机器学习

北京大学 龚诚欣 https://wqgcx.github.io/

1 绪论

1.2 基本术语

数据集：记录的集合
示例/样本：关于一个事件或对象的描述，也称为特征向量
属性/特征：反映对象在某方面的表现或性质的事项；张成的空间称为属性空间
属性值：属性上的取值
维数：样本 $\boldsymbol{x}_i$ 属性描述的个数
学习/训练：从数据中学得模型的过程，通过执行某个学习算法完成
假设：学得模型对应的关于数据的某种潜在的规律
样例：拥有标记（关于示例结果的信息）的示例
标记空间/输出空间：所有标记的集合
分类：预测离散值的学习任务
回归：预测连续值的学习任务
测试：学得模型后，使用其进行预测的过程
测试样本：被预测的样本
聚类：将训练集中的西瓜分成若干组，每组称为一个簇
监督学习：拥有标记信息的训练数据的学习任务，如分类和回归
无监督学习：不拥有标记信息的训练数据的学习任务，如聚类
泛化能力：学得模型适用于新样本的能力

1.3 假设空间

我们可以把学习过程看作一个在所有假设（的训练结果）组成的空间中进行搜索的过程，搜索目标是找到与训练集匹配的假设。现实问题往往具有很大的假设空间，但学习过程是基于有限样本训练集进行的，因此可能有多个假设与训练集一致，称之为版本空间（即结果符合给定的样本集）。

1.4 归纳偏好

归纳偏好：机器学习算法在学习过程中对某种类型假设的偏好
Occam's razor（奥卡姆剃刀）：若有多个假设与观察一直，则选最简单的那个
No Free Lunch Theorem（没有免费的午餐定理）：
1° 对所有可能的的目标函数求平均，得到的所有学习算法的“非训练集误差”的期望值相同；
2° 对任意固定的训练集，对所有的目标函数求平均，得到的所有学习算法的“非训练集误差”的期望值也相同；
3° 对所有的先验知识求平均，得到的所有学习算法的“非训练集误差”的期望值也相同；
4° 对任意固定的训练集，对所有的先验知识求平均，得到的所有学习算法的“非训练集误差”的期望值也相同。
意义：脱离具体问题，空泛地谈论“什么学习算法更好”毫无意义。要谈论算法的相对优劣，必须要针对具体的学习问题。学习算法自身的归纳偏好与问题是否相配，往往会起到决定性的作用。

2 模型评估与选择

2.1 经验误差与过拟合

错误率：分类错误的样本数占样本总数的比例
误差：学习器的实际预测输出与样本的真实输出之间的差异
训练误差/经验误差：学习器在训练集上的误差
泛化误差：学习器在新样本上的误差
过拟合：将训练样本自身的一些特点当成了潜在样本都会具有的一般性质，导致泛化性能的下降；过拟合是无法避免的
欠拟合：训练样本的一般性质尚未学好

2.2 评估方法

考虑只有一个包含 m 个样例的数据集 $D=\{(\mathbf{x}_1,y_1),(\mathbf{x}_2,y_2),\cdots(\mathbf{x}_m,y_m)\}$，对 D 进行适当处理，从中产生出训练集 S 和测试集 T

• 留出法

直接将数据集 D 划分为两个互斥的集合，应尽可能保持数据分布的一致性
一般要采用若干次随机划分
问题：若 S 大，则 T 评估结果可能不够稳定准确；若 T 大，则 S 与 D 差别更大，训练出的模型可能由较大差别，降低了评估结果的保真性。

• p 次 k 折交叉验证法

将数据集 D 划分为 k 个大小相似的互斥子集，每个子集尽可能保持数据分布的一致性。每次用 k-1 个子集的并集作为训练集，余下的那个子集作为测试集。随机使用不同的划分重复 p 次。
若 k=|D|，则得到留一法，该方法不受随机样本划分方式的影响。绝大多数情况下，留一法中被实际评估的模型与期望评估的用 D 训练出的模型很相似。
问题：留一法的开销难以忍受，而且估计结果也未必永远准确（NFL 定理）

• 自助法

给定包含 m 个样本的数据集 D，对它进行采样产生数据集 D'：每次随机从 D 中挑选一个样本，拷贝放入 D'，再将该样本放回初始数据集 D 中，使得该样本在下次采样时仍有可能被采到；这个过程重复执行 m 次后，得到了包含 m 个样本的数据集 D'。计算得到初试数据集 D 中约有 36.8%的样本未出现在采样数据集 D'中，可将 D'用作训练集，D\D'用作测试集。该方法对数据集较小、难以有效划分训练/测试集时很有用，对集成学习等方法也有很大的好处。
问题：改变了初始数据集得到分布，会引入估计偏差。在数据量足够时，留出法和交叉验证法更常用。

• 调参与最终模型

大多数学习算法都有参数需要设定，参数配置不同，学得模型的性能往往有显著差别；因此，除了对使用学习算法进行选择，还需要对算法参数进行设定。
在模型选择完成后，学习算法和参数配置已经选定，这时应该用整个数据集 D 重新训练模型。
通常把学得模型在实际使用中遇到的数据称为测试数据，模型评估与选择中用于评估测试的数据集常称为验证集。

2.3 性能度量

给定样例集 $D=\{(\mathbf{x}_1,y_1),(\mathbf{x}_2,y_2),\cdots(\mathbf{x}_m,y_m)\}$，考虑均方误差：

$$E(f;D)=\frac{1}{m}\sum_{i=1}^{m}(f(\boldsymbol{x}_i)-y_i)^2$$

对于一般的数据分布 D 和概率密度函数 p(·)，考虑均方误差：

$$E(f;D)=\int_{x\sim D}(f(\boldsymbol{x})-y)^2 p(\boldsymbol{x})d\boldsymbol{x}$$

• 错误率与精度

错误率： $E(f;D):=\int_{x\sim D}\chi(f(\boldsymbol{x})\neq y)p(\boldsymbol{x})d\boldsymbol{x}$

精度： $acc(f;D):=\int_{x\sim D}\chi(f(\boldsymbol{x})=y)p(\boldsymbol{x})d\boldsymbol{x}$

• 查准率、查全率与 F1

考虑真正例(TP)、假正例(FP)、真反例(TN)、假反例(FN)，则 TP+FP+TN+FN=样例总数。

混淆矩阵：

真实情况	预测结果	
	正例	反例
正例	TP	FN
反例	FP	TN

查准率定义为 $P:=\frac{TP}{TP+FP}$，查全率定义为 $R:=\frac{TP}{TP+FN}$。

查准率和查全率通常是一对矛盾的度量。

我们可以根据学习器的预测结果对样例按可能性高低到进行排序，逐个把样本作为正例进行预测，可以得到 P-R 曲线。如果一个学习器的 P-R 曲线完全包住另一个学习器的 P-R 曲线，则可断言前者优于后者。

人们设计了一些综合考虑查准率、查全率的性能度量：

平衡点（BEP）：查准率=查全率时的取值。

F1 度量： $F1:=\frac{2\times P\times R}{P+R}=\frac{2\times TP}{样例总数+TP-TN}$

在一些应用中，对查准率和查全率的重视程度有所不同。我们考虑 F1 度量的一般形式： $F_\beta=\frac{(1+\beta^2)\times P\times R}{(\beta^2\times P)+R}(\beta>0)$。

β＞1 时查全率有更大影响，β＜1 时查准率有更大影响。

我们希望在 n 个二分类混淆矩阵上综合考察查准率和查全率。

先分别计算查准率和查重率，得到$(P_1,R_1),(P_2,R_2),\cdots,(P_n,R_n)$，再计算平均：

宏查准率： $macro-P:=\frac{1}{n}\sum_{i=1}^{n}P_i$；宏查全率： $macro-R:=\frac{1}{n}\sum_{i=1}^{n}R_i$

$$macro-F1:=\frac{2\times(macro-P)\times(macro-R)}{(macro-P)+(macro-R)}$$

也可以先将混淆矩阵对应元素进行平均，得到 $\overline{TP},\overline{FP},\overline{TN},\overline{FN}$，再基于这些数据计算：

微查准率： $micro-P:=\frac{\overline{TP}}{\overline{TP}+\overline{FP}}$；微查全率： $micro-R:=\frac{\overline{TP}}{\overline{TP}+\overline{FN}}$

$$micro-F1:=\frac{2\times(micro-P)\times(micro-R)}{(micro-P)+(micro-R)}$$

• ROC 与 AUC

很多学习器是为测试样本产生一个实值或概率预测，然后将这个预测值与一个分类阈值进行比较，大于阈值则分为正类，否则为反类。

ROC 曲线（受试者工作特征 Receiver Operating Characteristic 曲线）：纵坐标为真正例率 TPR，横坐标为假正例率 FPR

$TPR:=\frac{TP}{TP+FN}$； $FPR:=\frac{FP}{TN+FP}$

与 P-R 图类似，若一个学习器的 ROC 曲线完全包住另一个学习器的 ROC 曲线，则可断言前者性能优于后者。若曲线交叉，则可以考虑 ROC 曲线下的面积，即 AUC（Area Under ROC Curve）。

离散情形绘制 ROC 曲线：给定 m^+个正例和 m^-个反例，先将分类阈值设为最大，预测所有样例均为反例，此时 TPR 和 FPR 均为 0，标记点为(0,0)；然后再从高到低依次调整分类阈值为当前预测值。设前一个标记点坐标为(x,y)，若当前样本为真正例，则坐标为 $(x,y+\frac{1}{m^+})$；若为假正例，则坐标为 $(x+\frac{1}{m^-},y)$；若是一组样本，则只记录这组全部坐标的起点和终点（中间点忽略）。然后依次用线段连接相邻点即可。

设 ROC 曲线是由坐标为$\{(x_1,y_1),(x_2,y_2),\cdots,(x_m,y_m)\}$的点按序连接而形成的曲线，则 $AUC=\frac{1}{2}\sum_{i=1}^{m-1}(x_{i+1}-x_i)(y_i+y_{i+1})$。

AUC 考虑的是样本预测的排序质量，因此它与排序误差紧密联系。给定 m^+个正例和 m^-个反例，令 D^+和 D^-分别表示正、反例集合，则排序“损失”定义为：

$$l_{rank}=\frac{1}{m^+m^-}\sum_{x^+\in D^+}\sum_{x^-\in D^-}\left(\chi(f(\boldsymbol{x}^+)<f(\boldsymbol{x}^-))+\frac{1}{2}\chi(f(\boldsymbol{x}^+)=f(\boldsymbol{x}^-))\right);$$

即考虑每一个正例和每一个反例，若正例预测值小于反例，则记 1 个“罚分”；若相等，则记 0.5 个“罚分”。

容易看出，l_{rank}对应的是 ROC 曲线之上的面积，因此有 $AUC=1-l_{rank}$。

• 代价敏感错误率与代价曲线

为权衡不同类型错误所造成的不同损失，可为错误赋予“非均等代价”。

二分类代价矩阵（cost matrix）：

真实类别	预测类别	
	第 0 类	第 1 类
第 0 类	0	$cost_{01}$
第 1 类	$cost_{10}$	0

代价敏感错误率：（设第 0 类为正类，第 1 类为反类，D^+代表样例集 D 的正例子集，D^-代表反例子集）

$$E(f;D;cost)=\frac{1}{m}\left(\sum_{x_i\in D^+}\chi(f(x_i)\neq y_i)\times cost_{01}+\sum_{x_i\in D^-}\chi(f(x_i)\neq y_i)\times cost_{10}\right)$$

代价曲线：横轴是取值为[0,1]的正例概率代价

$P(+)cost=\dfrac{p\times cost_{01}}{p\times cost_{01}+(1-p)\times cost_{10}}$，p 是样例为正例的概率

纵轴是取值为[0,1]的归一化代价

$cost_{norm}=\dfrac{FNR\times p\times cost_{01}+FPR\times(1-p)\times cost_{10}}{p\times cost_{01}+(1-p)\times cost_{10}}$，FPR 是假正例率，FNR 是假反例率，FNR=1-TPR。

代价曲线的绘制：ROC 曲线上每一点对应了代价平面上的一条线段，设 ROC 曲线上点的坐标为(FPR,TPR)，相应计算出 FNR，然后在代价平面上绘制一条从(0,FPR)到(1,FNR)的线段，线段下的面积即表示了该条件下的期望总体代价；如此将 ROC 曲线上的每个点转化为代价平面上的一条线段，然后取所有线段的下界，围成的面积即为在所有条件下学习器的期望总体代价。

2.5 偏差与方差

对测试样本 $\boldsymbol{x}$，设 y_D 为 $\boldsymbol{x}$ 在数据集中的标记，y 为 $\boldsymbol{x}$ 的真实标记，$f(\boldsymbol{x};D)$为训练集 D 上学得模型 f 在 $\boldsymbol{x}$ 上的预测输出。以回归任务为例：

学习算法的期望预测为：$\bar{f}(\boldsymbol{x})=\mathrm{E}_D[f(\boldsymbol{x};D)]$；

使用样本数相同的不同训练集产生的方差为：$var(\boldsymbol{x})=\mathrm{E}_D\left[\left(f(\boldsymbol{x};D)-\bar{f}(\boldsymbol{x})\right)^2\right]$；

噪声为：$\varepsilon^2=\mathrm{E}_D\left[(y_D-y)^2\right]$，便于讨论设$\mathrm{E}_D\left[y_D-y\right]=0$；

期望输出与真实标记的差别称为偏差，即：$bias^2(\boldsymbol{x})=\left(\bar{f}(\boldsymbol{x})-y\right)^2$；

因此期望泛化误差（学习器输出结果与实际标记偏离程度的期望）：

$E(f;D)=\mathrm{E}_D\left[\left(f(\boldsymbol{x};D)-y_D\right)^2\right]=bias^2(\boldsymbol{x})+var(\boldsymbol{x})+\varepsilon^2$；

也就是说，泛化误差可分解为偏差、方差与噪声之和。

偏差度量了学习算法的期望预测与真实结果的偏离程度，即刻画了学习算法本身的拟合能力；方差度量了同样大小的训练集的变动所导致的学习性能的变化，即刻画了数据扰动所造成的影响；噪声则表达了在当前任务上任何学习算法所能达到的期望泛化误差的下界，即刻画了学习问题本身的难度。

