \alpha_1 \\ \alpha_2 \end{bmatrix} = \begin{bmatrix} 40000 \\ 79800 \end{bmatrix}$$

最优化估计方法

❑ 最小二乘法(Least Squares Method)

– 梯度下降法

$$\min_{\boldsymbol{\alpha}} J(\boldsymbol{\alpha}) = \|\mathrm{A}\boldsymbol{\alpha} - \boldsymbol{y}\|_2^2$$

For j in range(epochs)

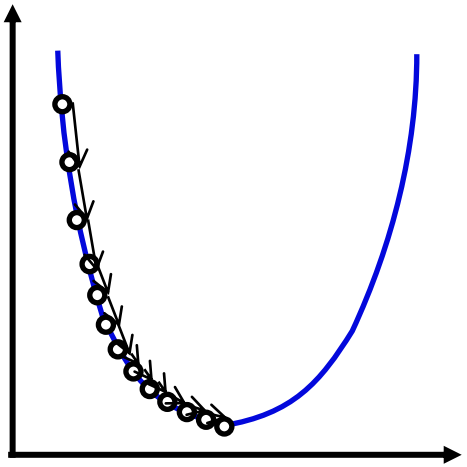
For (x_k, y_k) in training set

$$\boldsymbol{\alpha} = \boldsymbol{\alpha} - \lambda \frac{\partial}{\partial \boldsymbol{\alpha}} \boldsymbol{J}(\boldsymbol{\alpha})$$

End

End

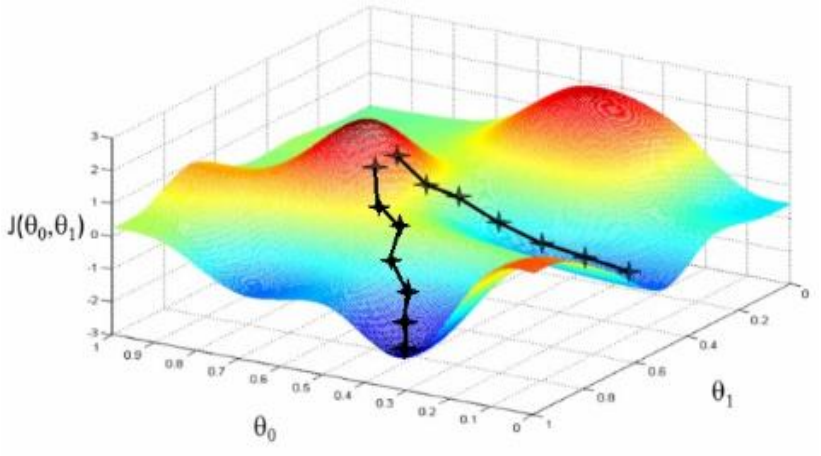
学习率 Learning rate

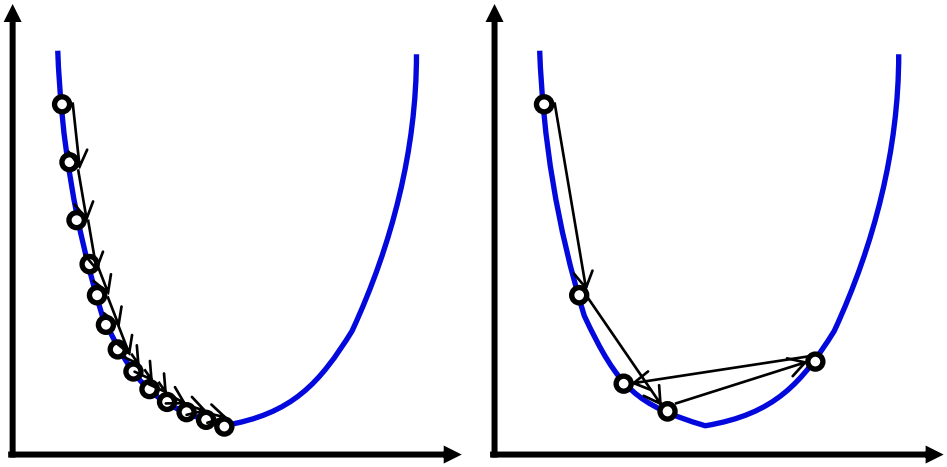


最优化估计方法

❑ 最小二乘法(Least Squares Method)

- 初始值会导致梯度下降法得到不同的局部最优解
  - 随机初始值
  - 已有网络参数，fine-tune
- 学习率影响收敛速度，甚至是否收敛





2 机器学习基础

机器学习基本概念

❑ 机器学习定义

– 让计算机具有像人一样的学习和思考能力的技术的总称。具体来说是从已知数据中获得规律，并利用规律对未知数据进行预测的技术

❑ 机器学习分类

– 有监督学习（Supervised Learning）：有老师（环境）的情况下，学生（计算机）从老师（环境）那里获得对错指示、最终答案的学习方法。跟学师评

– 无监督学习（ Unsupervised Learning）：没有老师（环境）的情况下，学生（计算机）自学的过程，一般使用一些既定标准进行评价。自学标评

– 强化学习（Reinforcement Learning）：没有老师（环境）的情况下，学生（计算机）对问题答案进行自我评价的方法。自学自评

数据集

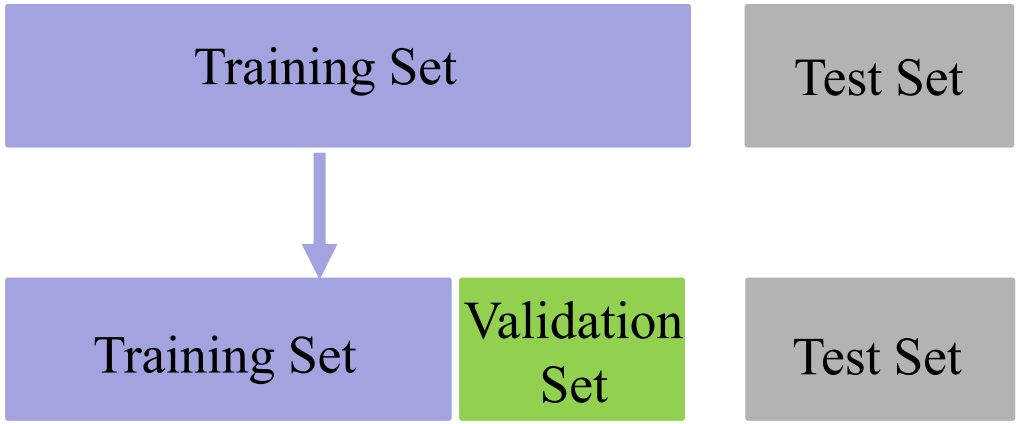
❑ 数据集：观测样本的集合

- 具体地，$D=\{\boldsymbol{x}_1,\boldsymbol{x}_2,\cdots,\boldsymbol{x}_n\}$表示一个包含$n$个样本的数据集，其中，$\boldsymbol{x}_i$是一个向量，表示数据集的第$i$个样本，其维度$d$称为样本空间的维度
- 向量$\boldsymbol{x}_i$的元素称为样本的特征，其取值可以是连续的，也可以是离散的
- 从数据集中学出模型的过程，便称为“学习”或“训练”

数据集

❑ 数据集分类

- 训练集(Training set)：用于模型拟合的数据样本
- 验证集(Validation set)：是模型训练过程中单独留出的样本集，它可以用于调整模型的超参数和用于对模型的能力进行初步评估
  - 例如SVM中参数c（控制分类错误的惩罚程度）和核函数的选择，或者选择网络结构
- 测试集(Test set)：用来评估模最终模型的泛化能力。但不能作为调参、选择特征等算法相关的选择的依据

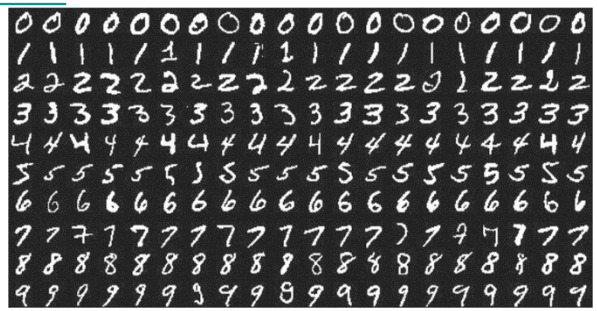


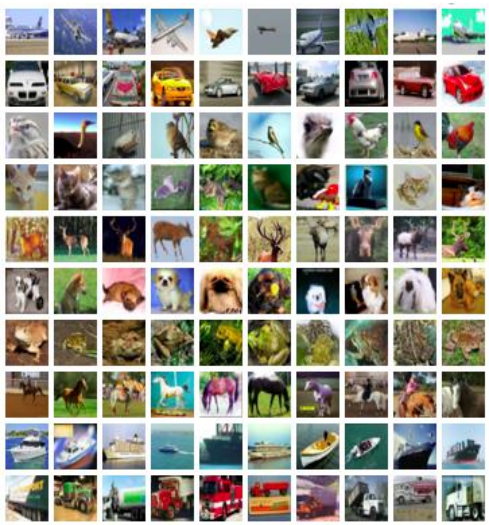
通常可以选择训练集、验证集和测试集数据比例为6:2:2

数据集

❑ 常见数据集

- 图像分类
  - MNIST(手写数字) http://yann.lecun.com/exdb/mnist/
  - CIFAR-10, CIFAR-100, ImageNet
    - https://www.cs.toronto.edu/~kriz/cifar.html
    - http://www.image-net.org/





数据集

❑ 常见数据集

– 电影评论情感分类

- Large Movie Review Dataset v1.0
  - http://ai.stanford.edu/~amaas/data/sentiment/

– 图像生成诗歌

- 数据集： https://github.com/researchmm/img2poem

come on down to my boat baby
come on down where we can play
come on down to my boat baby
come on down we'll sail away

my walls outside must have some flowers
my walls within must have some books
a house that's small a garden large
and in it leafy nooks

误差分析

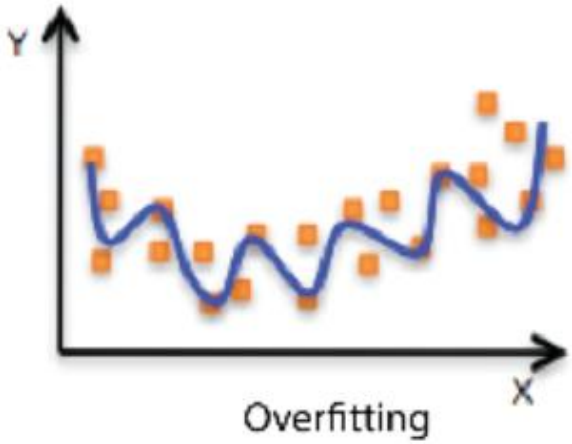
❑ 误差

- 算法实际预测输出与样本真实输出之间的差异
  - 模型在训练集上的误差称为“训练误差”
  - 模型在总体样本上的误差称为“泛化误差”
  - 模型在测试集上的误差称为“测试误差”
- 由于我们无法知道总体样本会，所以我们只能尽量最小化训练误差，导致训练误差和泛化误差有可能存在明显差异