一般来说，偏差与方差是有冲突的，这被称为偏差-方差窘境。训练不足时，学习器拟合能力不够强，训练数据的扰动不足以使学习器产生显著变化，此时偏差主导了泛化错误率；在训练程度充足后，学习器的你和能力已非常强，训练数据发生的轻微扰动都会导致学习器发生显著变化；若训练数据自身的、非全局的特性被学习器学到了，则将发生过拟合。

3 线性模型

3.1 基本形式

给定 d 个属性描述的示例 $\boldsymbol{x}=(x_1,x_2,\cdots,x_d)$，其中 x_i 是 $\boldsymbol{x}$ 在第 i 个属性上的取值，线性模型试图学的一个通过属性的线性组合来进行预测的函数，即 $f(\boldsymbol{x})=\boldsymbol{\omega}^T\boldsymbol{x}+b$。$\boldsymbol{\omega}$和 b 确定后，模型就得以确定。

3.2 线性回归

给定数据集 D = $\{(\boldsymbol{x}_1,y_1),\cdots,(\boldsymbol{x}_m,y_m)\}$，线性回归试图学得 $f(\boldsymbol{x}_i)=\omega\boldsymbol{x}_i+b$，使得 $f(\boldsymbol{x}_i)\approx y_i$。先考虑一维情况。

均方误差最小化：$(\omega^*,b^*)=\arg\min_{(\omega,b)}\sum_{i=1}^{m}(f(x_i)-y_i)^2$。基于均方误差最小化来进行模型求解的方法称为“最小二乘法”。

最优解：$\omega=\dfrac{\sum_{i=1}^{m}y_i(x_i-\bar{x})}{\sum_{i=1}^{m}x_i^2-\frac{1}{m}\left(\sum_{i=1}^{m}x_i\right)^2}$，$b=\dfrac{1}{m}\sum_{i=1}^{m}(y_i-\omega x_i)$。

再考虑多元线性回归。方便讨论，记 $\hat{\boldsymbol{\omega}}=(\boldsymbol{\omega};b)$；把数据集 D 表示成一个 m×(d+1) 大小的矩阵 **X**，其中每行对应于一个示例，前 d 个元素对应于示例的 d 个属性值，最后一个元素恒置为 1；再把标记也写成向量 $\boldsymbol{y}=(y_1,\cdots,y_m)$，那么有：

$\hat{\boldsymbol{\omega}}^*=\arg\min_{\hat{\omega}}(\boldsymbol{y}-\boldsymbol{X}\hat{\boldsymbol{\omega}})^T(\boldsymbol{y}-\boldsymbol{X}\hat{\boldsymbol{\omega}})$。

记 $E_{\hat{\omega}}=(\boldsymbol{y}-\boldsymbol{X}\hat{\boldsymbol{\omega}})^T(\boldsymbol{y}-\boldsymbol{X}\hat{\boldsymbol{\omega}})$，则 $\dfrac{\partial E_{\hat{\omega}}}{\partial\hat{\boldsymbol{\omega}}}=2\boldsymbol{X}^T(\boldsymbol{X}\hat{\boldsymbol{\omega}}-\boldsymbol{y})$。

当 $\boldsymbol{X}^T\boldsymbol{X}$ 为满秩矩阵时，得 $\hat{\boldsymbol{\omega}}^*=(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y}$，此时 $f(\hat{\boldsymbol{x}}_i)=\hat{\boldsymbol{x}}_i^T(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y}$。

现实生活中，$\boldsymbol{X}^T\boldsymbol{X}$ 往往不是满秩矩阵。此时可解出多个 $\hat{\boldsymbol{\omega}}$，它们都能使均方误差最小化。选择哪一个解作为输出，将由学习算法的归纳偏好决定。

对数线性回归：$\ln y=\boldsymbol{\omega}^T\boldsymbol{x}+b$。

广义线性模型：$y=g^{-1}(\boldsymbol{\omega}^T\boldsymbol{x}+b)$，其中 g(·)单调可微，称为联系函数。

3.3 对数几率回归

考虑二分类任务，利用 Sigmoid 函数，即是 $y=\dfrac{1}{1+e^{-(\boldsymbol{\omega}^T\boldsymbol{x}+b)}}$（真实标记为{0,1}）.

将 y 视为样本 $\boldsymbol{x}$ 作为正例的可能性，则 $1-y$ 是其反例可能性，两者的比值称为“几率”，对几率取对数则得到对数几率：$\ln\dfrac{y}{1-y}=\boldsymbol{\omega}^T\boldsymbol{x}+b$。我们实际上是在用线性回归模型的预测结果去逼近真实标记的对数几率，称为“对数几率回归”。

$\ln\dfrac{p(y=1\mid\boldsymbol{x})}{p(y=0\mid\boldsymbol{x})}=\boldsymbol{\omega}^T\boldsymbol{x}+b$，那么 $p(y=1\mid\boldsymbol{x})=\dfrac{e^{\boldsymbol{\omega}^T\boldsymbol{x}+b}}{1+e^{\boldsymbol{\omega}^T\boldsymbol{x}+b}}$，$p(y=0\mid\boldsymbol{x})=\dfrac{1}{1+e^{\boldsymbol{\omega}^T\boldsymbol{x}+b}}$

采用“极大似然法”。对率回归模型最大化“对数似然”$l(\boldsymbol{\omega},b)=\sum_{i=1}^{m}\ln p(y_i\mid\boldsymbol{x}_i;\boldsymbol{\omega},b)$，

即是每个样本属于其真实标记的概率越大越好。令 $\boldsymbol{\beta}=(\boldsymbol{\omega};b)$，$\hat{\boldsymbol{x}}=(\boldsymbol{x};1)$，从而有 $\boldsymbol{\omega}^T\boldsymbol{x}+b=\boldsymbol{\beta}^T\hat{\boldsymbol{x}}$；再设 $p_1(\hat{\boldsymbol{x}};\boldsymbol{\beta})=p(y=1\mid\hat{\boldsymbol{x}};\boldsymbol{\beta}), p_0(\hat{\boldsymbol{x}};\boldsymbol{\beta})=p(y=0\mid\hat{\boldsymbol{x}};\boldsymbol{\beta})$，则似然项可重写为 $p(y_i\mid\boldsymbol{x}_i;\boldsymbol{\omega},b)=y_ip_1(\hat{\boldsymbol{x}}_i;\boldsymbol{\beta})+(1-y_i)p_0(\hat{\boldsymbol{x}}_i;\boldsymbol{\beta})$，代入知最大化 $l(\boldsymbol{\omega},b)$ 等价于最小化 $l(\boldsymbol{\beta})=\sum_{i=1}^{m}\left(-y_i\boldsymbol{\beta}^T\boldsymbol{x}_i+\ln\left(1+e^{\boldsymbol{\beta}^T\boldsymbol{x}_i}\right)\right)$。这是关于$\boldsymbol{\beta}$的高阶可导连续凸函数，经典的数值优化算法都可求得其最优解。以牛顿法为例，其第 $t+1$ 轮迭代解的更新公式为 $\boldsymbol{\beta}^{t+1}=\boldsymbol{\beta}^t-\left(\frac{\partial^2 l(\boldsymbol{\beta})}{\partial\boldsymbol{\beta}\partial\boldsymbol{\beta}^T}\right)^{-1}\frac{\partial l(\boldsymbol{\beta})}{\partial\boldsymbol{\beta}}$，其中

$$\frac{\partial l(\boldsymbol{\beta})}{\partial\boldsymbol{\beta}}=-\sum_{i=1}^{m}\hat{\boldsymbol{x}}_i(y_i-p_1(\boldsymbol{x}_i;\boldsymbol{\beta}))，\quad \frac{\partial^2 l(\boldsymbol{\beta})}{\partial\boldsymbol{\beta}\partial\boldsymbol{\beta}^T}=\sum_{i=1}^{m}\hat{\boldsymbol{x}}_i\hat{\boldsymbol{x}}_i^T p_1(\hat{\boldsymbol{x}}_i;\boldsymbol{\beta})(1-p_1(\hat{\boldsymbol{x}}_i;\boldsymbol{\beta}))。$$

4 决策树

4.1 基本流程

一棵决策树包含一个根节点、若干个内部结点和若干个叶结点。叶结点对应于决策结果，其他每个结点则对应于一个属性测试；根结点包含样本全集，从根结点到每个叶结点的路径对应了一个测试判定序列。

输入： 训练集 $D=\{(\boldsymbol{x}_1,y_1),(\boldsymbol{x}_2,y_2),\ldots,(\boldsymbol{x}_m,y_m)\}$;
　　属性集 $A=\{a_1,a_2,\ldots,a_d\}$.
过程： 函数 TreeGenerate(D, A)
1: 生成结点 node;
2: **if** D 中样本全属于同一类别 C **then**
3:　将 node 标记为 C 类叶结点; **return**
4: **end if**
5: **if** $A=\varnothing$ **OR** D 中样本在 A 上取值相同 **then**
6:　将 node 标记为叶结点, 其类别标记为 D 中样本数最多的类; **return**
7: **end if**
8: 从 A 中选择最优划分属性 a_*;
9: **for** a_* 的每一个值 a_*^v **do**
10:　为 node 生成一个分支; 令 D_v 表示 D 中在 a_* 上取值为 a_*^v 的样本子集;
11:　**if** D_v 为空 **then**
12:　　将分支结点标记为叶结点, 其类别标记为 D 中样本最多的类; **return**
13:　**else**
14:　　以 TreeGenerate(D_v, $A\setminus\{a_*\}$)为分支结点
15:　**end if**
16: **end for**
输出： 以 node 为根结点的一棵决策树

接下来我们重点讨论如何选择最优划分属性。

4.2 划分选择

信息熵：当前样本集合 D 中第 k 类（标记种类数）样本占比为 p_k(k=1,2,⋯,|y|)，

定义 D 的信息熵为：$Ent(D)=-\sum_{k=1}^{|y|}p_k\log_2 p_k$。Ent(D)值越小，D 的纯度越高。

假定离散属性 a 有 V 个可能的取值$\{a^1,a^2,\cdots,a^V\}$，若使用 a 来对样本集 D 进行划分，则会产生 V 个分支结点，其中第 v 个分支结点包含了 D 中所有在属性 a 上取值为 a^v 的样本，记为 D^v。定义用属性 a 对样本集 D 进行划分所获得的“信息增益”：$Gain(D,a)=Ent(D)-\sum_{v=1}^{V}\frac{|D^v|}{|D|}Ent(D^v)$。信息增益越大，用属性 a 来进行划分所获得的“纯度提升”越大。（ID3 决策树算法）

增益率：$Gain\_ratio(D,a)=\frac{Gain(D,a)}{IV(a)}$，其中 $IV(a)=-\sum_{v=1}^{V}\frac{|D^v|}{|D|}\log_2\frac{|D^v|}{|D|}$称为属性 a 的固有值（“固有”：与标记无关）。属性 a 的可能取值数目越多，IV(a)的值通常会越大。一般来说，先从候选划分属性中找出信息增益高于平均水平的属性，再从中选择增益率最高的。（C4.5 决策树算法）

基尼指数：$Gini(D)=1-\sum_{k=1}^{|y|}p_k^2$，反映了数据集 D 中随机抽取两个样本，其类别标记不一致的概率；因此 Gini(D)值越小，数据集 D 的纯度越高。

属性 a 的基尼指数定义为：$Gini\_index(D,a)=\sum_{v=1}^{V}\frac{|D^v|}{|D|}Gini(D^v)$。于是，我们在候选属性集合 A 中，选择那个使得划分后基尼指数最小的属性作为最优划分属性。我们也可以按照特征 A 是否取值为 a_i 对数据集 D 进行二分；$D=D_i^A\cup[D-D_i^A]$，$A=a_i$ 的基尼指数定义为 $Gini_d(D,A=a_i)=\frac{|D_i^A|}{|D|}Gini(D_i^A)+\frac{|D-D_i^A|}{|D|}Gini(D-D_i^A)$

$Gini_d(D,A=a_i)$越小，特征 $A=a_i$ 分类能力越强。因此 CART 算法可以是二路划分，即是选择属性 A 及最小 Gini 指数对应值 a_i。（CART 决策树算法）

相关研究表明，采用不同的最优特征选择准则，对决策树的规模有影响，但对其泛化能力影响不大。

4.3 剪枝处理

剪枝是决策树学习算法对付“过拟合”的主要手段。

预剪枝：在决策树生成过程中，对每个结点在划分前先进行估计，若当前结点的划分不能带来决策树泛化性能的提升，则停止划分并将当前结点标记为叶结点。

后剪枝：先从训练集生成一颗完整的决策树，然后自底向下地对非叶结点进行考察，若将该结点对应的子树替换为叶结点能带来决策树泛化性能提升，则将该子树替换为叶结点。

判断决策树泛化性能的提升可以采用 2.2 节介绍的性能评估方法。

一方面，预剪枝降低了过拟合的风险，显著减少了决策树的训练时间开销和测试时间开销；另一方面，有些分支的当前划分虽不能提升泛化性能，但在其基础上进行的后续分化却有可能导致性能显著提高；预剪枝基于“贪心”本质禁止一些分支展开，给预剪枝决策树带来了欠拟合的风险。

后剪枝决策树通常比预剪枝决策树保留了更多的分支，一般情况下欠拟合风险小，泛化能力往往优于预剪枝决策树；但是后剪枝过程实在生成完全决策树之后进行的，并且要自底向下注意考察，训练时间开销比其他决策树大很多。

4.4 连续与缺失值

• 连续值处理

给定样本集 D 和连续属性 a，假定 a 在 D 上出现了 n 个不同的取值，将这些值从小到大排序，记为$\{a_1,a_2,\cdots,a_n\}$。基于划分点 t 可将 D 分为子集 D_t^-（取值不大于 t 的样本）和 D_t^+（取值大于 t 的样本）。对连续属性 a，我们往往考察包含 n-1 个元素的候选划分点集合 $T_a=\{\frac{a_i+a_{i+1}}{2},1\le i\le n-1\}$，然后就可以像离散属性值一样来考察这些划分点，选择最优的划分点来进行样本集合的划分。例如，我们有

$$Gain(D,a)=\max_{t\in T_a} Gain(D,a,t)=Ent(D)-\min_{t\in T_a}\sum_{\lambda\in\{-,+\}}\frac{|D_t^\lambda|}{|D|}Ent(D_t^\lambda)。$$

• 缺失值处理

不完整样本，即样本的某些属性值缺失。我们需解决 2 个问题：1°如何在属性值缺失的情况下进行属性选择？2°给定划分属性，若样本在该属性上的值缺失，如何对样本进行划分？

给定训练集 D 和属性 a，令 D'表示在属性 a 上没有缺失值的样本子集。对于问题 1，我们可以根据 D'来判断属性 a 的优劣。令 D'^v 表示 D'中在属性 a 上取值为 a^v 的样本子集，D'_k 表示 D'中第 k 类（标记种类数）的样本子集。假定我们为每个样本 $\boldsymbol{x}$ 赋予一个权重 $\omega_{\boldsymbol{x}}$，并定义：

$$\rho=\frac{\sum_{\boldsymbol{x}\in D'}\omega_{\boldsymbol{x}}}{\sum_{\boldsymbol{x}\in D}\omega_{\boldsymbol{x}}};\quad p'_k=\frac{\sum_{\boldsymbol{x}\in D'_k}\omega_{\boldsymbol{x}}}{\sum_{\boldsymbol{x}\in D}\omega_{\boldsymbol{x}}};\quad r'_v=\frac{\sum_{\boldsymbol{x}\in D'^v}\omega_{\boldsymbol{x}}}{\sum_{\boldsymbol{x}\in D}\omega_{\boldsymbol{x}}}。\text{（不是简单的个数比例）}$$

直观地看，对属性 a，ρ表示无缺失值样本所占的比例，p'_k 表示无缺失值样本中第 k 类所占的比例，r'^v 则表示无缺失值样本中在属性 a 上取值 a^v 的样本所占的比例。我们将信息增益的计算式推广为：

$$Gain(D,a)=\rho\times Gain(D',a)=\rho\times\left(Ent(D')-\sum_{v=1}^{V}r'_v\,Ent(D'^v)\right),$$

其中 $Ent(D'^v)=-\sum_{k=1}^{|y|}p'_k\log_2 p'_k$。（C4.5 决策树）

在学习开始时，每个样例的权值均为 1；划分后，出现属性缺失的样本同时进入所有分支，但权重在子结点中调整为$\frac{|D^v|}{|D|}$（绝对数量之比）。显然有该样本所有子结点的权重之和是 1。

4.5 多变量决策树

我们把每个属性视为坐标空间中的一个坐标轴，则 d 个属性描述的样本就对应了 d 维空间中的一个数据点，对样本分类意味着在这个坐标空间中寻找不同类样本之间的分类边界。普通的决策树所形成的分类边界往往由若干个与坐标轴平行的

分段组成。
多变量决策树：能实现复杂划分的决策树，允许使用若干特征的线性组合作为切分当前数据集的最优特征，相当于对每个非叶结点学习一个合适的线性分类器，可以用斜的划分边界来逼近复杂的真实分类边界。

5 神经网络

5.1 神经元模型

神经网络：具有适应性的简单单元组成的广泛并行互连的网络，它的组织能够模拟生物神经系统对真实世界物体所作出的交互反应。其最基本的成分是神经元模型，即上述定义中的“简单单元”。
M-P 神经元模型：神经元接收到来自 n 个其他神经元传递过来的输入信号，这些输入信号通过带权重的连接进行传递，神经元接收到的总输入值将与神经元的阈值进行比较，然后通过激活函数 $y = f(\sum_{i=1}^{n} \omega_i x_i - \theta)$ 处理以产生神经元的输出。常采用 Sigmoid（形似 S 的函数）函数（如 $\text{sigmoid}(x) = \frac{1}{1+e^{-x}}$）作为激活函数。

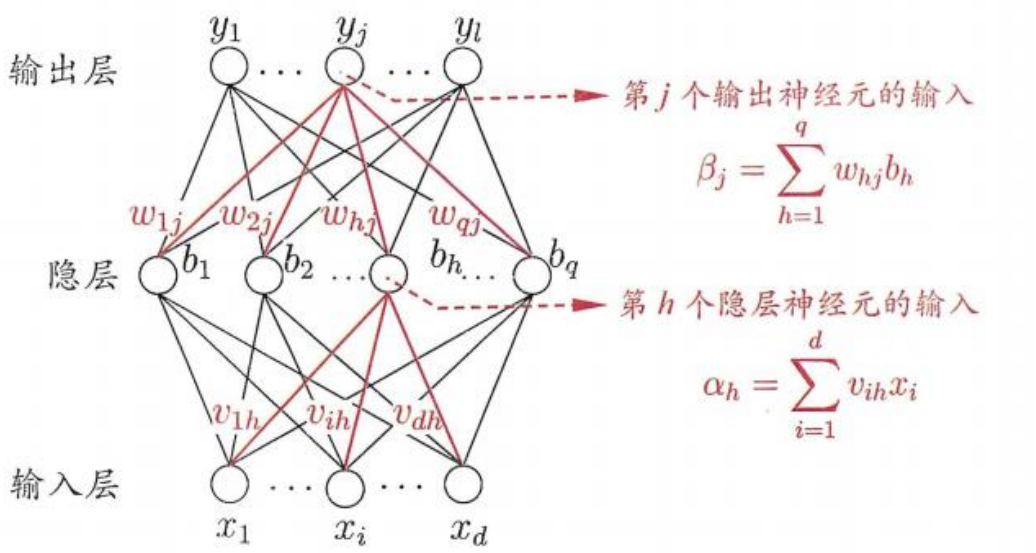
5.2 感知机与多层网络

感知机由两层神经元组成，输入层接收外界输入信号后传递给输出层，输出层是 M-P 神经元，亦称“阈值逻辑单元”。
阈值θ可以看作一个固定输入为−1.0 的哑结点所对应的连接权重ω_{n+1}，这样就可统一为权重的学习。感知机的学习规则如下：对训练样例$(\boldsymbol{x},y)$，若当前感知机的输入为 y'，则感知机的权重这样调整：$\omega_i \leftarrow \omega_i + \Delta\omega_i$，$\Delta\omega_i = \eta(y - y')x_i$。
若两类学习模式是线性可分的，即存在一个线性超平面能将它们纷纷开，则感知机的学习过程一定会收敛而求得适当的权向量$\boldsymbol{\omega}=(\omega_1,\omega_2,\cdots,\omega_{n+1})$；否则感知机学习过程将会发生振荡，$\boldsymbol{\omega}$难以稳定下来，不能求得合适解。
要解决非线性可分问题，需考虑使用多层功能神经元。输吹曾与输入层之间的神经元称为隐含层，隐含层和输出层神经元都是拥有激活函数的功能神经元。
更一般地，常见的神经网络是层级结构，每层神经元与下一层神经元全互连，神经元之间不存在同层连接，也不存在跨层连接。这样的神经网络结构通常称为“多层前馈神经网络”。神经网络“学”到的东西，蕴含在连接权与阈值中。

5.3 误差逆传播(error BackPropagation,BP)算法

给定训练集 $D = \{(\boldsymbol{x}_1,\boldsymbol{y}_1),\cdots,(\boldsymbol{x}_m,\boldsymbol{y}_m)\}$, $\boldsymbol{x}_i \in \mathrm{R}^d, \boldsymbol{y}_i \in \mathrm{R}^l$，即输入示例由 d 个属性描述，输出 l 维实值向量。便于讨论，我们给出一个拥有 d 个输入神经元，l 个输出神经元，q 个隐层神经元的多层前馈（指网络拓扑结构上不存在环或回路）网络结构，其中输出层的第 j 个神经元的阈值用θ_j表示，隐层第 h 个神经元的阈值用γ_h表示，输入层第 i 个神经元与隐层第 h 个神经元之间的连接权为 V_{ih}，隐层第 h 个神经元与输出层第 j 个神经元之间的连接权为 W_{hj}。记隐层第 h 个神经元接收到的输入为$\alpha_h=\Sigma_{i=1,\cdots,d}V_{ih}x_i$，输出层第 j 个神经元接收到的输出为$\beta_j=\Sigma_{h=1,\cdots,q}W_{hj}b_h$，其中 b_h 为隐层第 h 个神经元的输出。

对训练例$(\boldsymbol{x}_k,\boldsymbol{y}_k)$，假定神经网络的输出为 $\boldsymbol{y}'_k = (y'^1_k,\cdots,y'^l_k)$，即 $y'^j_k = f(\beta_j-\theta_j)$，则网络在$(\boldsymbol{x}_k,\boldsymbol{y}_k)$上的均方误差为 $E_k = \frac{1}{2}\sum_{j=1}^{l}(y'^j_k - y^j_k)^2$ 。



图中的网络需要有(d+*l*+1)q+*l* 个参数需确定。BP 是一个迭代学习算法，在迭代的每一轮中采用广义的感知机学习规则对参数进行更新估计，任意参数的更新估计式为 *v*←*v*+Δ*v*。基于梯度下降策略，以目标的负梯度方向对参数进行调整，对误差 E_k，给定学习率η，有 $\Delta W_{hj} = -\eta \frac{\partial E_k}{\partial W_{hj}} = -\eta \frac{\partial E_k}{\partial y'^j_k} \cdot \frac{\partial y'^j_k}{\partial \beta_j} \cdot \frac{\partial \beta_j}{\partial W_{hj}}$，

其中 $\frac{\partial \beta_j}{\partial W_{hj}} = b_h$，$-\frac{\partial E_k}{\partial y'^j_k} \frac{\partial y'^j_k}{\partial \beta_j} = -(y'^j_k - y^j_k) f'(\beta_j - \theta_j) = y'^j_k (1 - y'^j_k)(y^j_k - y'^j_k) := g_j$。

那么得到 $\Delta W_{hj} = \eta g_j b_h$，类似得到 $\Delta\theta_j = -\eta g_j$，$\Delta V_{ih} = \eta e_h x_i$，$\Delta\gamma_h = -\eta e_h$，

其中 $e_h = b_h(1-b_h)\sum_{j=1}^{l} W_{hj} g_j$。

学习率η∈(0,1)控制着算法每一轮迭代中的更新步长，若太大则容易振荡，太小则收敛速度过慢。

输入: 训练集 $D = \{(\boldsymbol{x}_k, \boldsymbol{y}_k)\}_{k=1}^{m}$;
学习率 η.

过程:

在$(0,1)$范围内随机初始化网络中所有连接权和阈值
repeat
  for all $(\boldsymbol{x}_k, \boldsymbol{y}_k) \in D$ **do**
    根据当前参数和式(5.3) 计算当前样本的输出 $\hat{\boldsymbol{y}}_k$;
    根据式(5.10) 计算输出层神经元的梯度项 g_j;
    根据式(5.15) 计算隐层神经元的梯度项 e_h;
    根据式(5.11)-(5.14) 更新连接权 w_{hj}, v_{ih} 与阈值 θ_j, γ_h
  end for
until 达到停止条件

输出: 连接权与阈值确定的多层前馈神经网络

BP 算法的目标是最小化训练集 D 上的累计误差 $E=\frac{1}{m}\sum_{k=1}^{m}E_k$。我们上面介绍的“标准 BP 算法”是仅针对一个训练样例更新连接权和阈值，如果类似地推导出基于累计误差最小化的更新规则，就就得到了累计误差逆传播算法（累计 BP 算法）。一般来说，标准 BP 算法每次更新只针对单个样例，参数更新的非常频繁，而且对不同样例进行更新的效果可能出现“抵消”现象。因此，为了达到同样的累计误差极小点，标准 BP 算法往往需进行更多次数的迭代。累计 BP 算法直接针对累计误差最小化，它在读取整个训练集 D 一遍后才对参数进行更新，其参数更新的频率低很多。但在很多任务中，累计误差下降到一定程度之后，进一步下降会非常缓慢，这时候标准 BP 往往会更快获得较好的界，尤其是在训练集 D 非常大时更明显。

Hornik 证明：只需一个包含足够多神经元的隐层，多层前馈网络就能以任意精度逼近任意复杂度的连续函数。然而，如何设置隐层神经元的个数仍是个未决问题，实际应用中通常靠“试错法”调整。

由于强大的表示能力，BP 神经网络经常遭遇过拟合，训练误差持续降低，但测试误差却可能上升。有两种策略常用来缓解 BP 网络的过拟合：一种是“早停”，即是将数据分成训练集和验证集，训练集用来计算梯度、更新连接权和阈值，验证集用来估计误差，若训练集误差降低但验证集误差升高，则停止训练；另一种是“正则化”，即是在误差目标函数中增加一个用于描述网络复杂度的部分，例如连接权与阈值的平方和（使得训练过程偏好比较小的连接权和阈值，使网络输出更加“光滑”），误差目标函数改为 $\lambda\frac{1}{m}\sum_{k=1}^{m}E_k+(1-\lambda)\sum_i\omega_i^2$，其中 $\lambda\in(0,1)$ 用于对经验误差与网络复杂度这两项进行这种，常通过交叉验证法来估计。

5.4 全局最小与局部极小

跳出局部极小的方法：

1. 多组不同参数值初始化多个神经网络，在可能陷入不同的局部极小中选择有可能获得更接近全局最小的结果；
2. 模拟退火技术，每一步都以一定概率接受比当前解更差的结果；
3. 使用随机梯度下降，在计算梯度时加入随机因素；
4. 遗传算法。