误差分析

❑ 过拟合

- 是指模型能很好地拟合训练样本，而无法很好地拟合测试样本的现象，从而导致泛化性能下降
- 为防止“过拟合”，可以选择减少参数、降低模型复杂度、正则化等

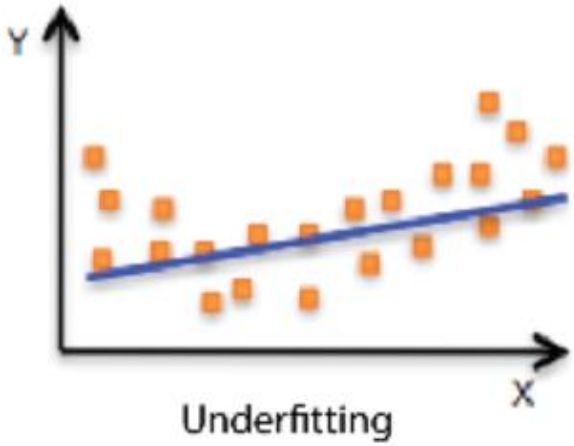


训练集误差小，测试集误差大

误差分析

❑ 欠拟合

- 是指模型还没有很好地训练出数据的一般规律，模型拟合程度不高的现象
- 为防止“欠拟合”，可以选择调整参数、增加迭代深度、换用更加复杂的模型等

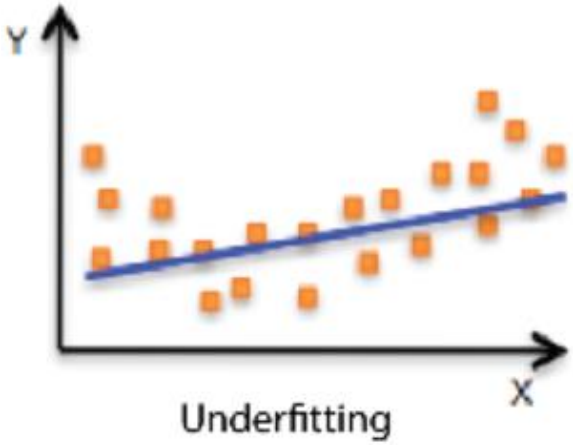


训练集和测试集误差都比较大

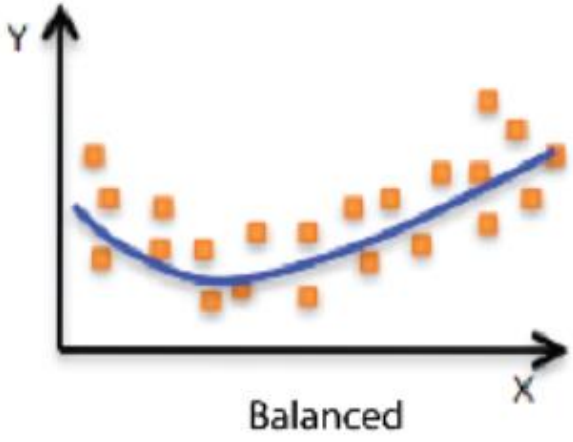
误差分析

❑ 模型训练的状态

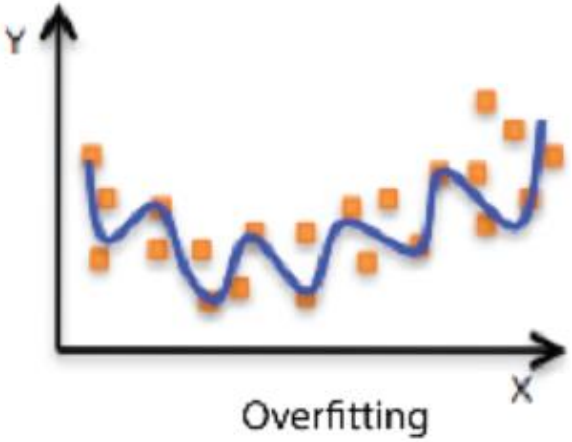
– 欠拟合、合适和过拟合的模型



训练集和测试集误差都比较大



训练集和测试集误差都比较小



训练集误差小
测试集误差大

误差分析

❑ 泛化误差分析

– 假设数据集上需要预测的样本为Y，特征为X，潜在模型为Y=f(X)+ε，其中ε~N(0,σ_ε)是噪声，估计的模型为$\hat{f}$(X)

$$Err(\hat{f}) = \mathrm{E}\left[\left(Y - \hat{f}(\mathrm{X})\right)^2\right]$$

⇩

$$Err(\hat{f}) = \mathrm{Bias}^2(\hat{f}) + \mathrm{Var}(\hat{f}) + \sigma_\varepsilon^2$$

泛化误差可分解为：偏差+方差

误差分析

❑ 泛化误差分析

– 假设数据集上需要预测的样本为Y，特征为X，潜在模型为Y=f(X)+ε，其中ε~N(0,σ_ε)是噪声，估计的模型为$\hat{f}$(X)

$$Err(\hat{f}) = \mathrm{E}\left[\left(Y - \hat{f}(\mathrm{X})\right)^2\right]$$

$$Err(\hat{f}) = \mathrm{E}\left[\left(f(X) + \varepsilon - \hat{f}(\mathrm{X})\right)^2\right]$$

$$Err(\hat{f}) = \mathrm{E}\left[\left(f(X) - \hat{f}(\mathrm{X})\right)^2 + 2\varepsilon\left(f(X) - \hat{f}(\mathrm{X})\right) + \varepsilon^2\right]$$

$$Err(\hat{f}) = \mathrm{E}\left[\left(E\left(\hat{f}(\mathrm{X})\right) - f(X) + \hat{f}(\mathrm{X}) - E\left(\hat{f}(\mathrm{X})\right)\right)^2\right] + \sigma_\varepsilon^2$$

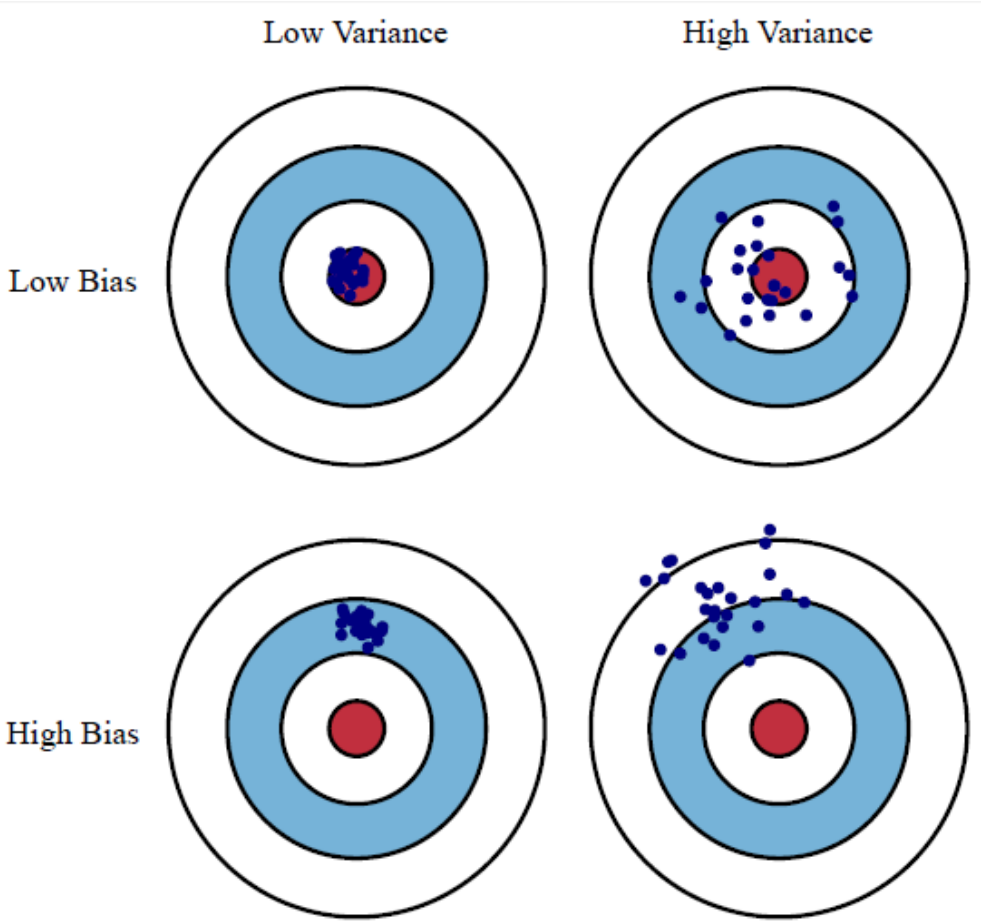
$$Err(\hat{f}) = \mathrm{E}\left[\left(E\left(\hat{f}(\mathrm{X})\right) - f(X)\right)\right]^2 + \mathrm{E}\left[\left(\hat{f}(\mathrm{X}) - E\left(\hat{f}(\mathrm{X})\right)\right)^2\right] + \sigma_\varepsilon^2$$

$$Err(\hat{f}) = \mathrm{Bias}^2(\hat{f}) + \mathrm{Var}(\hat{f}) + \sigma_\varepsilon^2$$

误差分析

❑ 泛化误差分析

- 偏差（bias）反映了模型在样本上的期望输出与真实标记之间的差距，即模型本身的精准度，反映的是模型本身的拟合能力
- 方差（variance）反映了模型在不同训练数据集下学得的函数的输出与期望输出之间的误差，即模型的稳定性，反应的是模型的波动情况



误差分析

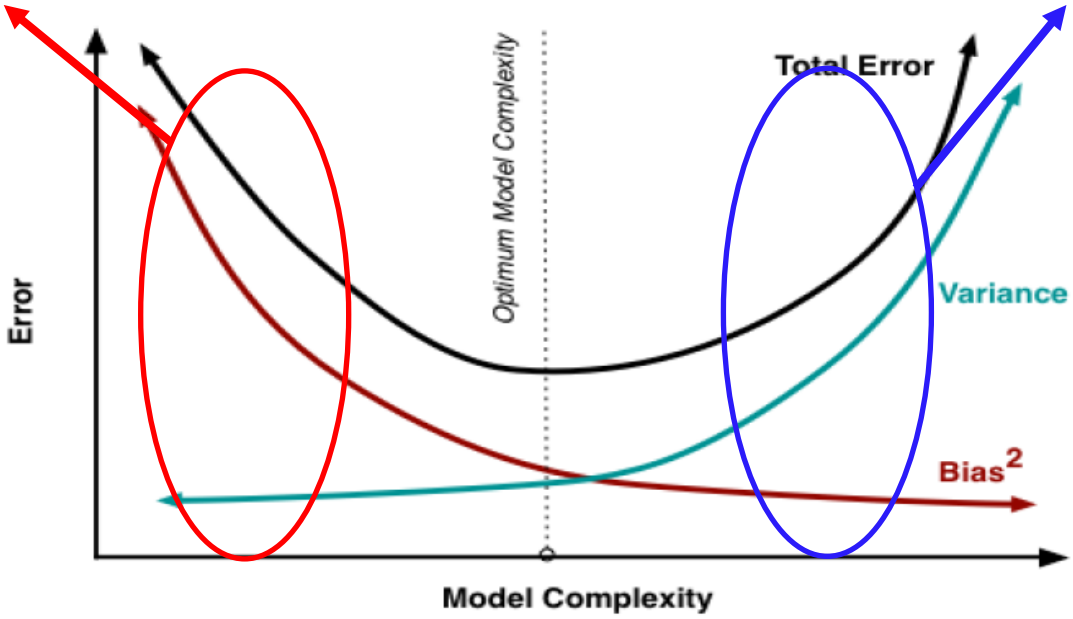
❑ 泛化误差分析

欠拟合：高偏差低方差

- 寻找更好的特征，提升对数据的刻画能力
- 增加特征数量
- 重新选择更加复杂的模型

过拟合：低偏差高方差

- 增加训练样本数量
- 减少特征维数，高维空间密度小
- 加入正则化项，使得模型更加平滑

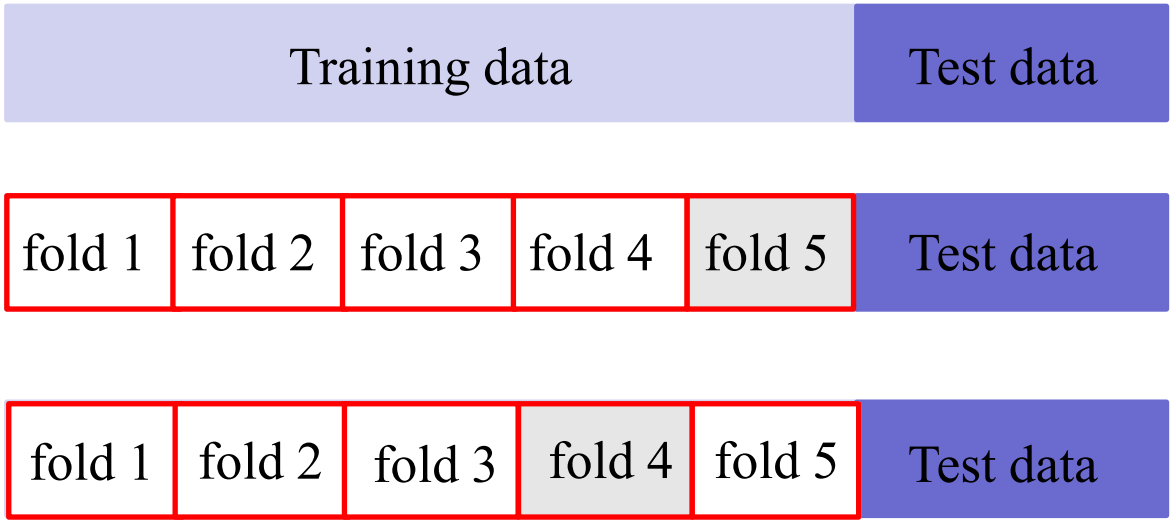


误差分析

❑ 交叉验证(Cross Validation)

– 基本思路：将训练集划分为K份，每次采用其中K-1份作为训练集，另外一份作为验证集，在训练集上学得函数后，然后在验证集上计算误差---K折交叉验证

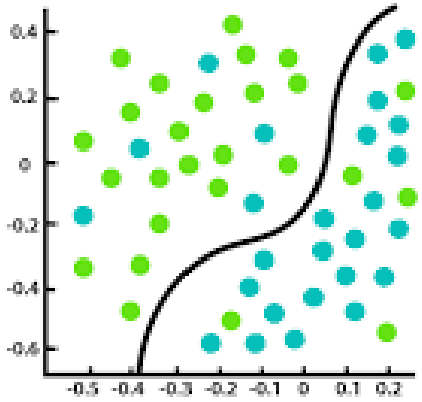
- K折重复多次，每次重复中产生不同的分割
- 留一交叉验证(Leave-One-Out)



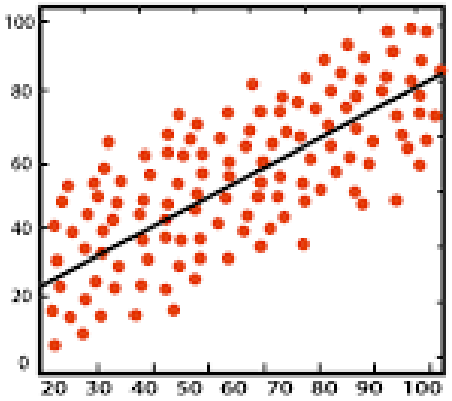
代表性机器学习方法

❑ 机器学习分类

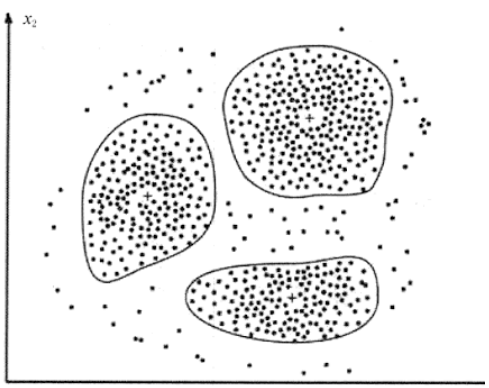
- 有监督学习：代表任务“分类”和“回归”
- 无监督学习：代表任务“聚类”和“降维”



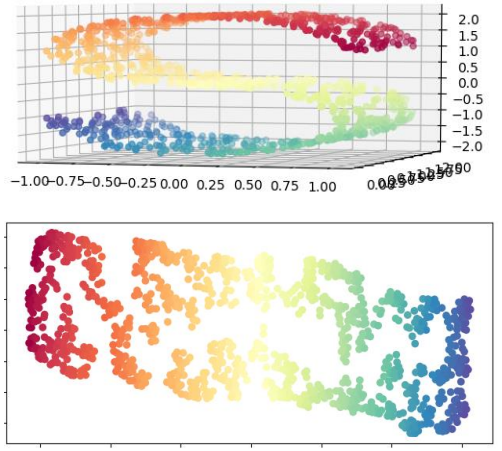
分类



回归



聚类



降维

代表性机器学习方法

❑ 有监督学习

- 数据集有标记（答案）
- 数据集通常扩展为(x_i,y_i)，其中$y_i \in \mathrm{Y}$是x_i的标记，Y是所有标记的集合，称为“标记空间”或“输出空间”
- 监督学习的任务是训练出一个模型用于预测y的取值，根据$D=\{(x_1,y_1),(x_2,y_2),\cdots,(x_n,y_n)\}$，训练出函数$f$，使得$f(x) \cong y$
- 若预测的值是离散值，如年龄，此类学习任务称为“分类”
- 若预测的值是连续值，如房价，此类学习任务称为“回归”

代表性机器学习方法

❑ 无监督学习

- 数据集没有标记信息（自学）
- 聚类：我们可以使用无监督学习来预测各样本之间的关联度，把关联度大的样本划为同一类，关联度小的样本划为不同类，这便是“聚类”
- 降维：我们也可以使用无监督学习处理数据，把维度较高、计算复杂的数据，转化为维度低、易处理、且蕴含的信息不丢失或较少丢失的数据，这便是“降维”

有监督学习-线性回归

❑ 线性回归

- 在样本属性和标签中找到一个线性关系的方法
- 根据训练数据找到一个线性模型，使得模型产生的预测值与样本标签的差距最小
- 若用x_i^k表示第k个样本的第i个属性，则线性模型一般形式为：

$$f(\boldsymbol{x}^k) = w_1 x_1^k + w_2 x_2^k + \cdots + w_m x_m^k + b = \sum_{i=1}^{m} w_i x_i^k + b$$

- 线性回归学习的对象就是权重向量w和偏置向量b。如果用最小均方误差来衡量预测值与样本标签的差距，那么线性回归学习的目标可以表示为：

$$(\boldsymbol{w}^*, b^*) = \underset{(\boldsymbol{w},b)}{\operatorname{argmin}} \sum_{k=1}^{n} \left(f(\boldsymbol{x}^k) - \boldsymbol{y}^k\right)^2 = \underset{(\boldsymbol{w},b)}{\operatorname{argmin}} \sum_{i=1}^{n} \left(\boldsymbol{w}^T \boldsymbol{x}^k + b - \boldsymbol{y}^k\right)^2$$

有监督学习-逻辑回归

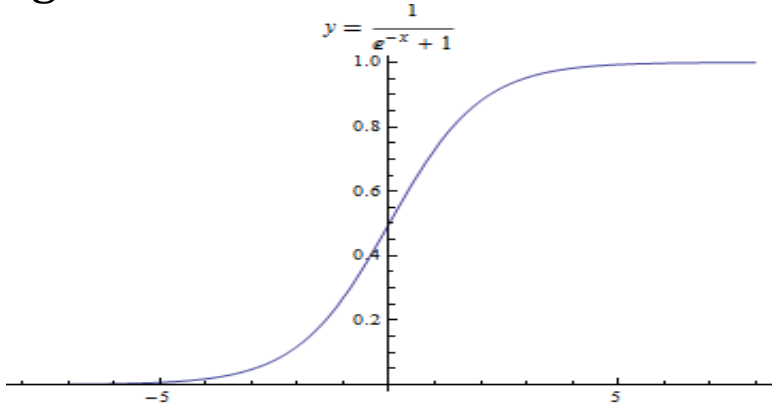
❑ 逻辑回归(Logistic regression)

– 利用sigmoid函数，将线性回归产生的预测值压缩到0和1之间。此时将y视作样本为正例的可能性，即:

$$g\left(f\left(\boldsymbol{x}^k\right)\right)=\begin{cases}1, & \dfrac{1}{1+e^{-\left(\boldsymbol{w}^T\boldsymbol{x}^k+b\right)}}\geq 0.5\\ 0, & otherwise\end{cases}$$

注意，逻辑回归本质上属于分类算法, sigmoid函数的具体表达形式为:

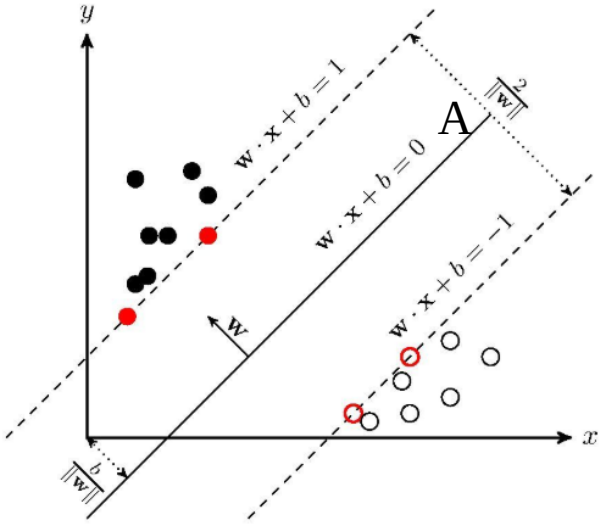
$$g(x)=\frac{1}{1+e^{-x}}$$



有监督学习-SVM

❑ 支持向量机（Support Vector Machine, SVM）

- 是有监督学习中最具有影响力的方法之一，是基于线性判别函数的一种模型
- SVM基本思想：对于线性可分的数据，能将训练样本划分开的超平面有很多，于是我们寻找“位于两类训练样本正中心的超平面”，即margin最大化。从直观上看，这种划分对训练样本局部扰动的承受性最好。事实上，这种划分的性能也表现较好



SVM会选择直线A作为分界面！

有监督学习-SVM

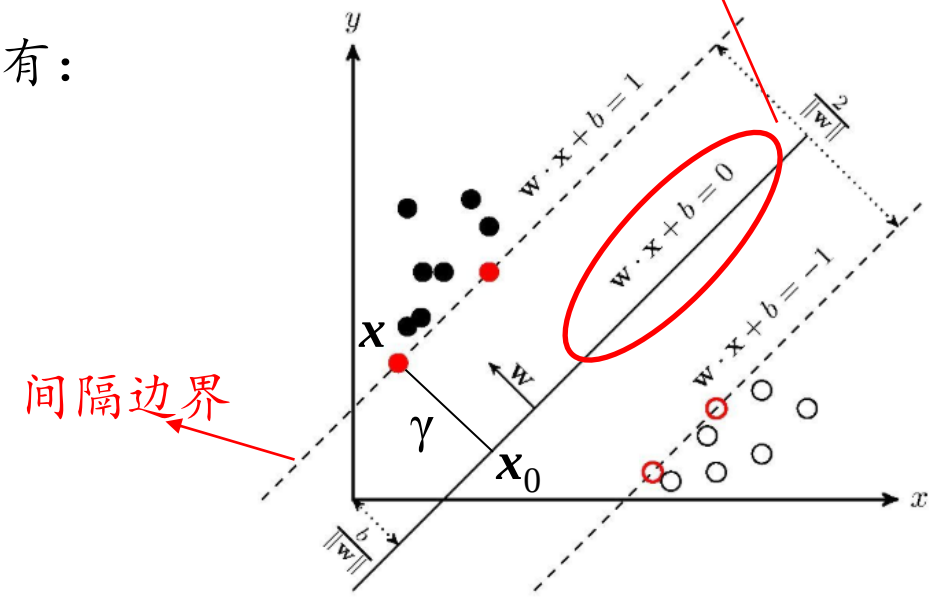
❑ 以线性分类为例

- 二类可分数据集$D=\{(x_1,y_1),(x_2,y_2),\cdots,(x_n,y_n)\}$，其中$y$=1和$y$= -1分别表示两类样本
- 定义分类的超平面$f(\boldsymbol{x})=\boldsymbol{w}^T\boldsymbol{x}+b$，决策边界(decision boundary)
- “最合适”的分类标准：超平面距离两边数据的间隔最大