上述用于跳出局部极小的技术大多是启发式（基于直观或经验构造的算法），理论上缺乏保障。

5.5 其他常见神经网络

RBF（径向基函数）网络：单隐层前馈神经网络，使用径向基函数（某种沿径向对称的标量函数，通常定义为样本到数据中心之间欧氏距离的单调函数）作为激活函数，输出层是对隐层神经元输出的线性组合，可表示为 $\varphi(\boldsymbol{x})=\sum_{i=1}^{q}w_i\rho(\boldsymbol{x},c_i)$，其中 q 是隐层神经元个数，$\boldsymbol{c}_i$ 和 w_i 分别是第 i 个隐层神经元所对应的中心和权重，ρ是径向基函数。已经证明具有足够多隐层神经元的 RBF 网络能以任意精度逼近任意连续函数。训练时，先采用随机采样、聚类等方法确定 $\boldsymbol{c}_i$，再利用 BP 算法。

竞争性学习：无监督学习策略，网络的输出神经元相互竞争，每一时刻仅有一个

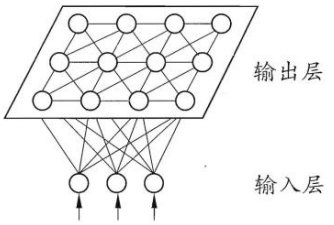
竞争获胜的神经元被激活，其余被抑制，称为“胜者通吃”原则。

ART（自适应谐振理论）网络：一种竞争性学习。由比较层、识别层、识别阈值和重置模块构成。比较层负责接收输入样本，并传递给识别层。识别层每个神经元对应一个模式类，神经元数目可在训练过程中动态增长以增加新的模式类。在接收到比较曾的输入信号后，识别层神经元之间相互竞争产生获胜神经元。竞争最简单方式是计算输入向量与每个识别层神经元所对应模式类的代表向量之间的距离，距离最小者胜。获胜神经元向其他识别层神经元发送信号，抑制其激活。若输入向量与获胜神经元所对应的代表向量之间的相似度大于识别阈值，则当前输入样本将归为该代表向量所属类别，同时更新连接权，使得以后在接收到相似输入样本时会计算出更大的相似度。若相似度不大于识别阈值，则重置模块将在识别层增设一个神经元，其代表向量就设置为当前输入向量。

识别阈值对 ART 网络的性能有重要影响。当识别阈值较高时，输入样本将会被分成比较多、比较精细的模式类，而如果识别阈值比较低，则会产生比较少、比较粗略的模式类。

ART 缓解了竞争性学习中的“可塑性-稳定性窘境”，并可以增量学习或在线学习。（增量学习：学得模型后，再接受新样例时，仅需根据新样例对模型进行更新，不必重新训练整个模型，并且先前学得的有效信息不会被冲掉；在线学习：每获得一个新样本就进行一次模型更新，是增量学习的特例。）

SOM（自组织映射）网络：竞争性学习，将输入的高维数据映射到低维空间，保持输入数据在高维空间的拓扑结构，即将高维空间中相似的样本点映射到网络输出层中的邻近神经元。

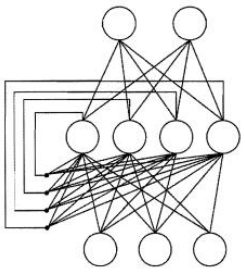


SOM 网络中每个神经元拥有一个权向量，网络接收输入向量后，将会确定输出层获胜神经元，它决定了该输入向量在低维空间中的位置。SOM 训练目标就是为每个输出层神经元找到合适的权向量，以达到保持拓扑结构的目的。

训练过程“在接收到一个训练样本后，每个神经元会计算该样本与自身携带的权向量之间的距离，距离最近的神经元成为竞争获胜者，称为最佳匹配单元。然后，最佳匹配单元及其邻近神经元的权向量将被调整，以使得这些权向量与当前输入样本的距离缩小。这个过程不断迭代，直至收敛。

CC（级联相关）网络：将网络结构当作学习目标，并希望能在训练过程中找到最符合数据特点的网络结构。主要包含两个成分：“级联”和“相关”。级联是指建立层次连接的层次结构。在开始训练时，网络只有输入层和输出层，处于最小拓扑结构；随着训练的进行，新的隐层神经元逐渐加入，从而创建起层级结构。当新的隐层神经元加入时，其输入端连接权值是冻结固定的。相关是指通过最大化新神经元的输出与网络误差之间的相关性来训练相关的参数。

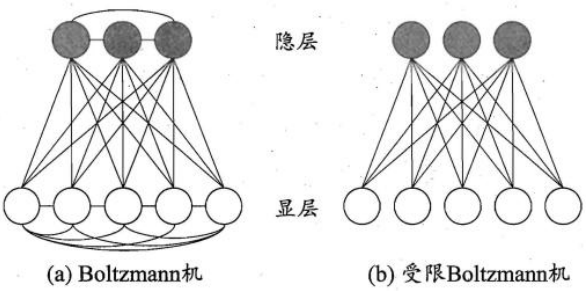
递归神经网络：允许网络中出现环形结构，从而可让一些神经元的输出反馈回来作为输入信号。这使得网络在 *t* 时刻的输出状态还与之前时刻的网络状态有关，从而能处理与时间有关的动态变化。



Elman 网络：递归神经网络之一。隐层神经元的输出被反馈回来，与下一时刻输入层神经元提供的信号一起作为隐层神经元在下一时刻的输入。采用 Sigmoid 激活函数，网络训练采用推广的 BP 算法进行。

Boltzmann 机：为网络状态定义一个“能量”，能量最小化

时网络达到理想状态，网络的训练就是最小化能量函数。神经元分为显层和隐层，显层用于表示数据的输入与输出，隐层则被理解为数据的内在表达。Boltzmann 机中的神经元都是布尔型的，1 表示激活，0 表示抑制。令向量 $\boldsymbol{s}\in\{0,1\}^n$ 表示 n 个神经元的状态，w_{ij} 表示神经元 i,j 之间的连接权，θ_i 表示神经元 i 的阈值，则状态向量 $\boldsymbol{s}$ 所对应的 Boltzmann 机能量定义为 $E(\boldsymbol{s})=-\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}w_{ij}s_is_j-\sum_{i=1}^{n}\theta_is_i$。若网络中的神经元以任意不依赖于输入值的顺序进行更新，则网络最终将达到 Boltzmann 分布，此时状态向量 $\boldsymbol{s}$ 出现的概率将仅有其能量与所有可能状态向量的能量确定：$P(\boldsymbol{s})=\frac{e^{-E(\boldsymbol{s})}}{\Sigma_t e^{-E(t)}}$。Boltzmann 机的训练过程就是将每个训练样本视为一个状态变量，使其出现的概率尽可能大。标准的 Boltzmann 机是一个全连接图，训练网络的复杂度很高，这使其难以用于解决现实任务，故常采用受限 Boltzmann 机。受限 Boltzmann 机常用“对比散度”CD 算法来进行训练。假定网络中有 d 个显层神经元和 q 个隐层神经元，令 $\boldsymbol{v}$ 和 $\boldsymbol{h}$ 分别表示显层与隐层的状态变量，则由于同一层不存在连接，有 $P(\boldsymbol{v}\mid\boldsymbol{h})=\prod_{i=1}^{d}P(v_i\mid\boldsymbol{h})$，$P(\boldsymbol{h}\mid\boldsymbol{v})=\prod_{i=1}^{q}P(h_i\mid\boldsymbol{v})$。CD 算法对每个训练样本 $\boldsymbol{v}$，先计算出隐层神经元状态的概率分布，然后根据这个概率分布采样得到 $\boldsymbol{h}$；再根据 $\boldsymbol{h}$ 产生 $\boldsymbol{v}'$，$\boldsymbol{v}'$产生 $\boldsymbol{h}'$；连接权的更新公式为$\Delta w=\eta(\boldsymbol{v}\boldsymbol{h}^T-\boldsymbol{v}'\boldsymbol{h}'^T)$。

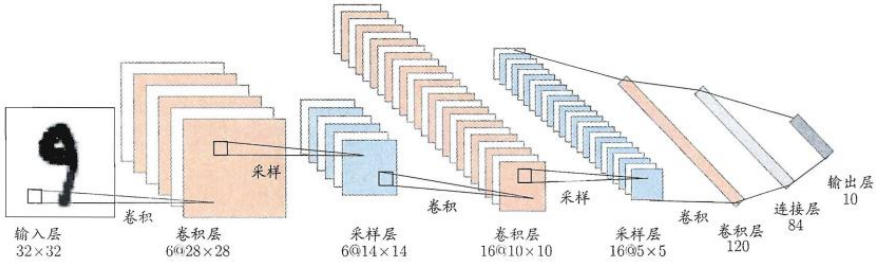


5.6 深度学习

随着大数据时代的到来，计算能力大幅提高，计算数据大幅增加，以深度学习为代表的复杂模型受到青睐。

以增加隐层数目为例。经典算法容易导致发散，可以采用无监督逐层训练法。基本思想是每次训练一层隐结点，上一层隐结点的输出作为输入，本层隐结点的输出作为下一层隐结点的输入，称为预训练。预训练完成后，再对整个网络进行微调训练。

另一种方法是权共享，让一组神经元使用相同的连接权。以卷积神经网络 CNN 为例。网络输入是一个 32×32 的手写数字图像，输出是其识别结果，CNN 复合多个卷积层和采样层对输入信号进行加工，在连接层实现与输出目标之间的映射。每个卷积层包含多个特征映射，每个特征映射是一个由多个神经元构成的平面，通过一种卷积滤波器提取输入的一种特征。采用 BP 算法进行训练，但在训练中每一组神经元都采用相同的连接权，大大减少了需要训练的参数数目。



6 支持向量机

6.1 间隔与支持向量

给定训练样本集 D={($\boldsymbol{x}_1$,y_1),…,($\boldsymbol{x}_m$,y_m)}，$y_i \in \{-1,+1\}$，最基本的想法是基于训练集 D 在样本空间中找到一个划分超平面，将不同类别的样本分开。

在样本空间中，划分超平面可通过$\boldsymbol{\omega}^T\boldsymbol{x}+b=0$来描述，记为$(\boldsymbol{\omega},b)$。样本空间任一点$\boldsymbol{x}$到超平面$(\boldsymbol{\omega},b)$的距离可写为$r=\frac{|\boldsymbol{\omega}^T\boldsymbol{x}+b|}{\|\boldsymbol{\omega}\|}$。假设超平面$(\boldsymbol{\omega},b)$能将训练样本正确分类，那么可以选择合适的$(\boldsymbol{\omega},b)$，使得$\begin{cases}\omega^T x+b\geq+1, y_i=+1\\ \omega^T x+b\leq-1, y_i=-1\end{cases}$成立。距离超平面最近的几个训练样本点使上式取等号，被称为“支持向量”。两个异类支持向量到超平面的距离之和为$\gamma=2/\|\boldsymbol{\omega}\|$，称为间隔。我们需要找到满足上式中的约束参数$(\boldsymbol{\omega},b)$使得间隔$\gamma$最大，即是$\min\limits_{\boldsymbol{\omega},b}\frac{1}{2}\|\boldsymbol{\omega}\|^2\ s.t.\ y_i(\boldsymbol{\omega}^T\boldsymbol{x}_i+b)\geq 1$。这就是支持向量机的基本型。

6.2 对偶问题

拉格朗日乘子法：$L(\boldsymbol{\omega},b,\boldsymbol{\alpha})=\frac{1}{2}\|\boldsymbol{\omega}\|^2+\sum\limits_{i=1}^{m}\alpha_i(1-y_i(\boldsymbol{\omega}^T\boldsymbol{x}_i+b))$。令 L 对$\boldsymbol{\omega}$和 b 的偏导为 0，得$\begin{cases}\boldsymbol{\omega}=\sum\limits_{i=1}^{m}\alpha_i y_i\boldsymbol{x}_i\\ 0=\sum\limits_{i=1}^{m}\alpha_i y_i\end{cases}$，得到对偶问题：

$$\max_{\boldsymbol{\alpha}}\sum_{i=1}^{m}\alpha_i-\frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j y_i y_j\boldsymbol{x}_i^T\boldsymbol{x}_j\ \ s.t.\sum_{i=1}^{m}\alpha_i y_i=0,\alpha_i\geq 0$$

解出$\boldsymbol{\alpha}$，代回得到模型$f(\boldsymbol{x})=\sum\limits_{i=1}^{m}\alpha_i y_i\boldsymbol{x}_i^T\boldsymbol{x}+b$。上述过程需满足 KKT 条件，即是

$\begin{cases}\alpha_i\geq 0\\ y_i f(\boldsymbol{x}_i)-1\geq 0\\ \alpha_i(y_i f(\boldsymbol{x}_i)-1)=0\end{cases}$。若$\alpha_i=0$，则该样本不会再模型中出现；若$y_i f(\boldsymbol{x}_i)=1$，则所对应的样本位于最大间隔边界上，是一个支持向量。这表明训练完成后，大部分训练样本都不需保留，最终模型仅与支持向量有关。

• SMO 算法

回到这个问题：$\max\limits_{\boldsymbol{\alpha}}\sum\limits_{i=1}^{m}\alpha_i-\frac{1}{2}\sum\limits_{i=1}^{m}\sum\limits_{j=1}^{m}\alpha_i\alpha_j y_i y_j\boldsymbol{x}_i^T\boldsymbol{x}_j\ \ s.t.\sum\limits_{i=1}^{m}\alpha_i y_i=0,\alpha_i\geq 0$。

算法基本思路：先固定α_i之外的所有参数，然后求α_i上的极值。在约束条件下，

我们每次选择两个变量α_i,α_j，固定其他参数，不断执行下述步骤直至收敛：
1. 选取一对需要更新的变量；2. 固定其他参数，求解获得更新后的变量。
注意到只需选取α_i,α_j中有一个不满足 KKT 条件，目标函数就会在迭代后增大。直观来看，KKT 条件违背的程度越大，则变量更新后可能导致的目标函数值增幅越大。于是，SMO 先选取违背 KKT 条件程度最大的变量。第二个变量应选择一个使目标函数值增长最快的变量，这里采用启发式，使选取的两变量所对应样本之间的间隔最大。直观来看，这两个变量差别很大，与对两个相似的变量进行更新相比，它们进行更新可以给目标函数值带来更大的变化。