样本$\boldsymbol{x}$到超平面的距离为γ，那么有：

$$\boldsymbol{x}=\boldsymbol{x}_0+\gamma\frac{\boldsymbol{w}}{\|\boldsymbol{w}\|}$$

$$\gamma=\frac{\boldsymbol{w}^T\boldsymbol{x}+b}{\|\boldsymbol{w}\|}=\frac{f(\boldsymbol{x})}{\|\boldsymbol{w}\|}$$



有监督学习-SVM

❑ 以线性分类为例

– 目标函数

$$\arg\max_{\boldsymbol{w},b} \operatorname{argmin}_{\boldsymbol{x}_i \in D} \frac{|\boldsymbol{w}^T \boldsymbol{x}_i + b|}{\sqrt{\sum_{i=1}^{d} \boldsymbol{w}_i^2}} \qquad s.t.\ \forall \boldsymbol{x}_i \in D:\ y_i(\boldsymbol{w}^T \boldsymbol{x}_i + b) \geq 0$$

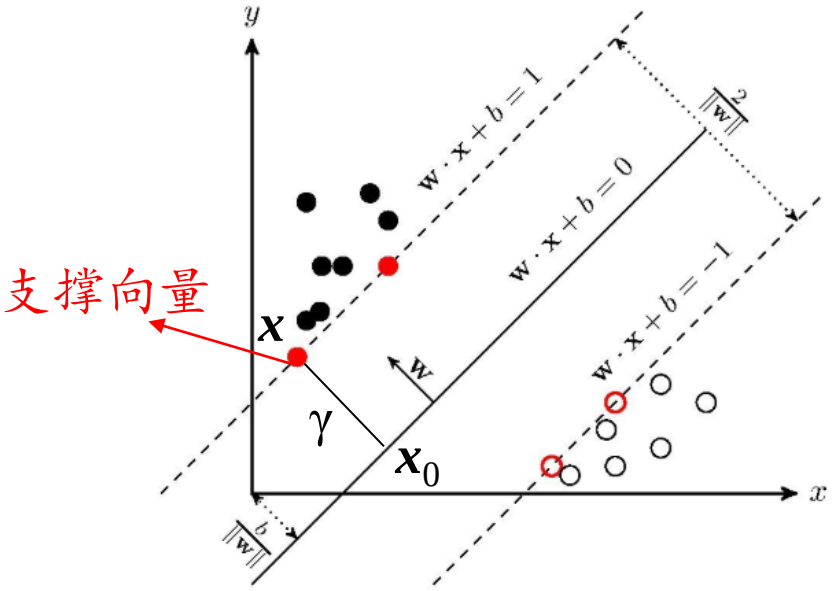
通常为方便优化，我们选择加强约束条件：

$$\forall \boldsymbol{x}_i \in D:\ |\boldsymbol{w}^T \boldsymbol{x}_i + b| \geq 1$$

那么，原问题可以近似为：

$$\operatorname{argmin}_{\boldsymbol{w},b} \frac{1}{2} \sum_{i=1}^{d} \boldsymbol{w}_i^2$$

$$\forall \boldsymbol{x}_i \in D:\ |\boldsymbol{x}_i \boldsymbol{w} + b| \geq 1$$



有监督学习-SVM

❑ 线性不可分

- 特征空间存在超曲面（hypersurface）将正类和负类分开
- 核函数(kernel function)
  - 使用非线性函数将非线性可分问题从原始的特征空间映射至更高维
  - 决策边界的超平面表示为： $\boldsymbol{w}^T\emptyset(\boldsymbol{x})+b=0$
  - 定义映射函数的内积为核函数

$$\boldsymbol{K}(\boldsymbol{x_i},\boldsymbol{xj})=\emptyset(\boldsymbol{x_i})^T\emptyset(\boldsymbol{x_j})$$

  - 常见核函数

名称	解析式
多项式核(polynomial kernel)	$\boldsymbol{K}(\boldsymbol{x_i},\boldsymbol{xj})=(\boldsymbol{x_i}^T\boldsymbol{x_j})^n$
径向基函数核(RBF kernel)	$\boldsymbol{K}(\boldsymbol{x_i},\boldsymbol{xj})=exp\left(-\frac{\|\boldsymbol{x_i}-\boldsymbol{x_j}\|^2}{2\sigma^2}\right)$
拉普拉斯核(Laplacian kernel)	$\boldsymbol{K}(\boldsymbol{x_i},\boldsymbol{xj})=exp\left(-\frac{\|\boldsymbol{x_i}-\boldsymbol{x_j}\|}{2\sigma^2}\right)$
Sigmoid核(Sigmoid kernel)	$\boldsymbol{K}(\boldsymbol{x_i},\boldsymbol{xj})=tanh[a(\boldsymbol{x_i}^T\boldsymbol{x_j})-b],\ a,b>0$

有监督学习-决策树

❑ 决策树(Decision Tree)

– 是一种基于树结构进行决策的机器学习方法，这恰是人类面临决策时一种很自然的处理机制

- 在这些树的结构里，叶子节点给出类标而内部节点代表某个属性
- 例如，银行在面对是否借贷给客户的问题时，通常会进行一系列的决策。银行会首先判断：客户的信贷声誉是否良好？良好的话，再判断客户是否有稳定的工作？ 不良好的话，可能直接拒绝，也可能判断客户是否有可抵押物？……这种思考过程便是决策树的生成过程

– 下面，我们以一个实例，来介绍决策树中的ID3（Iterative Dichotomiser 3）算法

有监督学习算法-决策树

如何根据该数据集，判断出银行借贷的准则？

ID	年龄	有工作	有自己的房子	信贷情况	类别
1	青年	否	否	一般	否
2	青年	否	否	好	否
3	青年	是	否	好	是
4	青年	是	是	一般	是
5	青年	否	否	一般	否
6	中年	否	否	一般	否
7	中年	否	否	好	否
8	中年	是	是	好	是
9	中年	否	是	非常好	是
10	中年	否	是	非常好	是
11	老年	否	是	非常好	是
12	老年	否	是	好	是
13	老年	是	否	好	是
14	老年	是	否	非常好	是
15	老年	否	否	一般	否

有监督学习算法-决策树

- 决策树的生成过程中，最重要的因素便是根节点的选择，即选择哪种特征作为决策因素
  - ID3算法使用信息增益作为准则

$$G(Y, X) = H(Y)-H(Y|X)$$

ID	年龄	有工作	有自己的房子	信贷情况	类别
1	青年	否	否	一般	否
2	青年	否	否	好	否
3	青年	是	否	好	是
4	青年	是	是	一般	是
5	青年	否	否	一般	否
6	中年	否	否	一般	否
7	中年	否	否	好	否
8	中年	是	是	好	是
9	中年	否	是	非常好	是
10	中年	否	是	非常好	是
11	老年	否	是	非常好	是
12	老年	否	是	好	是
13	老年	是	否	好	是
14	老年	是	否	非常好	是
15	老年	否	否	一般	否

有监督学习算法-决策树

- 决策树的生成过程中，最重要的因素便是根节点的选择，即选择哪种特征作为决策因素
  - ID3算法使用信息增益作为准则
  - 示例中样本集合D中有15个样本，9借贷样本，6个不借贷样本，那么样本集合的熵，或者说不确定性是

$$H(\mathrm{D}) = -\sum_{k=1}^{K} \frac{|\mathrm{C}_k|}{|\mathrm{D}|} \log_2 \frac{|\mathrm{C}_k|}{|\mathrm{D}|}$$

$$H(\mathrm{D}) = -\frac{9}{15}\log_2\frac{9}{15} - \frac{6}{15}\log_2\frac{6}{15} = 0.971$$

  - 属性集合{年龄，有工作，有房子，借贷情况}

ID	年龄	有工作	有自己的房子	信贷情况	类别
1	青年	否	否	一般	否
2	青年	否	否	好	否
3	青年	是	否	好	是
4	青年	是	是	一般	是
5	青年	否	否	一般	否
6	中年	否	否	一般	否
7	中年	否	否	好	否
8	中年	是	是	好	是
9	中年	否	是	非常好	是
10	中年	否	是	非常好	是
11	老年	否	是	非常好	是
12	老年	否	是	好	是
13	老年	是	否	好	是
14	老年	是	否	非常好	是
15	老年	否	否	一般	否

有监督学习算法-决策树

- 决策树的生成过程中，最重要的因素便是根节点的选择，即选择哪种特征作为决策因素
  - ID3算法使用信息增益作为准则
  - 信息增益表示得知属性 A_i 的信息后而使得样本集合不确定度减少的程度，那么需要计算得知A_i后样本集合的熵，即条件熵

$$H(D|A_i)=\sum_{i=1}^{n}\frac{|D_i|}{|D|}H(D_i)=-\sum_{i=1}^{n}\frac{|D_i|}{|D|}\sum_{k=1}^{K}\frac{|D_{ik}|}{|D_i|}\log_2\frac{|D_{ik}|}{|D_i|}=\sum_{i=1}^{n}\frac{|D_i|}{|D|}H(Di)$$

例如：A_1是年龄属性

ID	年龄	有工作	有自己的房子	信贷情况	类别
1	青年	否	否	一般	否
2	青年	否	否	好	否
3	青年	是	否	好	是
4	青年	是	是	一般	是
5	青年	否	否	一般	否
6	中年	否	否	一般	否
7	中年	否	否	好	否
8	中年	是	是	好	是
9	中年	否	是	非常好	是
10	中年	否	是	非常好	是
11	老年	否	是	非常好	是
12	老年	否	是	好	是
13	老年	是	否	好	是
14	老年	是	否	非常好	是
15	老年	否	否	一般	否

$$H(D|A_i)=\frac{5}{15}H(D_1)+\frac{5}{15}H(D_2)+\frac{5}{15}H(D_3)=0.888$$

$$H(D_1)=-\frac{2}{5}log_2\frac{2}{5}-\frac{3}{5}log_2\frac{3}{5}=0.9710$$

$$H(D_2)=-\frac{3}{5}log_2\frac{3}{5}-\frac{2}{5}log_2\frac{2}{5}=0.9710$$

$$H(D_3)=-\frac{4}{5}log_2\frac{2}{5}-\frac{1}{5}log_2\frac{1}{5}=0.7219$$

有监督学习算法-决策树

- 决策树的生成过程中，最重要的因素便是根节点的选择，即选择哪种特征作为决策因素
  - ID3算法使用信息增益作为准则
  - 信息增益

$$g(\mathrm{D},\mathrm{A}_i) = H(\mathrm{D}) - H(\mathrm{D}|\mathrm{A}_i)=0.083$$

$$g(\mathrm{D},\mathrm{A}_2) =0.324 \qquad g(\mathrm{D},\mathrm{A}_3) =0.420 \qquad g(\mathrm{D},\mathrm{A}_4) =0.363$$

➢ 由于特征A_3（有房子）的信息增益最大，所以选择A_3作为最优特征

有监督学习算法-决策树

- A_3取值为“是”的情况下，均可贷款，无需再细分，为叶结点；
- 将A_3取值为“否”的样本提取出，构成子数据集D_2。

ID	年龄	有工作	有自己的房子	信贷情况	类别
1	青年	否	否	一般	否
2	青年	否	否	好	否
3	青年	是	否	好	是
4	青年	是	是	一般	是
5	青年	否	否	一般	否
6	中年	否	否	一般	否
7	中年	否	否	好	否
8	中年	是	是	好	是
9	中年	否	是	非常好	是
10	中年	否	是	非常好	是
11	老年	否	是	非常好	是
12	老年	否	是	好	是
13	老年	是	否	好	是
14	老年	是	否	非常好	是
15	老年	否	否	一般	否

有监督学习算法-决策树

❑ 进一步地，$H(\mathrm{D}_2) = -\frac{3}{9}\log_2\frac{3}{9} - \frac{6}{9}\log_2\frac{6}{9} = 0.918$

$$g(\mathrm{D}_2, \mathrm{A}_1) = H(\mathrm{D}_2) - H(\mathrm{D}_2|\mathrm{A}_1)=0.251$$

$$g(\mathrm{D}_2, \mathrm{A}_2) = H(\mathrm{D}_2) - H(\mathrm{D}_2|\mathrm{A}_2)=0.918$$

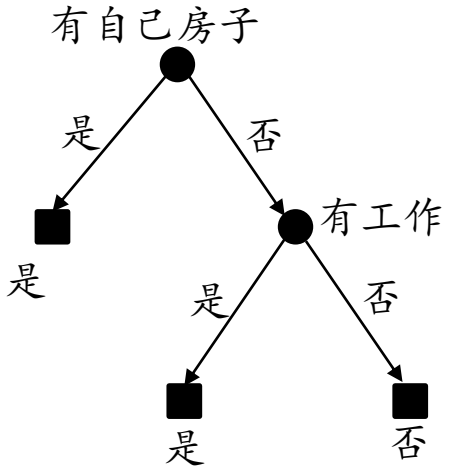
$$g(\mathrm{D}_2, \mathrm{A}_4) = H(\mathrm{D}_2) - H(\mathrm{D}_2|\mathrm{A}_4)=0.474$$

- 由于特征A_2（有工作）的信息增益最大，所以选择A_2作为最优特征；
- 又因为该特征划分的两个样本集合都属于同一个类别，所以算法结束。

ID	年龄	有工作	有自己的房子	信贷情况	类别
1	青年	否	否	一般	否
2	青年	否	否	好	否
3	青年	是	否	好	是
4	青年	是	是	一般	是
5	青年	否	否	一般	否
6	中年	否	否	一般	否
7	中年	否	否	好	否
8	中年	是	是	好	是
9	中年	否	是	非常好	是
10	中年	否	是	非常好	是
11	老年	否	是	非常好	是
12	老年	否	是	好	是
13	老年	是	否	好	是
14	老年	是	否	非常好	是
15	老年	否	否	一般	否

有监督学习算法-决策树

- 显然，决策树的生成是一个递归过程，有三种情况会导致递归返回：
  - 当前结点包含的样本属于同一类别
  - 当前属性集为空，或所有样本在所有属性取值相同
  - 当前结点包含的集合为空



银行借贷准则对应的决策树！

有监督学习算法-随机森林

❑ 随机森林(Random Forest)

- 集成学习(Ensemble learning)
  - 组合多个弱监督模型以期得到一个更好更全面的强监督模型，集成学习潜在的思想是即便某一个弱分类器得到了错误的预测，其他的弱分类器也可以将错误纠正回来
- 该算法用随机的方式建立起一棵棵决策树，然后由这些决策树组成一个森林，其中每棵决策树之间没有关联，当有一个新的样本输入时，就让每棵树独立的做出判断，按照多数原则决定该样本的分类结果

有监督学习算法-随机森林

❑ 随机森林构建的基本步骤

- 随机有放回地从训练集中的抽取m 个训练样本,训练集D_t
- 从D_t对应的特征属性中随机选择部分特征，构建决策树
- 重复上述步骤构建多个决策树

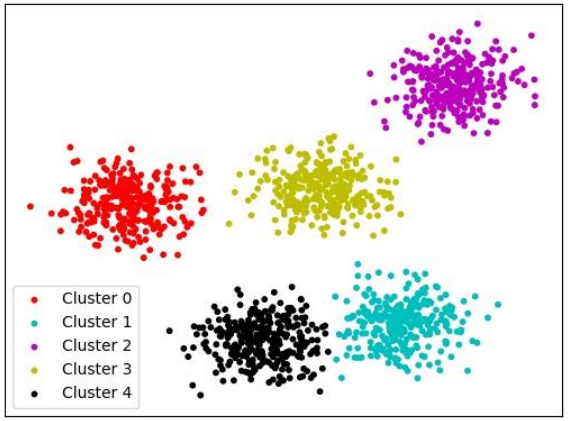
❑ 预测步骤

- 向建立好的随机森林中输入一个新样本
- 随机森林中的每棵决策树都独立的做出判断
- 将得到票数最多的分类结果作为该样本最终的类别

无监督学习方法-聚类

❑ 聚类

- 目的：将数据分成多个类别，在同一个类内，对象（实体）之间具有较高的相似性，在不同类内，对象之间具有较大的差异
- 对一批没有类别标签的样本集，按照样本之间的相似程度分类，相似的归为一类，不相似的归为其它类。这种分类称为聚类分析，也称为无监督分类
- 常见方法
  - K-Means聚类、均值漂移聚类、基于密度的聚类等



无监督学习方法-聚类

- **K-means聚类是一个反复迭代的过程，算法分为四个步骤：**
  - 1) 选取数据空间中的K个对象作为初始中心，每个对象代表一个聚类中心；
  - 2) 对于样本中的数据对象，根据它们与这些聚类中心的欧氏距离，按距离最近的准则将它们分到距离它们最近的聚类中心（最相似）所对应的类；
  - 3) 更新聚类中心：将每个类别中所有对象所对应的均值作为该类别的聚类中心，计算目标函数的值；
  - 4) 判断聚类中心和目标函数的值是否发生改变，若不变，则输出结果，若改变，则返回2）

无监督学习-降维

❑ 降维

- 目的：将原始样本数据的维度d降低到一个更小的数m，且尽量使得样本蕴含信息量损失最小，或还原数据时产生的误差最小

❑ 降维的优势

- 数据在低维下更容易处理、更容易使用；
- 相关特征，特别是重要特征更能在数据中明确的显示出来；
- 如果只有二维或者三维的话，能够进行可视化展示；
- 去除数据噪声，降低算法开销等

无监督学习-降维

❑ 主成分分析法

假设我们有N个无标签的d维样本$\boldsymbol{x}_i$，记为$\mathrm{X}_{d\times n}=(\boldsymbol{x}_1,\boldsymbol{x}_2,\ldots,\boldsymbol{x}_{\mathrm{n}})$，我们想寻找一个矩阵$\mathrm{L}_{m\times d}$，那么$\mathrm{Y}=\mathrm{LX}$便是降维后的样本$\mathrm{Y}_{m\times n}=(\boldsymbol{y}_1,\boldsymbol{y}_2,\ldots,\boldsymbol{y}_{\mathrm{n}})$,我们希望新的样本Y具有最大的方差，具体步骤如下所示：

1.计算样本协方差矩阵$S_{d\times d}=\frac{1}{n}\mathrm{X}\mathrm{X}^T$;

2.对S进行特征分解;

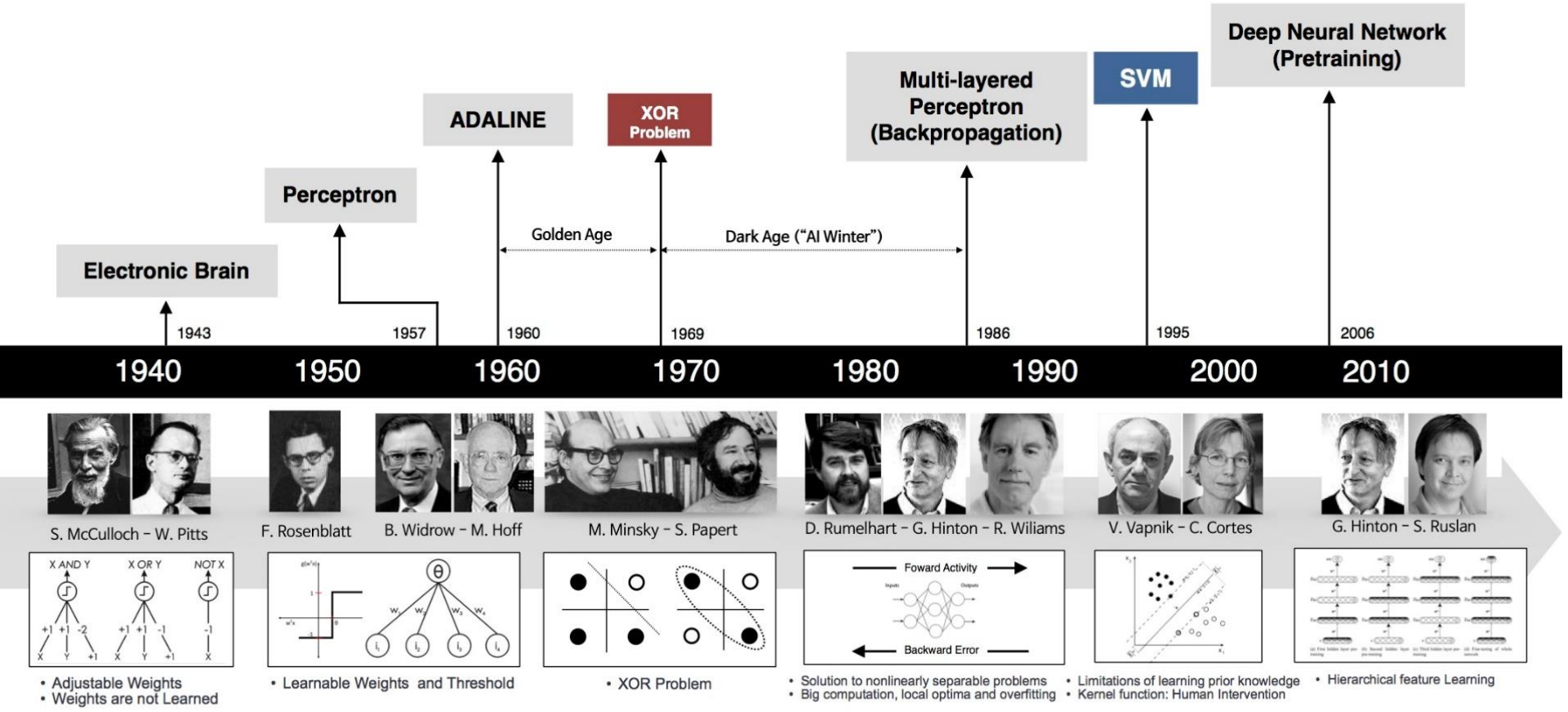
3.找出S最大的m个特征值$(\lambda_1,\lambda_2,\ldots,\lambda_m)$和相应的特征向量$(\boldsymbol{u}_1,\boldsymbol{u}_2,\ldots,\boldsymbol{u}_m)$;

4.令$\mathrm{U}_{d\times m}=(\boldsymbol{u}_1,\boldsymbol{u}_2,\ldots,\boldsymbol{u}_m)$;

5.$\mathrm{L}=\mathrm{U}^T$即为所求的变换矩阵。

3 神经元模型

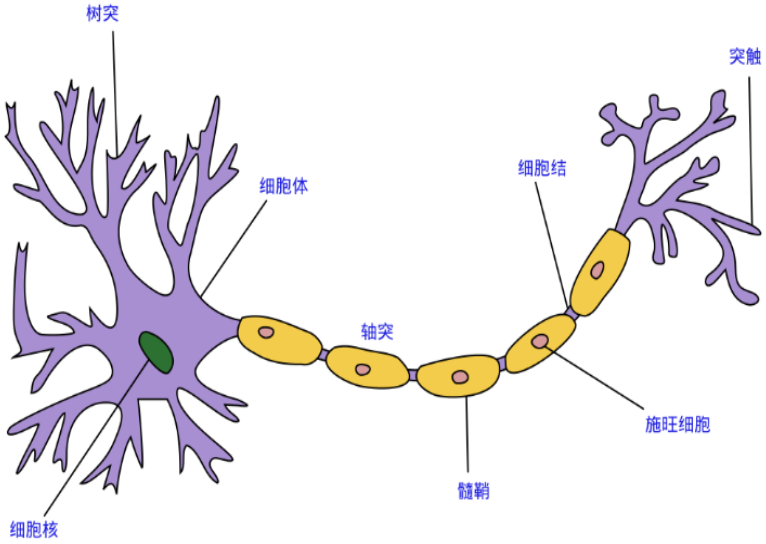
神经网络的发展史

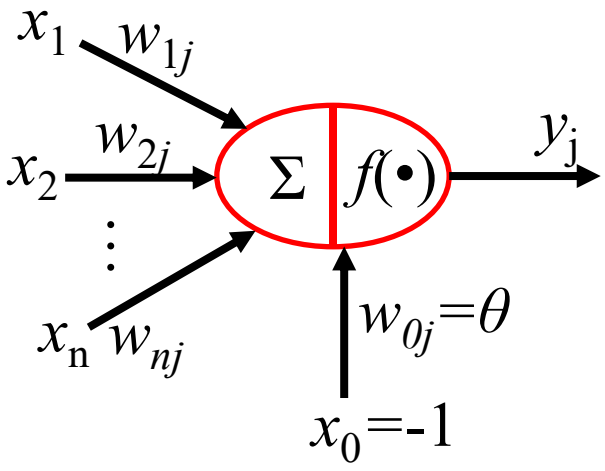

Deep Neural Network (Pretraining)
SVM
Multi-layered Perceptron (Backpropagation)
XOR Problem
ADALINE
Perceptron
Electronic Brain
Golden Age
Dark Age ("AI Winter")
1943
1957
1960
1969
1986
1995
2006
1940
1950
1960
1970
1980
1990
2000
2010
S. McCulloch - W. Pitts
F. Rosenblatt
B. Widrow - M. Hoff
M. Minsky - S. Papert
D. Rumelhart - G. Hinton - R. Wiliams
V. Vapnik - C. Cortes
G. Hinton - S. Ruslan
X AND Y
X OR Y
NOT X
Foward Activity
Backward Error
• Adjustable Weights
• Weights are not Learned
• Learnable Weights and Threshold
• XOR Problem
• Solution to nonlinearly separable problems
• Big computation, local optima and overfitting
• Limitations of learning prior knowledge
• Kernel function: Human Intervention
• Hierarchical feature Learning