6.3 核函数

在现实任务中，原始样本空间内也许并不存在一个能正确划分两类样本的超平面。我们可将样本从原始空间映射到一个更高维的特征空间，使得样本在这个特征空间内线性可分。
令$\Phi(\boldsymbol{x})$表示将 $\boldsymbol{x}$ 映射后的特征向量，在特征空间中划分超平面所对应的模型可表示为$f(\boldsymbol{x})=\boldsymbol{\omega}^T\Phi(\boldsymbol{x})+\mathrm{b}$，类似有$\min\limits_{\omega,b}\dfrac{1}{2}\|\boldsymbol{\omega}\|^2, s.t.\ y_i(\boldsymbol{\omega}^T\Phi(\boldsymbol{x}_i)+b)\geq 1$，其对偶问题是

$$\max_{\boldsymbol{\alpha}}\sum_{i=1}^{m}\alpha_i-\frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j y_i y_j\Phi(\boldsymbol{x}_i)^T\Phi(\boldsymbol{x}_j)\ s.t.\sum_{i=1}^{m}\alpha_i y_i=0,\alpha_i\geq 0\text{。}$$

定义核函数$\kappa(\boldsymbol{x}_i,\boldsymbol{x}_j)=\left\langle\Phi(\boldsymbol{x}_i),\Phi(\boldsymbol{x}_j)\right\rangle$，则上式改写为

$$\max_{\boldsymbol{\alpha}}\sum_{i=1}^{m}\alpha_i-\frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j y_i y_j\kappa(\boldsymbol{x}_i,\boldsymbol{x}_j)\ s.t.\sum_{i=1}^{m}\alpha_i y_i=0,\alpha_i\geq 0\text{。}$$

求解后得到$f(\boldsymbol{x})=\sum\limits_{i=1}^{m}\alpha_i y_i\kappa(\boldsymbol{x},\boldsymbol{x}_i)+b$。这表明模型最优解可以通过训练样本的核函数展开，称为“支持向量展式”。
核函数定理：令 X 为输入空间，$\kappa(\cdot,\cdot)$是定义在 X×X 上的对称函数，则κ是核函数当且仅当对于任意数据 D=$\{\boldsymbol{x}_1,\boldsymbol{x}_2,\cdots,\boldsymbol{x}_m\}$，“核矩阵”$\mathbf{K}$ 总是半正定的：其中有$\mathbf{K}=\left(\kappa(\boldsymbol{x}_i,\boldsymbol{x}_j)\right)_{ij}$。

事实上，任何一个核函数都能找到一个与之对应的映射Φ，即是核函数隐式地定义了一个称为“再生核希尔伯特空间”的特征空间。
常用的核函数：

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

6.4 软间隔与正则化

事实上，D 往往线性不可分。我们对每个样例$\boldsymbol{x}_i$引入松弛变量ξ_i，以使得$\boldsymbol{x}_i$满足

放宽以后的约束 $y_i(\boldsymbol{\omega}^T\boldsymbol{x}_i+b)+\xi_i\geq 1$。对于正确分类但 $y_i(\boldsymbol{\omega}^T\boldsymbol{x}_i+b)<1$ 的样例点，$0<\xi<1$；对于正好落在超平面上的样本，$\xi=1$；分类错误的，$\xi>1$。把所有$\xi\neq 0$ 的点视为特异点，于是原问题转化为下面的优化问题：

$\min_{\boldsymbol{\omega},b,\xi}\frac{1}{2}\|\boldsymbol{\omega}\|^2+C\sum_{i=1}^{N}\xi_i\ s.t.\ y_i(\boldsymbol{\omega}^T\boldsymbol{x}_i+b)\geq 1-\xi_i,\xi_i\geq 0$；其中 C 是惩罚函数。我们称这种向量机为软间隔支持向量机。其对偶问题为：

$\max_{\boldsymbol{\alpha}}\sum_{i=1}^{m}\alpha_i-\frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j y_i y_j\boldsymbol{x}_i^T\boldsymbol{x}_j\ s.t.\sum_{i=1}^{m}\alpha_i y_i=0,0\leq\alpha_i\leq C$。

与 D 线性可分时相比，目标函数完全一样，约束条件只是增加了α_i的上界。

7 贝叶斯分类器

7.1 贝叶斯决策论

在所有相关概率都已知的理想情形下，贝叶斯决策论考虑如何基于这些概率和误判损失来选择最优的类别标记。

假设有 N 种可能的类别标记，λ_{ij} 是将一个真实标记为 c_j 的样本误分类为 c_i 所产生的损失。基于后验概率 $P(c_i|\boldsymbol{x})$可获得将 $\boldsymbol{x}$ 分类为 c_i 所产生的期望损失，即在样本 $\boldsymbol{x}$ 上的“条件风险”：$R(c_i\mid\boldsymbol{x})=\sum_{j=1}^{m}\lambda_{ij}P(c_j\mid\boldsymbol{x})$。我们的任务是寻找一个判定准则 h:X→Y 以最小化总体风险 $R(h)=E_x[R(h(\boldsymbol{x})\mid\boldsymbol{x})]$。

贝叶斯判定准则：为最小化总体风险，只需在每个样本上选择那个能使条件风险 $R(c|\boldsymbol{x})$最小的类别标记 h*。此时 h*称为贝叶斯最优分类器，与之对应的总体风险 R(h*)称为贝叶斯风险。可以看出，欲使用贝叶斯判定准则来最小化决策风险，首先要获得后验概率 $P(c|\boldsymbol{x})$。主要有两种策略：

1. 判别式模型：给定 $\boldsymbol{x}$，通过直接建模 $P(c|\boldsymbol{x})$来预测 c（决策树、BP 网络）；
2. 生成式模型：联合概率分布 $P(\boldsymbol{x},c)$建模，然后再获得 $P(c|\boldsymbol{x})$。

对于生成式模型，考虑 $P(c\mid\boldsymbol{x})=\frac{P(\boldsymbol{x},c)}{P(\boldsymbol{x})}=\frac{P(c)P(\boldsymbol{x}\mid c)}{P(\boldsymbol{x})}$。其中 P(c)是类“先验”概率，$P(\boldsymbol{x}|c)$是样本 $\boldsymbol{x}$ 相对于类标记 c 的类条件概率，或称为“似然”，$P(\boldsymbol{x})$是用于归一化的“证据”因子。对给定样本 $\boldsymbol{x}$，$P(\boldsymbol{x})$与类标记无关。

类先验概率 P(c)表达了样本空间中各类样本所占的比例，根据大数定律，当训练集包含充足的独立同分布样本时，P(c)可通过各类样本出现的频率进行估计。

对类条件概率 $P(\boldsymbol{x}|c)$来说，由于它设计关于 $\boldsymbol{x}$ 所有属性的联合概率，直接根据样本出现的频率来估计将会遇到严重的困难。因为很多样本取值在训练集中根本没有出现。

7.2 极大似然估计(MLE)

估计类条件概率的一种常用策略是先假定其具有某种确定的概率分布形式，再基于训练样本对概率分布的参数进行估计。具体地，记关于类别 c 的类条件概率为 $P(\boldsymbol{x}|c)$，假设 $P(\boldsymbol{x}|c)$具有确定的形式并且被参数向量$\boldsymbol{\theta}_c$唯一确定，则我们的任务就

是利用训练集 D 估计参数$\boldsymbol{\theta}_c$。下面用 P($\boldsymbol{x}$|$\boldsymbol{\theta}_c$)来代替 P($\boldsymbol{x}$|c)。
令 D_c 表示训练集 D 种第 c 类样本组成的集合，假设这些样本是独立同分布的，则参数$\boldsymbol{\theta}_c$对于数据集 D_c 的似然是 $P(D_c \mid \boldsymbol{\theta}_c)=\prod_{\boldsymbol{x}\in D_c} P(\boldsymbol{x} \mid \boldsymbol{\theta}_c)$。这个连乘操作易造成下溢，通常使用对数似然 $LL(\boldsymbol{\theta}_c)=\sum_{\boldsymbol{x}\in D_c}\log P(\boldsymbol{x} \mid \boldsymbol{\theta}_c)$，即是 $\hat{\boldsymbol{\theta}}_c=\arg\max_{\boldsymbol{\theta}_c} LL(\boldsymbol{\theta}_c)$。

7.3 朴素贝叶斯分类器

采用“属性条件独立性假设”，即是所有属性相互独立。

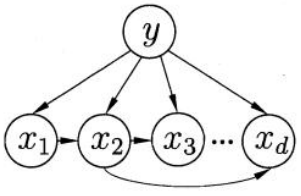
改写为 $p(c \mid \boldsymbol{x})=\frac{P(c)}{P(\boldsymbol{x})}\prod_{i=1}^{d} P(x_i \mid c)$，其中 d 是属性数目，$x_i$ 是 $\boldsymbol{x}$ 的第 i 个属性值。

基于贝叶斯判定准则有 $h_{nb}(\boldsymbol{x})=\arg\max_{c\in Y} P(c)\prod_{i=1}^{d} P(x_i \mid c)$（函数值即为类别）。

先验概率估计为 $P(c)=\frac{|D_c|}{|D|}$，条件概率估计为 $P(x_i \mid c)=\frac{|D_{c,x_i}|}{|D_c|}$。对于连续属性，考虑概率密度函数即可。需注意，**若某个属性值在训练集中没有与某个类同时出现过，判别时可能出现问题**。
为了避免其他属性携带的信息被训练集中未出现的属性值“抹去”，在估计概率值时通常要进行“平滑”，常用“拉普拉斯修正”。具体来说，令 N 表示训练集 D 中可能的类别数，N_i 表示第 i 个属性可能的取值数，则先验概率 P(c)修正为 $\hat{P}(c)=\frac{|D_c|+1}{|D|+N}$，似然 P($x_i$|c)修正为 $\hat{P}(x_i \mid c)=\frac{|D_{c,x_i}|+1}{|D_c|+N_i}$。拉普拉斯修正避免了因训练集样本不充分导致概率估值为零的问题，并且训练集充分大时，修正过程引入的影响也变得可忽略。
现实过程中朴素贝叶斯分类器有很多使用方式。如对预测速度要求较高，可以先存储概率估值，预测时只需“查表”。如任务数据更替频繁，可采用“懒惰学习”，事先不进行训练，收到预测请求后再根据当前数据集进行概率估值；若数据不断增加，可在现有估值基础上，仅对新增样本的属性值所涉及的概率估值进行计数修正，实现增量学习。

7.4 半朴素贝叶斯分类器

是朴素贝叶斯分类器对属性独立性假设进行一定程度的放松。
基本想法是适当考虑一部分属性间的相互依赖信息，从而既不需要进行完全联合概率计算，又不至于彻底忽略了比较强的属性依赖关系。“独依赖估计”(ODE)是半朴素贝叶斯分类器最常用的一种策略，即假设每个属性在类别之外最多仅依赖于一个其他属性，$P(c \mid \boldsymbol{x}) \propto P(c)\prod_{i=1}^{d} P(x_i \mid c, pa_i)$，其中 pa_i 是属性 x_i 依赖的属性，称为父属性。问题的关键是确定每个属性的父属性。



• TAN 方法
在最大带权生成树算法的基础上，通过以下步骤将属性间依赖关系约简为右图的树形结构：

1°计算任意两个属性之间的条件互信息：

$$I(x_i,x_j\mid y)=\sum_{x_i,x_j;c\in Y}P(x_i,x_j\mid c)\log\frac{P(x_i,x_j\mid c)}{P(x_i\mid c)P(x_j\mid c)}。$$

2°以属性为结点构建完全图，任意两个结点之间边的权重设为 $I(x_i,x_j\mid y)$；
3°构建此完全图的最大带权生成树，挑选根变量，将边置为有向；
4°加入类别节点 y，增加从 y 到每个属性的有向边。
条件互信息刻画了属性 x_i 和 x_j 在已知类别情况下的相关性，因此 TAN 实际上仅保留了强相关属性之间的依赖性。

• SPODE 方法

假设所有属性都依赖于同一个属性，称为“超父”，然后通过交叉验证等模型选择方法来确定超父属性。

• AODE 方法

一种基于集成学习机制、更为强大的独依赖分类器。与 SPODE 不同，AODE 尝试将每个属性作为超父来构建 SPODE，然后将那些具有足够训练数据支撑的 SPODE 集成起来作为最终结果，即 $P(c\mid \boldsymbol{x})\propto\sum_{\substack{i=1\\|D_{x_i}|\geq m'}}^{d}\left(P(c,x_i)\prod_{j=1}^{d}P(x_j\mid c,x_i)\right)$。AODE 需要估计 $P(c,x_i)$和 $P(x_j\mid c,x_i)$，有 $\hat{P}(c,x_i)=\frac{|D_{c,x_i}|+1}{|D|+N\times N_i}$ $\hat{P}(x_j\mid c,x_i)=\frac{|D_{c,x_i,x_j}|+1}{|D_{c,x_i}|+N_j}$。

7.5 贝叶斯网

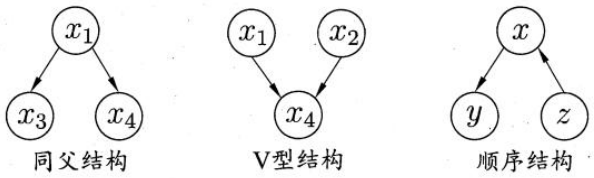
亦称信念网，借助有向无环图来刻画属性之间的依赖关系，并使用条件概率表来描述属性的联合概率分布。

一个贝叶斯网 B 由结构 G 和参数Θ两部分。网络结构 G 是一个有向无环图，每个结点对应于一个属性，若两个属性有直接依赖关系，则它们由一条边连接起来；参数Θ定量描述这种依赖关系，假设属性 x_i 在 G 中的父结点集为π_i,则Θ包含了每个属性的条件概率表 $P_B(x_i\mid\pi_i)$。