神经元模型

❑ M-P模型

– 1943 年，美国神经生理学家沃伦·麦卡洛克（Warren McCulloch）和数学家沃尔特·皮茨(Walter Pitts）对生物神经元进行建模，首次提出了一种形式神经元模型，并命名为McCulloch-Pitts模型，即后来广为人知的M-P模型

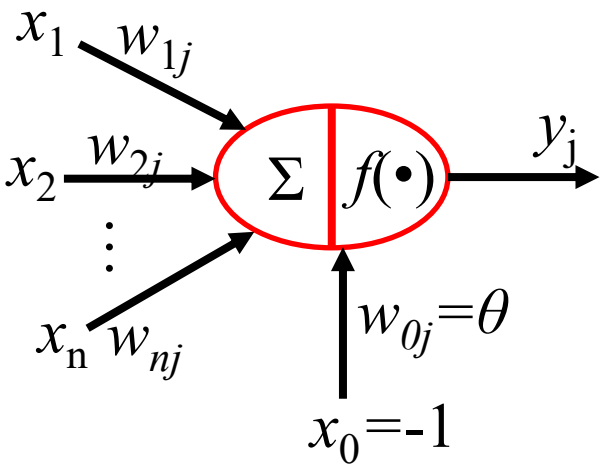




神经元模型

❑ M-P模型

– 在M-P模型中，神经元接受其他n个神经元的输入信号（0或1），这些输入信号经过权重加权并求和，将求和结果与阈值(threshold) θ 比较，然后经过激活函数处理，得到神经元的输出

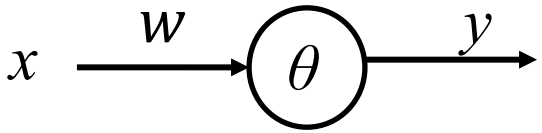


$$y = f\left(\sum_{i=1}^{n} \omega_{i\,j} x_i - \theta\right)$$

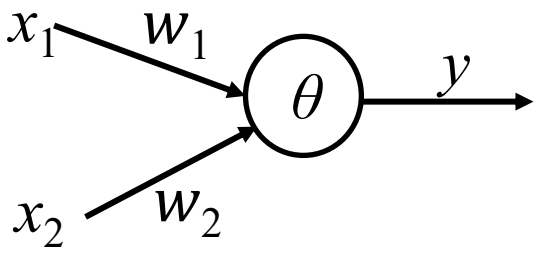
神经元模型

❑ M-P 模型可以表示多种逻辑运算

– 取反运算可以用单输入单输出模型表示，即如果输入为0则输出1，如果输入为1则输出0。由M-P模型的运算规则可得w=-2，θ=-1



– 逻辑或与逻辑与运算可以用双输入单输出模型表示。以逻辑与运算为例，w_1=1，w_2=1, θ=1.5



当时还没有通过对训练样本进行训练来确定参数的方法，上述参数只能人为事先计算后确定

4

感知器及多层感知器

感知器

❑ 感知器

– 1958 年，罗森布拉特（Roseblatt）提出了感知器，与 M-P 模型需要人为确定参数不同，感知器能够通过训练自动确定参数。训练方式为有监督学习，即需要设定训练样本和期望输出，然后调整实际输出和期望输出之差的方式（误差修正学习）

$$w_i \leftarrow w_i + \alpha(r - y)x$$

$$\theta \leftarrow \theta - \alpha(r - y)$$

其中，α是学习率，r和y分别是期望输出和实际输出。

感知器

❑ 感知器权重调整的基本思路

– 实际输出 y 与期望输出 r 相等时， w 和 θ 不变

– 实际输出 y 与期望输出 r 不相等时，调整 w 和 θ 的值

$$w_i \leftarrow w_i + \alpha(r - y)x$$

$$\theta \leftarrow \theta - \alpha(r - y)$$

实际输出y和期望输出r不相等时的调整策略

实际输出$y = 0$, 期望输出$r = 1$时（未激活）

- 减小θ
- 增大 $x_i = 1$ 的连接权重 ω_i,
- $x_i = 0$ 的连接权重不变

实际输出$y = 1$, 期望输出$r = 0$时（激活过度）

- 增大θ
- 降低 $x_i = 1$ 的连接权重 ω_i,
- $x_i = 0$ 的连接权重不变

感知器

❑ 感知器训练过程

0. 训练准备

- 准备 N 个训练样本x_i和期望输出r_i
- 初始化参数ω_i和θ

1. 调整参数

1.1. 迭代调整，直到误差为0或小于某个指定数值

1.1.1. 逐个加入训练样本，计算实际输出

：实际输出和期望输出相等时，参数不变

：实际输出和期望输出不同时，通过误差修正学习调整参数

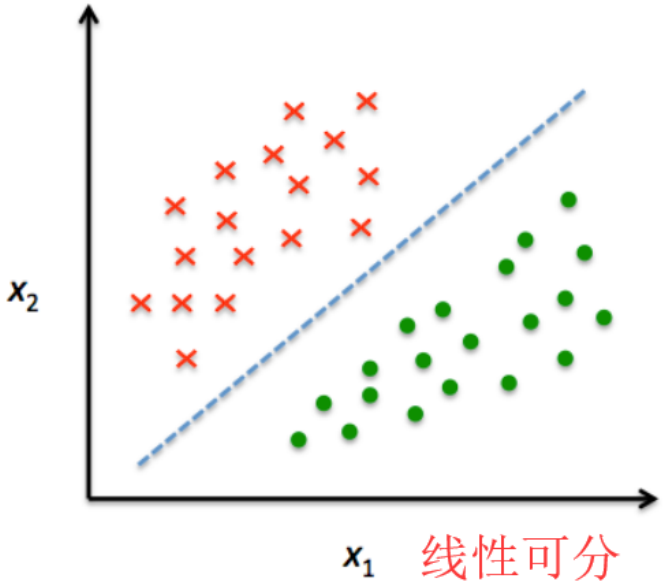
重复上述步骤 1.1.1

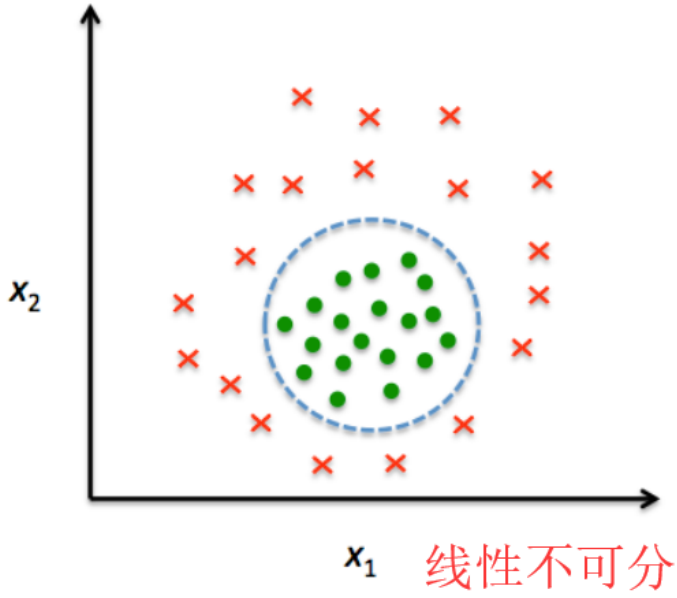
重复上述步骤1.1

多层感知器

- 单层感知器只能解决线性可分问题，而不能解决线性不可分问题，为了解决线性不可分问题，我们需要使用多层感知器

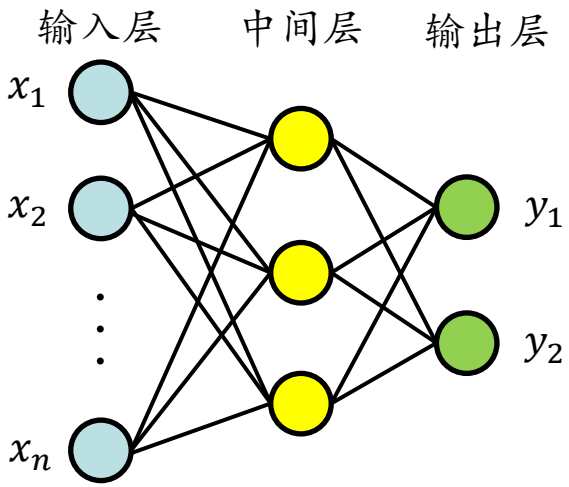
$$y = f\left(\sum_{i=1}^{n} \omega_{i\,j} x_i - \theta\right)$$





多层感知器

- 多层感知器指的是由多层结构的感知器递阶组成的输入值向前传播的网络，也被称为前馈网络或正向传播网络
  - 以三层结构的多层感知器为例，它由输入层、中间层及输出层组成
    - 与M-P 模型相同，中间层的感知器通过权重与输入层的各单元相连接，通过阈值函数计算中间层各单元的输出值
    - 中间层与输出层之间同样是通过权重相连接



5

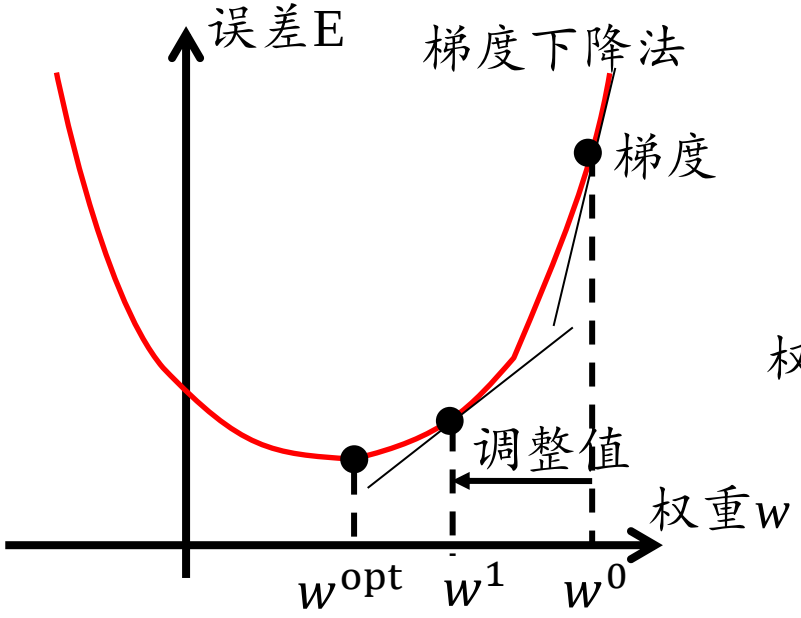
BackPropagation (BP)算法

BP算法

- 多层感知器的训练使用**误差反向传播算法**（Error Back Propagation），即**BP算法**
  - BP算法最早由沃博斯于1974年提出，鲁梅尔哈特等人进一步发展了该理论
  - 基本过程：
    - 前向传播计算：由输入层经过隐含层向输出层的计算网络输出
    - 误差反向逐层传递：网络的期望输出与实际输出之差的误差信号由输出层经过隐含层逐层向输入层传递
    - 由“前向传播计算”与“误差反向逐层传递”的反复进行的网络训练过程

BP算法

- 多层感知器的训练使用**误差反向传播算法**（Error Back Propagation），即**BP算法**
  - BP算法就是通过比较实际输出和期望输出得到误差信号，把误差信号从输出层逐层向前传播得到各层的误差信号，再通过调整各层的连接权重以减小误差。权重的调整主要使用梯度下降法



误差公式：$E = \sum_{n=1}^{N} \|\boldsymbol{r}_n - \boldsymbol{y}_n\|^2$

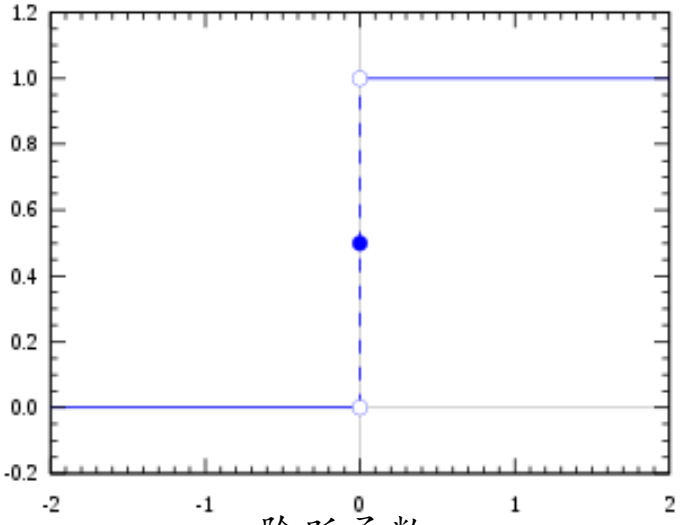
权重调整值公式：$\Delta w = -\alpha \frac{\partial E}{\partial w}$

BP算法

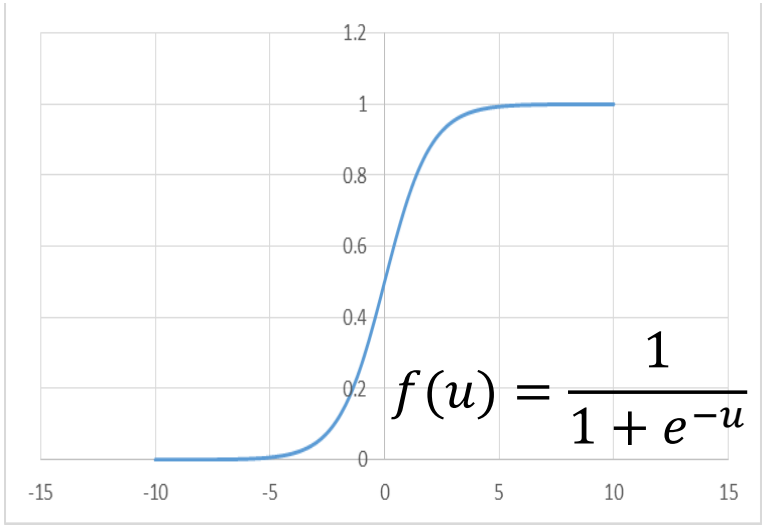
❑ BP算法中激活函数

– 通过误差反向传播算法调整多层感知器的连接权重时，一个瓶颈问题就是激活函数

- M-P 模型中使用阶跃函数作为激活函数，只能输出 0或 1，不连续所以不可导
- 为了使误差能够传播，鲁梅尔哈特等人提出使用可导函数sigmoid作为激活函数



阶跃函数



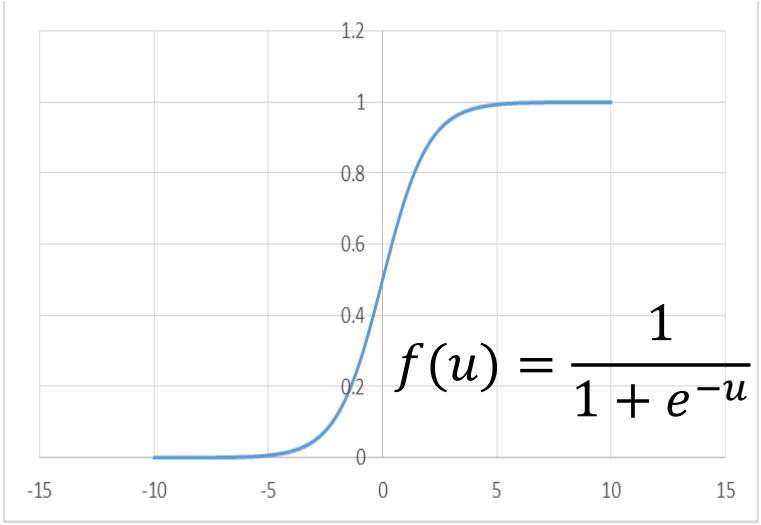
Sigmoid函数

BP算法

❑ BP算法中激活函数

– Sigmoid函数的导数

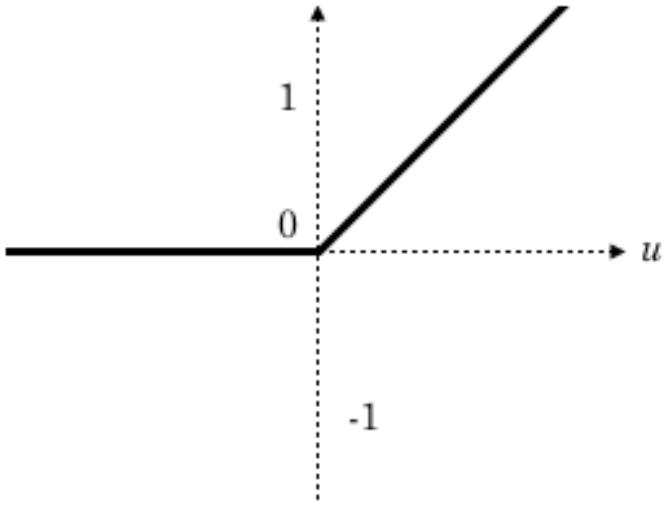
$$\frac{df(u)}{du} = f(u)(1- f(u))$$



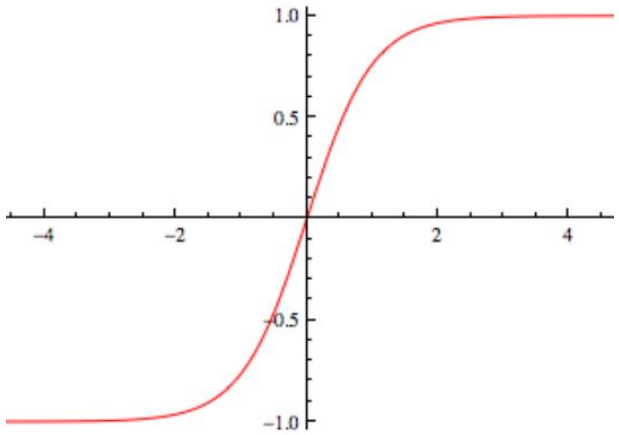
BP算法

❑ 其他常用的激活函数

– ReLU (Rectified Linear Unit,修正线性单元)和tanh等



ReLu



tanh

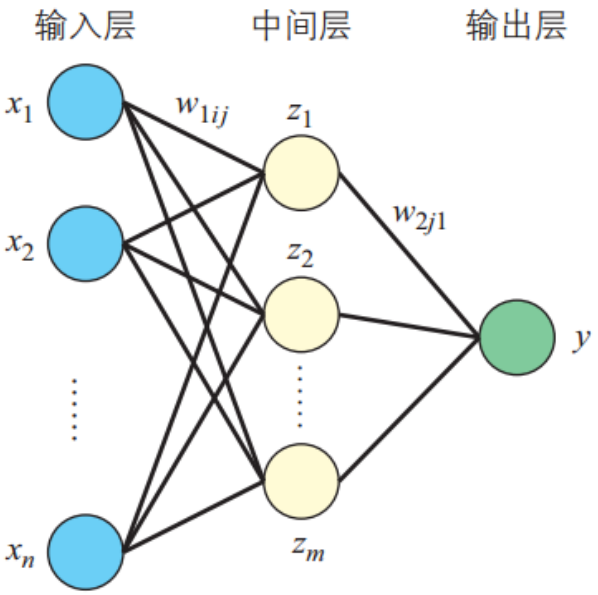
$$\mathrm{ReLu}(x) = \begin{cases} x, & x > 0 \\ 0, & otherwise \end{cases}$$

$$\tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$$

BP算法

❑ BP算法示例

- 以包含一个中间层和一个输出单元y的多层感知器为例：w_{1ij}表示输入层与中间层之间的连接权重，w_{2j1} 表示中间层与输出层之间的连接权重，i表示输入层单元，j表示中间层单元



➢ 首先调整中间层与输出层之间的连接权重，其中$y=f(u)$, f是激活函数，$u_{21} = \sum_{j=1}^{m} w_{2j1} z_j$, 把误差函数$E$对连接权重$w_{2j1}$的求导展开成复合函数求导：

$$\frac{\partial E}{\partial w_{2j1}} = \frac{\partial E}{\partial y}\frac{\partial y}{\partial u_{21}}\frac{\partial u_{21}}{\partial w_{2j1}}$$
$$= -(r-y)y(1-y)z_j$$

这里，z_j表示的是中间层的值

BP算法

- 第二，中间层到输出层的连接权重调整值如下所示：

$$\Delta w_{2j1} = \alpha(r - y)y(1 - y)z_j$$

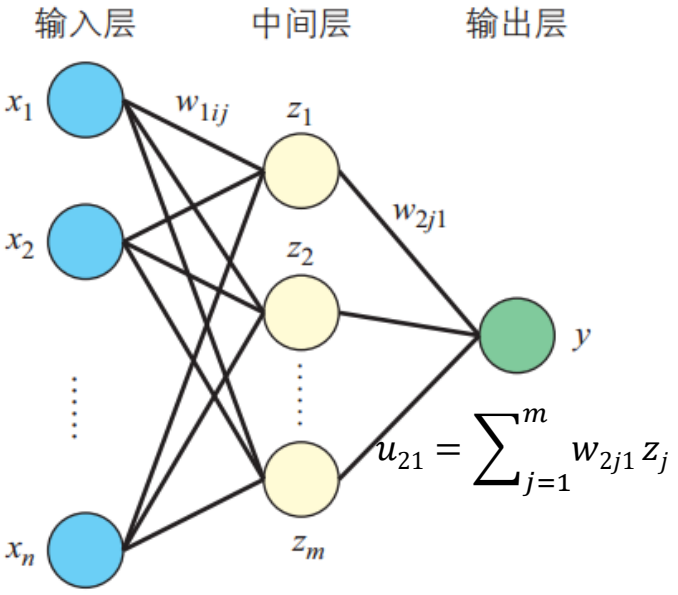
BP算法

❑ 第三，调整输入层与中间层之间的连接权重

– 输入层与中间层之间的连接权重调整值是根据输出层的误差函数确定的，求导公式如下所示：

$$\frac{\partial E}{\partial w_{1ij}} = \frac{\partial E}{\partial y}\frac{\partial y}{\partial u_{21}}\frac{\partial u_{21}}{\partial w_{1ij}}$$

$$= -(r-y)y(1-y)\frac{\partial u_{21}}{\partial w_{1ij}}$$



其中u_{21}由中间层的值z_j和连接权重w_{2j1}计算得到，中间层与输出层单元之间的激活值u_{21}对中间层的值 z_j 求导，结果只和连接权重w_{2j1}相关

$$\frac{\partial u_{21}}{\partial w_{1ij}} = \frac{\partial u_{21}}{\partial z_j}\frac{\partial z_j}{\partial w_{1ij}} \qquad \frac{\partial u_{21}}{\partial z_j} = w_{2j1}$$