• 结构

贝叶斯网假设每个属性与它的非后裔属性独立，于是 B=<G,Θ>将属性 $x_1,\cdots,x_d$ 的联合概率分布定义为

$$P_B(x_1,\cdots,x_d)=\prod_{i=1}^{d}P_B(x_i\mid\pi_i)。$$



贝叶斯网中三个变量之间的典型依赖关系

在同父结构中，给定父结点 x_1 的取值，则 x_3 与 x_4 条件独立。在顺序结构中，给定 x 的值，则 y 与 z 条件独立。在 V 型结构中，给定子结点 x_4 的取值，则 x_1 与 x_2 必不独立；若 x_4 完全未知，则 x_1 和 x_2 却相互独立。这种独立性称为边际独立性，记为 $x_1\perp\!\!\!\perp x_2$。

为了分析有向图中变量间的条件独立性，可使用“有向分离”。我们先把有向图转变为无向图：1°找出所有图中的 V 型结构，在 V 型结构的两个父结点之间加上一条无向边；2°将所有有向边改为无向边。由此产生的无向图称为道德图，

父结点相连的过程称为道德化。基于道德图能直观、迅速地找到变量间的条件独立性。假定道德图中有变量 x,y 和变量集合 $\mathbf{z}=\{z_i\}$，若变量 x 和 y 能在图上被 $\mathbf{z}$ 分开，即从道德图中将变量集合 $\mathbf{z}$ 去除后，x 和 y 分属两个连通分支，则称变量 x 和 y 被 $\mathbf{z}$ 有向分离，$x \perp y \mid \mathbf{z}$ 成立。（道德化的蕴意：孩子的父母应建立牢靠的关系，否则是不道德的）

• 学习

贝叶斯网学习的首要任务就是根据训练数据集来找到结构最“恰当”的贝叶斯网。“评分搜索”是求解这一问题的常用办法。我们先定义一个评分函数，来评估贝叶斯网与训练数据的契合程度，然后基于评分函数寻找结构最优的贝叶斯网。

常用评分函数基于信息论准则，将学习问题看作数据压缩任务，目标是找到一个能以最短编码长度描述训练数据的模型，编码长度包括描述模型需要的编码位数和使用该模型描述数据所需的编码位数。对贝叶斯网学习而言，模型就是一个贝叶斯网，每个贝叶斯网描述了一个在训练数据上的概率分布，自有一套编码机制能使经常出现的样本有更短的编码。我们应选择综合编码长度最短的贝叶斯网，这就是最小描述长度 MDL 准则。

给定训练集 $D=\{\boldsymbol{x}_1,\cdots,\boldsymbol{x}_m\}$，贝叶斯网 $B=<G,\Theta>$在 D 上的评分函数可写为 $s(B|D)=f(\theta)|B|-LL(B|D)$，其中$|B|$是贝叶斯网的参数个数，$f(\theta)$表示描述每个参数$\theta$所需的编码位数，而 $LL(B|D)=\Sigma_{i=1,\cdots,m}\log P_B(\boldsymbol{x}_i)$是贝叶斯网的对数似然。显然，评分函数第一项是计算编码贝叶斯网 B 所需的编码位数，第二项是计算 B 所对应的概率分布 P_B 对 D 描述得有多好。于是，学习任务就转化为一个优化任务，即寻找一个贝叶斯网 B 使评分函数最小。

若贝叶斯网的网络结构 G 固定，则评分函数最小化等价于对参数Θ的极大似然估计，则可以根据 D 上的经验分布计算参数 $\theta_{x_i|\pi_i}=\hat{P}_D(x_i \mid \pi_i)$。

不幸的是，从所有可能网络结构空间搜索最优贝叶斯网结构是一个 NP 问题，难以快速求解。有两种常用的策略能在有限时间内求得近似解：第一种是贪心法，例如从某个网络结构出发，每次调整一条边，直到评分函数值不再降低；第二种是通过给网络结构施加约束来削减搜索空间，例如将网络结构限定为树形结构等。

• 推断

贝叶斯网训练好之后就能用来回答“查询”，即通过一些属性变量的观测值来推测其他属性变量的取值。通过已知变量观测值来推测待查询变量的过程称为“推断”，已知变量观测值称为“证据”。

根据贝叶斯网定义的联合概率分布来精确计算后验概率是 NP 难的。近似推断常使用吉布斯采样（一种随机采样方法）完成。

（注：也可采用变分推断）

令 $\mathbf{Q}=\{Q_1,\cdots,Q_n\}$表示待查询变量，$\mathbf{E}=\{E_1,\cdots,E_k\}$为证据变量，已知其取值为 $\mathbf{e}=\{e_1,\cdots,e_k\}$。目标是计算后验概率 $P(\mathbf{Q}=\mathbf{q}|\mathbf{E}=\mathbf{e})$。

吉布斯采样算法先随机产生一个与证据 $\mathbf{E}=\mathbf{e}$ 一致的样本 $\mathbf{q}^0$ 作为初始点，然后每步从当前样本出发产生下一个样本。具体地说，在第 t 次采样中，算法先假设 $\mathbf{q}^t=\mathbf{q}^{t-1}$，然后对非证据变量逐个进行采样改变其取值，采样概率根据贝叶斯网 B 和其他变量的当前取值计算获得，假定经过 T 次采得到的与 $\mathbf{q}$ 已知的样本共有 n_q 个，则可近似估算出后验概率 $P(\boldsymbol{Q}=\boldsymbol{q} \mid \boldsymbol{E}=\boldsymbol{e}) \approx \frac{n_q}{T}$。

输入: 贝叶斯网 $B = \langle G, \Theta \rangle$;
采样次数 T;
证据变量 $\mathbf{E}$ 及其取值 $\mathbf{e}$;
待查询变量 $\mathbf{Q}$ 及其取值 $\mathbf{q}$.

过程:

$n_q = 0$
$\mathbf{q}^0 =$ 对 $\mathbf{Q}$ 随机赋初值
for $t = 1, 2, \ldots, T$ **do**
  for $Q_i \in \mathbf{Q}$ **do**
    $\mathbf{Z} = \mathbf{E} \cup \mathbf{Q} \setminus \{Q_i\}$;
    $\mathbf{z} = \mathbf{e} \cup \mathbf{q}^{t-1} \setminus \{q_i^{t-1}\}$;
    根据 B 计算分布 $P_B(Q_i \mid \mathbf{Z} = \mathbf{z})$;
    $q_i^t =$ 根据 $P_B(Q_i \mid \mathbf{Z} = \mathbf{z})$ 采样所获 Q_i 取值;
    $\mathbf{q}^t =$ 将 $\mathbf{q}^{t-1}$ 中的 q_i^{t-1} 用 q_i^t 替换
  end for
  if $\mathbf{q}^t = \mathbf{q}$ **then**
    $n_q = n_q + 1$
  end if
end for

输出: $P(\mathbf{Q} = \mathbf{q} \mid \mathbf{E} = \mathbf{e}) \simeq \frac{n_q}{T}$

实质上，吉布斯采样是在贝叶斯网所有变量的联合状态空间与证据 **E=e** 一致的子空间中进行随机漫步，每一步仅依赖于前一步的状态，这是一个 Markov chain。在一定条件下，无论从什么初始状态开始，Markov chain 第 t 步的状态分布在 t→∞时必收敛于一个平稳分布；对吉布斯采样来说，这个分布恰好是 P(**Q**|**E**=**e**)。需要注意，若贝叶斯网中存在极端概率 0 或 1，则不能保证 Markov chain 存在平稳分布，吉布斯采样会给出错误的估计。

7.6 EM 算法

未观测变量学名是“隐变量”。令 **X** 表示已观测变量集，**Z** 表示隐变量集，Θ表示模型参数。对Θ做极大似然估计，则应最大化对数似然 LL(Θ|**X**,**Z**)=ln P(**X**,**Z**|Θ)。由于 **Z** 是隐变量，上式无法直接求解，此时我们可通过对 **Z** 计算期望，来最大化已观测数据的对数“边际似然”LL(Θ|**X**)=ln P(**X**|Θ)=ln Σ**z**P(**X**,**Z**|Θ)。

这里用 ln P(**X**|Θ),ln Σ**z**P(**X**,**Z**|Θ)表示 $\sum_{i=1}^{m} \ln p(x_i \mid \Theta), \sum_{i=1}^{m} \ln \sum_{z} p(x_i, z \mid \Theta)$。

EM(期望最大化)算法是常用的估计参数隐变量的算法，它是一种迭代式的方法，基本想法是：若参数Θ已知，则可根据训练数据推断出最优隐变量 **Z** 的值；反之，若 **Z** 的值已知，则可方便地对参数Θ做极大似然估计。

于是，以初始值Θ^0为起点，可迭代执行以下步骤直至收敛：

- 基于Θ^t推断隐变量 **Z** 的期望，记为 $\mathbf{Z}^t$;
- 基于已观测变量 **X** 和 $\mathbf{Z}^t$对参数Θ做极大似然估计，记为Θ^{t+1};

这就是 EM 算法的原型。

若我们不是取 **Z** 的期望，而是基于Θ^t计算隐变量 **Z** 的概率分布 P(**Z**|**X**,Θ^t)，则 EM 算法的两个步骤是：

- **E** 步：以当前参数Θ^t推断变量分布 P(**Z**|**X**,Θ^t)和后验概率 p(z|x_i,Θ^t)，并计算似然函数值 $Q(\Theta \mid \Theta^t) = \sum_{i=1}^{m} \sum_{z} p(z \mid x_i, \Theta^t) \ln \frac{p(z, x_i \mid \Theta)}{p(z \mid x_i, \Theta^t)}$。

• **M** 步：寻找参数最大化期望似然，即 $\Theta = \arg\max_{\Theta} Q(\Theta \mid \Theta^t)$。

即是利用当前估计的参数值来计算对数似然的期望值，然后寻找能使前面那步产生的似然期望最大化的参数值；这样反复直至收敛。

若使用梯度下降优化算法，求和的项数将随着隐变量的数目以指数级上升，会给梯度计算带来麻烦。

8 集成学习

8.1 个体与集成

集成学习(ensemble learning)通过构建并结合多个学习器来完成学习任务，有时也被称为多分类器系统、基于委员会的学习。

只包含同种类型个体学习器的集成称为是同质的，个体学习器称为基学习器，学习算法称为基学习算法；否则是异质的，个体学习器称为组件学习器，也不再用基学习算法。

要获得好的集成，个体学习器应“好而不同”，即要有一定的准确性和多样性。个体学习器的准确性和多样性是冲突的，如何产生并结合好而不同的个体学习器是集成学习研究的核心。

目前集成学习方法大致分为两大类：1°个体学习器间存在强依赖关系，必须串行生成的序列化方法；2°个体学习器不存在强依赖关系，可同时生成的并行化方法。前者代表是 Boosting，后者代表是 Bagging 和 Random Forest。

8.2 Boosting

是一族将弱学习器提升为强学习器的算法。先从初始训练集训练出一个基学习器，再根据基学习器的表现对训练样本分布进行调整，使得先前基学习器做错的训练样本在后续受到更多关注，然后基于调整后的样本分布来训练下一个基学习器；如此重复，直到基学习器数目达到事先指定的值 T，最终将这 T 个基学习器进行加权结合。

Boosting 族算法最著名的代表是 AdaBoosting，如图所示。其中 $y_i \in \{\pm 1\}$，f 是真实函数。

输入： 训练集 $D = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_m, y_m)\}$;
基学习算法 $\mathfrak{L}$;
训练轮数 T.

过程：

1: $\mathcal{D}_1(\boldsymbol{x}) = 1/m$.
2: **for** $t = 1, 2, \ldots, T$ **do**
3: $h_t = \mathfrak{L}(D, \mathcal{D}_t)$;
4: $\epsilon_t = P_{\boldsymbol{x} \sim \mathcal{D}_t}(h_t(\boldsymbol{x}) \neq f(\boldsymbol{x}))$;
5: **if** $\epsilon_t > 0.5$ **then break**
6: $\alpha_t = \frac{1}{2} \ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right)$;
7: $\mathcal{D}_{t+1}(\boldsymbol{x}) = \frac{\mathcal{D}_t(\boldsymbol{x})}{Z_t} \times \begin{cases} \exp(-\alpha_t), & \text{if } h_t(\boldsymbol{x}) = f(\boldsymbol{x}) \\ \exp(\alpha_t), & \text{if } h_t(\boldsymbol{x}) \neq f(\boldsymbol{x}) \end{cases}$
$= \frac{\mathcal{D}_t(\boldsymbol{x}) \exp(-\alpha_t f(\boldsymbol{x}) h_t(\boldsymbol{x}))}{Z_t}$
8: **end for**

输出： $H(\boldsymbol{x}) = \operatorname{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(\boldsymbol{x})\right)$

AdaBoost 算法有多种推导方式，容易理解的是基于“加性模型”，即基学习器

的线性组合 $H(\boldsymbol{x})=\sum_{i=1}^{T}\alpha_t h_t(\boldsymbol{x})$ 来最小化指数损失函数 $l_{\exp}(H\mid D)=E_{x\sim D}[e^{-f(\boldsymbol{x})H(\boldsymbol{x})}]$

若 $H(\boldsymbol{x})$能令指数损失函数最小化，则考虑

$$\frac{\partial l_{\exp}(H\mid D)}{\partial H(\boldsymbol{x})}=-e^{-H(\boldsymbol{x})}P(f(\boldsymbol{x})=1\mid\boldsymbol{x})+e^{-H(\boldsymbol{x})}P(f(\boldsymbol{x})=-1\mid\boldsymbol{x})$$

令此式为 0，则 $H(\boldsymbol{x})=\frac{1}{2}\ln\frac{P(f(\boldsymbol{x})=1\mid\boldsymbol{x})}{P(f(\boldsymbol{x})=-1\mid\boldsymbol{x})}$，因此有

$\operatorname{sgn}(H(\boldsymbol{x}))=\arg\max_{y\in\{\pm1\}}P(f(\boldsymbol{x})=y\mid\boldsymbol{x})$。

这意味着 sgn($H(\boldsymbol{x})$)达到了贝叶斯最优错误率。换言之，若指数损失函数最小化，则分类错误率最小化；这说明指数损失函数是分类任务原本 0/1 损失函数的一致的替代损失函数。

在 AdaBoost 算法中，第一个基分类器 h_1 是通过直接将基学习算法用于初始数据分布而得；此后迭代地生成 h_t 和α_t，当基分类器 h_t 基于分布 D_t 产生后，该基分类器的权重α_t 应使得$\alpha_t h_t$ 最小化指数损失函数

$l_{\exp}(\alpha_t h_t\mid D_t)=E_{\boldsymbol{x}\sim D}[e^{-f(\boldsymbol{x})\alpha_t h_t(\boldsymbol{x})}]=e^{-\alpha_t}(1-\varepsilon_t)+e^{\alpha_t}\varepsilon_t$，其中 ε_t 是错误率。

令指数损失函数导数为 0，得到 $\alpha_t=\frac{1}{2}\ln\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right)$，这恰是分类器权重更新公式。

AdaBoost 算法在获得 H_{t-1} 之后样本分布将进行调整，使下一轮的基学习器 h_t 能纠正 H_{t-1} 的一些错误，即最小化 $l_{\exp}(H_{t-1}+h_t\mid D)=E_{x\sim D}[e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})}e^{-f(\boldsymbol{x})h_t(\boldsymbol{x})}]$。

注意到 $f^2(\boldsymbol{x})=h_t^2(\boldsymbol{x})=1$，使用 Taylor 展开知

$$l_{\exp}(H_{t-1}+h_t\mid D)=E_{x\sim D}[e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})}(1-f(\boldsymbol{x})h_t(\boldsymbol{x})+\frac{1}{2})]$$

于是，理想的基学习器

$$h_t(\boldsymbol{x})=\arg\min_h l_{\exp}(H_{t-1}+h_t\mid D)=\arg\max_h E_{x\sim D}[e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})}f(\boldsymbol{x})h(\boldsymbol{x})]$$

$$=\arg\max_h E_{x\sim D}[\frac{e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})}}{E_{x\sim D}[e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})}]}f(\boldsymbol{x})h(\boldsymbol{x})]$$。令 $D_t(\boldsymbol{x})=\frac{D(\boldsymbol{x})e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})}}{E_{x\sim D}[e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})}]}$，则 $D_t(\boldsymbol{x})$ 表示一个分布。根据期望定义，有 $h_t(x)=\arg\max_h E_{x\sim D_t}[f(\boldsymbol{x})h(\boldsymbol{x})]$。

由 $f(\boldsymbol{x})h(\boldsymbol{x})=1-2\chi(f(\boldsymbol{x})\neq h(\boldsymbol{x}))$，从而 $h_t(\boldsymbol{x})=\arg\min_h E_{x\sim D_t}[\chi(f(\boldsymbol{x})\neq h(\boldsymbol{x}))]$。

由此可见，理想的 h_t 将在分布 D_t 下最小化分类误差。因此，弱分类器将基于分布 D_t 来训练，且针对 D_t 的分类误差应小于 0.5，这在一定程度上类似“残差逼近”的思想。考虑到 D_{t+1} 与 D_t 的关系，有

$D_{t+1}(\boldsymbol{x})=\frac{D(\boldsymbol{x})e^{-f(\boldsymbol{x})H_t(\boldsymbol{x})}}{E_{x\sim D}[e^{-f(\boldsymbol{x})H_t(\boldsymbol{x})}]}=D_t(\boldsymbol{x})e^{-f(\boldsymbol{x})\alpha_t h_t(\boldsymbol{x})}\frac{E_{x\sim D}[e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})}]}{E_{x\sim D}[e^{-f(\boldsymbol{x})H_t(\boldsymbol{x})}]}$，这恰是样本分布更

新公式。

于是从基于加性模型迭代式优化指数损失函数角度推导了 AdaBoost 算法。

Boosting 算法要求基学习器能对特定的数据分布进行学习，这可通过“重赋权法”（为每个训练样本重新赋予权重）实施，也可以通过“重采样法”（根据样本分布对训练集重新采样）处理。需要注意的是，训练每一轮都要检查当前生成的基学习器是否满足基本条件，一旦条件不满足，则当前基学习器被抛弃，学习过程停止，这可能导致最终集成只有很少的基学习器，性能不佳。若采用“重采样法”，则可获得“重启动”机会避免早停（抛弃当前基学习器，重新采样，再基于新样本训练），从而使得学习过程可以持续。

从偏差-方差分解角度来看，Boosting 主要关注降低偏差，因此 Boosting 能基于泛化性能相当弱的学习器构建出很强的集成。

8.3 Bagging 与随机森林

基本思想：希望集成中的个体学习器尽可能独立，并且个体学习器不能太差。

• Bagging

利用自助采样法，采样出 T 个含 m 个训练样本的采样集，基于每个采样集训练出一个基学习器。进行预测输出结合时，使用简单投票法。

输入： 训练集 $D = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_m, y_m)\}$;
基学习算法 $\mathfrak{L}$;
训练轮数 T.

过程：

1: **for** $t = 1, 2, \ldots, T$ **do**
2: $h_t = \mathfrak{L}(D, \mathcal{D}_{bs})$
3: **end for**

输出： $H(\boldsymbol{x}) = \arg\max_{y \in \mathcal{Y}} \sum_{t=1}^{T} \mathbb{I}(h_t(\boldsymbol{x}) = y)$

自助采样过程还给 Bagging 带来了另一个优点：每个基学习器只使用了初始训练集中约 63.2%的样本，剩下约 36.8%的样本可用作验证集来对泛化性能进行“保外估计”。设 D_t 表示 h_t 实际使用的训练样本集，令 $H^{oob}(\boldsymbol{x})$表示对样本 $\boldsymbol{x}$ 的包外预测，即仅考虑那些未使用 $\boldsymbol{x}$ 训练的基学习器在 $\boldsymbol{x}$ 上的预测，有

$$H^{oob}(\boldsymbol{x}) = \arg\max_{y \in Y} \sum_{t=1}^{T} \chi(h_t(\boldsymbol{x}) = y) \cdot \chi(\boldsymbol{x} \notin D_t)$$，则 Bagging 泛化误差的包外估计为

$$\varepsilon^{oob}(\boldsymbol{x}) = \frac{1}{|D|} \sum_{(\boldsymbol{x}, y) \in D} \chi(H^{oob}(\boldsymbol{x}) \neq y)$$。

从偏差-方差分解角度来看，Bagging 主要关注降低方差。

• 随机森林（Random Forest）

在以决策树为基学习器构建 Bagging 集成的基础上，在决策树的训练过程中引入随机属性选择。传统决策树在选择划分属性时选择最优属性，而在 RF 中先随机选择一个包含 k 个属性的子集，再选择一个最优的属性划分。推荐值 $k=\log_2 d$。

随机森林中基学习器的多样性不仅来自样本扰动，还来自属性扰动，使得个体学习器之间差异度增加。随机森林的训练效率往往优于 Bagging，这是因为 Bagging 使用确定型决策树，选择划分属性要对结点所有属性进行考察，而随机森林只需考察一个属性子集。

8.4 结合策略

学习器结合从 3 个方面带来好处：1°（统计）假设空间大，多个假设在训练集上达到同等性能使得泛化性能不佳，多个学习器减小这一风险；2°（计算）降低陷入局部极小的风险；3°（表示）扩大假设空间。

假定集成包含 T 个基学习器$\{h_1,\cdots,h_T\}$，本节介绍对 h_i 结合的常见策略。

• 平均法

简单平均法、加权平均法。一般来说，个体学习器性能差别较大使用加权平均，性能相近则使用简单平均。

• 投票法

绝对多数投票法：标记得票超过半数，则预测为此标记；否则拒绝。

相对多数投票法：预测为得票最多的标记。

加权投票法：与加权平均法类似。

不同类型个体学习器可能产生不同类型的 $h_i^j(\boldsymbol{x})$的值，常见有：

类标记/硬投票：$h_i^j(\boldsymbol{x})\in\{0,1\}$，即是 h_i 将样本预测为 c_j 则取值为 1，否则为 0。

类概率/软投票：$h_i^j(\boldsymbol{x})\in[0,1]$，相当于对后验概率 $P(c_j|\boldsymbol{x})$的一个估计。

• 学习法

通过另一个学习器来进行结合，如 Stacking 方法。把个体学习器称为初级学习器，用于结合的学习器称为次级学习器。

Stacking 先从初始数据集训练出初级学习器，然后“生成”一个新数据集用于训练次级学习器。在这个新数据集中，初级学习器的输出被当作样例输入特征，初始样本的标记仍被当作样例标记。我们假定出基学习器使用不同学习算法产生。

输入： 训练集 $D=\{(\boldsymbol{x}_1,y_1),(\boldsymbol{x}_2,y_2),\ldots,(\boldsymbol{x}_m,y_m)\}$;
初级学习算法 $\mathfrak{L}_1,\mathfrak{L}_2,\ldots,\mathfrak{L}_T$;
次级学习算法 $\mathfrak{L}$.

过程：

```
for t = 1,2,...,T do
    h_t = L_t(D);
end for
D' = ∅;
for i = 1,2,...,m do
    for t = 1,2,...,T do
        z_it = h_t(x_i);
    end for
    D' = D' ∪ ((z_i1, z_i2, ..., z_iT), y_i);
end for
h' = L(D');
```

输出： $H(\boldsymbol{x})=h'(h_1(\boldsymbol{x}),h_2(\boldsymbol{x}),\ldots,h_T(\boldsymbol{x}))$

为降低过拟合风险，一般通过使用交叉验证法、留一法等方式来产生次级学习器的训练样本，即是将初级学习器对留出部分的输出作为次级学习器的训练特征。研究表明，将初级学习器的输出类概率作为次级学习器的输入属性，用多相应线性回归作为次级学习算法效果较好。

8.5 多样性

• 误差-分歧分解

假设用个体学习器 $h_1,\cdots,h_T$ 通过加权平均法结合产生的集成来完成回归学习任务

f:$R^d \to R$。对示例 $\boldsymbol{x}$，定义学习器 h_i 的分歧为 $A(h_i|\boldsymbol{x})=(h_i(\boldsymbol{x})-H(\boldsymbol{x}))^2$，则集成的分歧定义为 $\overline{A}(h|\boldsymbol{x})=\sum_{i=1}^{T}\omega_i A(h_i|\boldsymbol{x})=\sum_{i=1}^{T}\omega_i(h_i(\boldsymbol{x})-H(\boldsymbol{x}))^2$ 。分歧表现了个体学习器在样本 $\boldsymbol{x}$ 上的不一致性，一定程度上反映了个体学习器的多样性。个体学习器 h_i 和集成 H 的平方误差为 $E(h_i|\boldsymbol{x})=(f(\boldsymbol{x})-h_i(\boldsymbol{x}))^2$，$E(H|\boldsymbol{x})=(f(\boldsymbol{x})-H(\boldsymbol{x}))^2$，从而 $\overline{A}(h_i|\boldsymbol{x})=\sum_{i=1}^{T}\omega_i E(h_i|\boldsymbol{x})-E(H|\boldsymbol{x})$。对 $\boldsymbol{x}$ 分布积分，得到：

$$\sum_{i=1}^{T}\omega_i\int A(h_i|\boldsymbol{x})p(\boldsymbol{x})d\boldsymbol{x}=\sum_{i=1}^{T}\omega_i\int E(h_i|\boldsymbol{x})p(\boldsymbol{x})d\boldsymbol{x}-\int E(H|\boldsymbol{x})p(\boldsymbol{x})d\boldsymbol{x}。$$

类似地，个体学习器 h_i 在全样本上的泛化误差和分歧项分别为：

$E_i=\int E(h_i|\boldsymbol{x})p(\boldsymbol{x})d\boldsymbol{x}$，$A_i=\int A(h_i|\boldsymbol{x})p(\boldsymbol{x})d\boldsymbol{x}$。

集成的泛化误差为 $E=\int E(H|\boldsymbol{x})p(\boldsymbol{x})d\boldsymbol{x}$。

令 $\overline{E}=\sum_{i=1}^{T}\omega_i E_i$ 表示个体学习器泛化误差的加权平均，$\overline{A}=\sum_{i=1}^{T}\omega_i A_i$ 表示个体学习器分歧的加权平均，则 $E=\overline{E}-\overline{A}$。这表明个体学习器准确性越高、多样性越大，则集成越好。但遗憾的是，显示很难对 $\overline{E}-\overline{A}$ 直接进行优化，因为 $\overline{A}$ 不是一个可直接操作的多样性度量。上面的推导过程只适用于回归学习，难以推广到分类学习任务上。

• 多样性度量

用于估算个体学习器的多样化程度。

给定数据集，对于二分类任务，h_i 和 h_j 预测结果列联表为

	$h_i=+1$	$h_i=-1$
$h_j=+1$	a	c
$h_j=-1$	b	d

并记 $a+b+c+d=m$。给出一些常见的多样性度量：

不合度量：$dis_{ij}=(b+c)/m \in [0,1]$。

相关系数：$\rho_{ij}=\frac{ad-bc}{\sqrt{(a+b)(a+c)(b+d)(c+d)}} \in [-1,1]$；0 无关，正值正相关，负值负相关。

Q-统计量：$Q_{ij}=\frac{ad-bc}{ad+bc}$，与相关系数符号相同，且 $|Q_{ij}| \geq |\rho_{ij}|$。

κ-统计量：p_1 是两个分类器达成一致的概率，p_2 是若两个分类器独立、达成一致的概率，估算为 $p_1=\frac{a+d}{m}$，$p_2=\frac{(a+b)(a+c)+(c+d)(b+d)}{m^2}$，则 $\kappa=\frac{p_1-p_2}{1-p_2}$。

若分类器完全一致，则κ=1；若分类器达成一致是独立性下的偶然，则κ=0。

• 多样性增强

数据样本扰动：产生不同数据子集已，利用不同数据子集训练。

输入属性扰动：从初始属性集中抽取若干个属性子集，利用不同属性子集训练。

输出表示扰动：随机改变训练样本的类标记，将分类转化为回归等等。

算法参数扰动：随机设置不同的模型参数/初值。

9 聚类

9.1 聚类任务

无监督学习。

试图将数据集中的样本划分为若干个不相交的子集，每个子集称为一个“簇”，每个簇可能对应于一些潜在的概念/类别。这些概念对聚类算法而言事先是未知的，聚类过程仅能自动形成簇结构，簇所对应的概念语义需由使用者来把握。