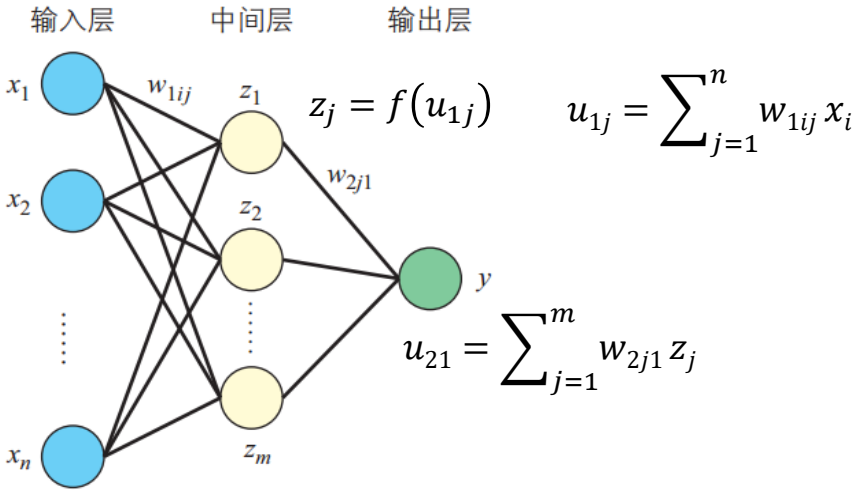
BP算法

❑ 中间层的值z_j对连接权重w_{1ij}求导

$$\frac{\partial z_j}{\partial w_{1ij}} = \frac{\partial z_j}{\partial u_{1j}} \frac{\partial u_{1j}}{\partial w_{1ij}}$$

和y一样，z_j也是sigmoid函数，对z_j求导，得到下式：

$$\frac{\partial z_j}{\partial u_{1j}} = z_j(1 - z_j)$$



BP算法

❑ 中间层的值z_j对连接权重w_{1ij}求导

$$\frac{\partial z_j}{\partial w_{1ij}} = \frac{\partial z_j}{\partial u_{1j}} \frac{\partial u_{1j}}{\partial w_{1ij}}$$

和y一样，z_j也是sigmoid函数，对z_j求导，得到下式：

$$\frac{\partial z_j}{\partial u_{1j}} = z_j(1 - z_j)$$

输入层与中间层单元之间的激活值u_{1j}对中间层与输出层之间的连接权重w_{1ij}求导，结果只和x_i相关：

$$\frac{\partial u_{1j}}{\partial w_{1ij}} = x_i$$

综上，输入层与中间层之间的链接权重调整值如下：

$$\Delta w_{1ij} = \alpha(r - y)y(1 - y)w_{2j1}z_j(1 - z_j)x_i$$

BP算法

❑ 中间层到输出层

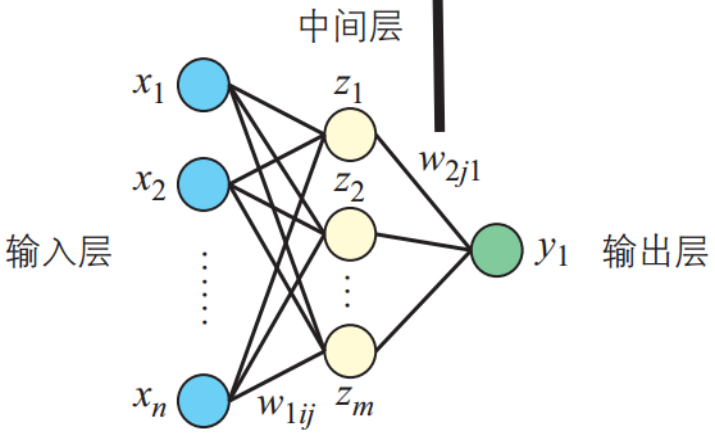
$$\frac{\partial E}{\partial w_{2j1}} = \frac{\partial E}{\partial y}\frac{\partial y}{\partial u_{21}}\frac{\partial u_{21}}{\partial w_{2j1}}$$

$$\frac{\partial E}{\partial w_{2j1}} = -(r-y)\,y(1-y)\,z_j$$

误差函数的导数

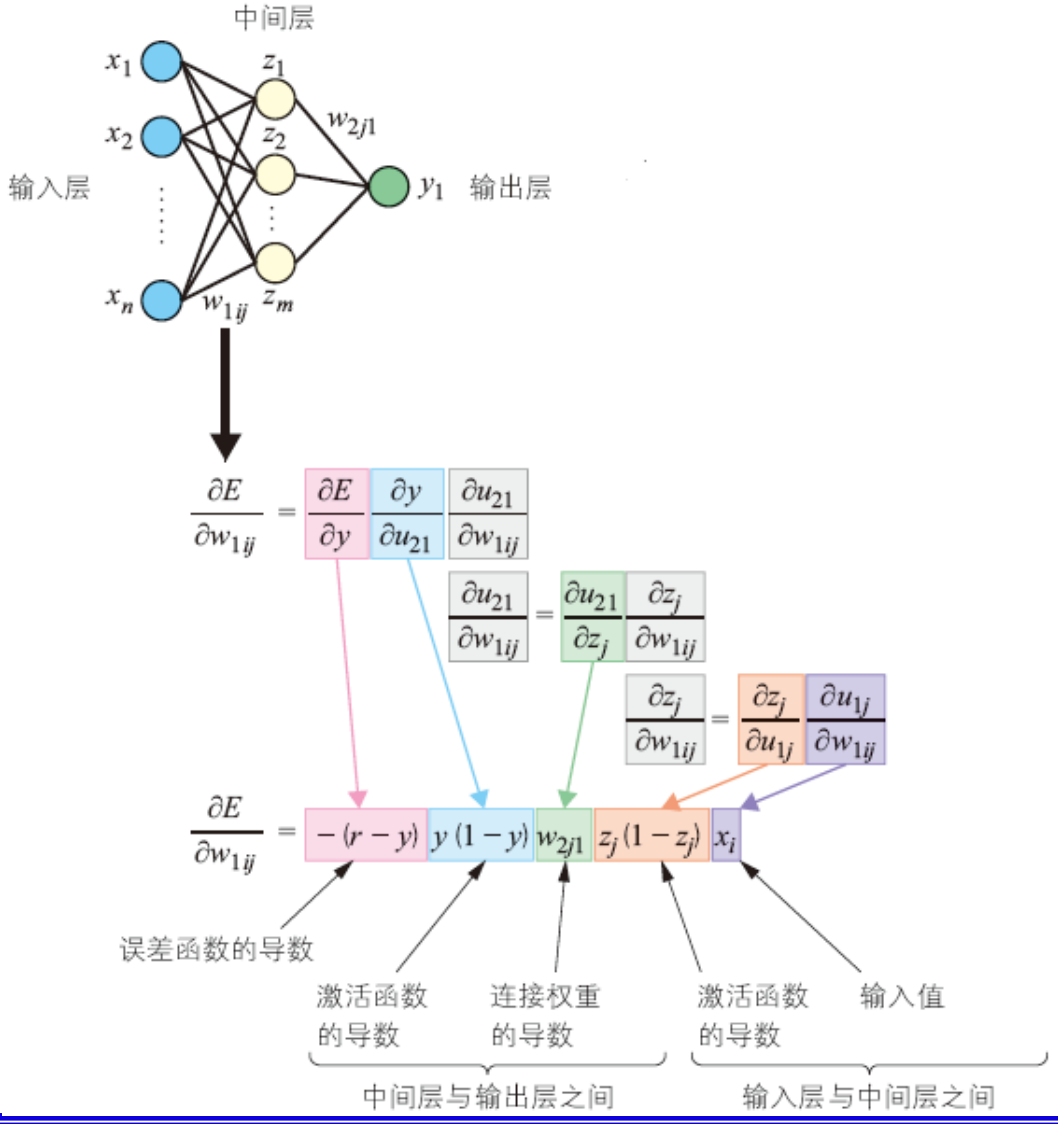
激活函数的导数

连接权重的导数



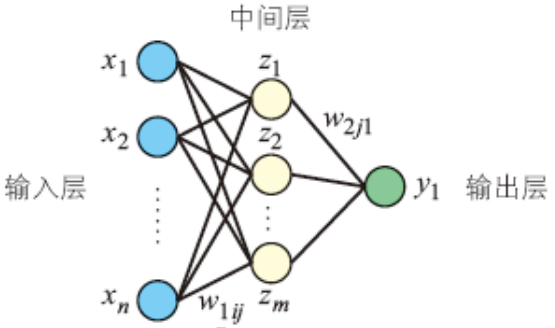
BP算法

❑ 输入层到中间层

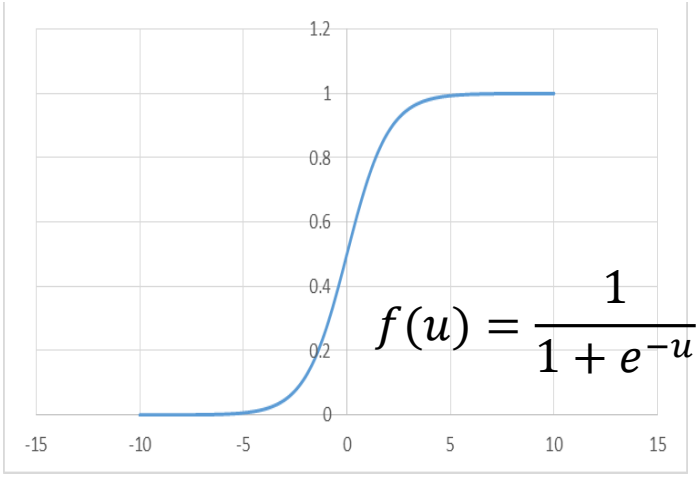


BP算法

❑ 输入层到中间层



$$\frac{df(u)}{du} = f(u)(1- f(u))$$



$$\frac{\partial E}{\partial w_{1ij}} = \frac{\partial E}{\partial y}\frac{\partial y}{\partial u_{21}}\frac{\partial u_{21}}{\partial w_{1ij}}$$

$$\frac{\partial u_{21}}{\partial w_{1ij}} = \frac{\partial u_{21}}{\partial z_j}\frac{\partial z_j}{\partial w_{1ij}}$$

$$\frac{\partial z_j}{\partial w_{1ij}} = \frac{\partial z_j}{\partial u_{1j}}\frac{\partial u_{1j}}{\partial w_{1ij}}$$

$$\frac{\partial E}{\partial w_{1ij}} = -(r-y)\, y(1-y)\, w_{2j1}\, z_j(1-z_j)\, x_i$$

误差函数的导数　激活函数的导数　连接权重的导数　激活函数的导数　输入值

中间层与输出层之间　输入层与中间层之间

6 中英文术语对照

中英文术语对照

- 标量：Scalar
- 向量：Vector
- 张量：Tensor
- 期望：Expectation
- 方差：Variance
- 熵：Entropy
- 过拟合：Overfitting
- 欠拟合：Underfitting
- 监督学习：Supervised learning
- 无监督学习：Unsupervised learning

中英文术语对照

- 数据集：Data set
- 训练集：Training set
- 验证集：Validation set
- 测试集：Testing set
- 泛化：Generalization
- 线性回归：Linear Regression
- 支持向量机：Support Vector Machine
- 决策树：Decision Tree
- 随机森林： Random Forest
- 感知器：Perceptron

中英文术语对照

- 反向传播算法: (error) Back Propagation, BP
- 梯度下降：Gradient Descent
- 修正线性单元：Rectified Linear Unit, ReLU

谢谢！