9.2 性能度量

聚类性能度量大致有两类，一类是将聚类结果与某个“参考模型”进行比较，称为“外部指标”；另一类是直接考察聚类结果而不利用任何参考模型，称为“内部指标”。

对数据集 D=$\{\boldsymbol{x}_1,\boldsymbol{x}_2,\cdots,\boldsymbol{x}_m\}$，假定通过聚类给出的簇划分为 C=$\{C_1,C_2,\cdots,C_k\}$，参考模型给出的簇划分为 C*=$\{C_1^*,C_2^*,\cdots,C_s^*\}$。相应地，令$\lambda$与$\lambda^*$表示与 C 和 C*的对应的簇标记向量。将样本两两配对考虑，定义

a=|SS|，SS=$\{(\boldsymbol{x}_i,\boldsymbol{x}_j)|\lambda_i=\lambda_j,\lambda_i^*=\lambda_j^*,i<j\}$；

b=|SD|，SD=$\{(\boldsymbol{x}_i,\boldsymbol{x}_j)|\lambda_i=\lambda_j,\lambda_i^*\neq\lambda_j^*,i<j\}$；

c=|DS|，DS=$\{(\boldsymbol{x}_i,\boldsymbol{x}_j)|\lambda_i\neq\lambda_j,\lambda_i^*=\lambda_j^*,i<j\}$；

d=|DD|，DD=$\{(\boldsymbol{x}_i,\boldsymbol{x}_j)|\lambda_i\neq\lambda_j,\lambda_i^*\neq\lambda_j^*,i<j\}$；

每个样本对$(\boldsymbol{x}_i,\boldsymbol{x}_j)$仅能出现在一个集合中，因此有 $a+b+c+d=m(m-1)/2$。

基于上式可以导出下面这些常用的聚类性能度量外部指标：

Jaccard 系数：$JC=\dfrac{a}{a+b+c}$；FM 指数：$FMI=\sqrt{\dfrac{a}{a+b}\cdot\dfrac{a}{a+c}}$；

Rand 指数：$RI=\dfrac{2(a+d)}{m(m-1)}$。

上述性能度量的结果均在[0,1]区间，值越大越好。

考虑聚类结果的簇划分 C=$\{C_1,C_2,\cdots,C_k\}$，定义：

$$avg(C)=\frac{2}{|C|(|C|-1)}\sum_{1\le i<j\le|C|}dist(\boldsymbol{x}_i,\boldsymbol{x}_j)；\quad diam(C)=\max_{1\le i<j\le|C|}dist(\boldsymbol{x}_i,\boldsymbol{x}_j)；$$

$$d_{\min}(C_i,C_j)=\min_{\boldsymbol{x}_i\in C_i,\boldsymbol{x}_j\in C_j}dist(\boldsymbol{x}_i,\boldsymbol{x}_j)；\quad d_{cen}(C_i,C_j)=dist(\boldsymbol{\mu}_i,\boldsymbol{\mu}_j)。$$

其中，dist(·,·)计算两个样本之间的距离；$\boldsymbol{\mu}$代表簇 C 的中心点 $\boldsymbol{\mu}=\dfrac{1}{|C|}\sum_{1\le i\le|C|}\boldsymbol{x}_i$。显然，avg(C)对应于簇 C 内样本见的平均距离，diam(C)对应于簇 C 内样本间的最远距离，$d_{min}(C_i,C_j)$对应于簇 C_i 与簇 C_j 最近样本间的距离，$d_{cen}(C_i,C_j)$对应于簇 C_i

与簇 C_j 中心点间的距离。
基于上式可导出下面这些常用的聚类性能度量内部指标：

DB 指数：$DBI=\frac{1}{k}\sum_{i=1}^{k}\max_{j\neq i}\left(\frac{avg(C_i)+avg(C_j)}{d_{cen}(C_i,C_j)}\right)$，其值越小越好；

Dunn 指数：$DI=\frac{\min\limits_{1\le i<j\le k} d_{\min}(C_i,C_j)}{\max\limits_{1\le l\le k} diam(C_l)}$，其值越大越好。

9.3 距离计算

将属性划分为“连续属性”和“离散属性”。
在讨论距离计算时，属性上是否定义了“序”关系更为重要。不能直接在属性值上计算距离的称为“无序属性”，反之为“有序属性”。

Minkowski distance：$dist_{mk}(\boldsymbol{x}_i,\boldsymbol{x}_j)=\left(\sum_{u=1}^{n}|x_{i,u}-x_{j,u}|^p\right)^{\frac{1}{p}}$，适用于有序属性。

VDM：令 $m_{u,a}$ 表示在属性 u 上取值为 a 的样本数，$m_{u,a,i}$ 表示在第 i 个样本簇中在属性 u 上取值为 a 的样本数，k 为样本簇数，则属性 u 在两个离散值 a,b 之间的 VDM 距离为 $VDM_p(a,b)=\sum_{i=1}^{k}\left|\frac{m_{u,a,i}}{m_{u,a}}-\frac{m_{u,b,i}}{m_{u,b}}\right|^p$，适用于无需属性。

将闵可夫斯基距离和 VDM 结合处理混合属性，设有 n_c 个有序属性，$n-n_c$ 个无序属性，则 $MinkovDM_p(\boldsymbol{x}_i,\boldsymbol{x}_j)=\left(\sum_{u=1}^{n_c}|x_{iu}-x_{ju}|^p+\sum_{u=n_c+1}^{n}VDM_p(x_{iu},x_{ju})\right)^{\frac{1}{p}}$。

当样本空间中不同属性重要性不同时，可使用加权距离。
通常基于某种形式的距离来定义“相似度度量”时，其未必满足距离公理，称为“非度量距离”。

9.4 原型聚类

假设聚类结构能够通过一组样本空间中具有代表性的点来刻画。
• k 均值算法

针对聚类所得簇划分 $C=\{C_1,C_2,\cdots,C_k\}$，最小化平方误差 $E=\sum_{i=1}^{k}\sum_{\boldsymbol{x}_i\in C_i}\|\boldsymbol{x}-\boldsymbol{\mu}_i\|_2^2$。这刻画了样本围绕簇均值向量的紧密程度。
最小化 E 是 NP 难问题，因此 k 均值算法采用了贪心策略，通过迭代优化来近似求解。
• 学习向量量化(LVQ)
假设样本带有类别标记，利用样本监督信息来辅助聚类。
给定样本集 $D=\{(\boldsymbol{x}_1,y_1),\cdots,(\boldsymbol{x}_m,y_m)\}$，每个样本是由 n 个属性描述的特征向量，$y_i\in Y$ 是样本的类别标记。LVQ 的目标是学得一组 n 维原型向量 $\{\boldsymbol{p}_1,\cdots,\boldsymbol{p}_q\}$，每个原型向量代表一个聚类簇，簇标记 $t_i\in Y$。
学得一组原型向量 $\{\boldsymbol{p}_1,\cdots,\boldsymbol{p}_q\}$ 后，即可实现对样本空间 X 的簇划分。对任意样本 $\boldsymbol{x}$，它将划入与其距离最近的原型向量所代表的簇中。换言之，每个原型向量 $\boldsymbol{p}_i$ 定义

了一个区域 R_i，该区域中每个样本与 $\boldsymbol{p}_i$ 的距离不大于它与其他原型向量 $\boldsymbol{p}_i'$的距离，由此形成了对样本空间 X 的簇划分{$R_1,\cdots,R_q$}，称为 Voronoi 剖分。

输入: 样本集 $D=\{\boldsymbol{x}_1,\boldsymbol{x}_2,\ldots,\boldsymbol{x}_m\}$;
聚类簇数 k.
过程:
从 D 中随机选择 k 个样本作为初始均值向量 $\{\boldsymbol{\mu}_1,\boldsymbol{\mu}_2,\ldots,\boldsymbol{\mu}_k\}$
repeat
  令 $C_i=\varnothing\ (1\leqslant i\leqslant k)$
  for $j=1,2,\ldots,m$ **do**
    计算样本 $\boldsymbol{x}_j$ 与各均值向量 $\boldsymbol{\mu}_i\ (1\leqslant i\leqslant k)$ 的距离: $d_{ji}=||\boldsymbol{x}_j-\boldsymbol{\mu}_i||_2$;
    根据距离最近的均值向量确定 $\boldsymbol{x}_j$ 的簇标记: $\lambda_j=\arg\min_{i\in\{1,2,\ldots,k\}} d_{ji}$;
    将样本 $\boldsymbol{x}_j$ 划入相应的簇: $C_{\lambda_j}=C_{\lambda_j}\bigcup\{\boldsymbol{x}_j\}$;
  end for
  for $i=1,2,\ldots,k$ **do**
    计算新均值向量: $\boldsymbol{\mu}_i'=\frac{1}{|C_i|}\sum_{\boldsymbol{x}\in C_i}\boldsymbol{x}$;
    if $\boldsymbol{\mu}_i'\neq\boldsymbol{\mu}_i$ **then**
      将当前均值向量 $\boldsymbol{\mu}_i$ 更新为 $\boldsymbol{\mu}_i'$
    else
      保持当前均值向量不变
    end if
  end for
until 当前均值向量均未更新
输出: 簇划分 $\mathcal{C}=\{C_1,C_2,\ldots,C_k\}$

k 均值算法

输入: 样本集 $D=\{(\boldsymbol{x}_1,y_1),(\boldsymbol{x}_2,y_2),\ldots,(\boldsymbol{x}_m,y_m)\}$;
原型向量个数 q, 各原型向量预设的类别标记 $\{t_1,t_2,\ldots,t_q\}$;
学习率 $\eta\in(0,1)$.
过程:
初始化一组原型向量 $\{\boldsymbol{p}_1,\boldsymbol{p}_2,\ldots,\boldsymbol{p}_q\}$
repeat
  从样本集 D 随机选取样本 $(\boldsymbol{x}_j,y_j)$;
  计算样本 $\boldsymbol{x}_j$ 与 $\boldsymbol{p}_i\ (1\leqslant i\leqslant q)$ 的距离: $d_{ji}=||\boldsymbol{x}_j-\boldsymbol{p}_i||_2$;
  找出与 $\boldsymbol{x}_j$ 距离最近的原型向量 p_{i^*}, $i^*=\arg\min_{i\in\{1,2,\ldots,q\}} d_{ji}$;
  if $y_j=t_{i^*}$ **then**
    $\boldsymbol{p}'=\boldsymbol{p}_{i^*}+\eta\cdot(\boldsymbol{x}_j-\boldsymbol{p}_{i^*})$
  else
    $\boldsymbol{p}'=\boldsymbol{p}_{i^*}-\eta\cdot(\boldsymbol{x}_j-\boldsymbol{p}_{i^*})$
  end if
  将原型向量 $\boldsymbol{p}_{i^*}$ 更新为 $\boldsymbol{p}'$
until 满足停止条件
输出: 原型向量 $\{\boldsymbol{p}_1,\boldsymbol{p}_2,\ldots,\boldsymbol{p}_q\}$

学习向量量化

• 高斯混合聚类

高斯分布： $p(\boldsymbol{x})=\dfrac{1}{(2\pi)^{\frac{n}{2}}|\Sigma|^{\frac{1}{2}}}e^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu})}$，$\boldsymbol{\mu}$是均值向量，Σ是协方差矩阵。

采用概率模型来表达聚类原型。

定义高斯混合分布： $p_M(\boldsymbol{x})=\sum_{i=1}^{k}\alpha_i p(\boldsymbol{x}\,|\,\boldsymbol{\mu}_i,\Sigma_i)$。该分布共由 k 个混合成分组成，

每个混合成分对应于一个高斯分布，α_i 称为混合系数，$\Sigma_{i=1,\ldots,n}\alpha_i=1$。
假设样本的生成过程由高斯混合分布给出：首先，根据$\alpha_1,\cdots,\alpha_k$定义的先验分布选择高斯混合成分，其中α_i为选择第 i 个混合分布的概率；然后根据被选择混合

分布的概率密度函数进行采样。

训练集 D=$\{\boldsymbol{x}_1,\cdots,\boldsymbol{x}_m\}$由上述过程生成，随机变量 $z_j\in\{1,2,\cdots,k\}$表示生成样本 $\boldsymbol{x}_j$ 的高斯混合成分。则有 $p_M(z_j=i\mid\boldsymbol{x}_j)=\dfrac{P(z_j=i)p_M(\boldsymbol{x}_j\mid z_j=i)}{p_M(\boldsymbol{x}_j)}=\dfrac{\alpha_i p(\boldsymbol{x}_j\mid\boldsymbol{\mu}_i,\Sigma_i)}{\sum_{l=1}^{k}\alpha_l(\boldsymbol{x}_j\mid\boldsymbol{\mu}_l,\Sigma_l)}$。

换言之，$p_M(z_j=i\mid\boldsymbol{x}_j)$是样本 $\boldsymbol{x}_j$ 由第 i 个高斯混合成分生成的后验概率，记为γ_{ji}。

当高斯混合分布已知时，高斯混合聚类把样本集 D 划分为 k 个簇 C=$\{C_1,\cdots,C_k\}$，每个样本的簇标记采用最大后验概率的簇标记。

给定样本集 D，利用极大似然估计确定模型参数：

$LL(D)=\ln\left(\prod_{j=1}^{m}p_M(\boldsymbol{x}_j)\right)=\sum_{j=1}^{m}\ln\left(\sum_{i=1}^{k}\alpha_i p(\boldsymbol{x}_j\mid\boldsymbol{\mu}_i,\Sigma_i)\right)$。常采用 EM 算法。

E 步：根据当前模型参数计算出后验概率γ_{ji}。

M 步：根据γ_{ji}采用极大似然法更新模型参数，即是

$\boldsymbol{\mu}_i=\dfrac{\sum_{j=1}^{m}\gamma_{ji}\boldsymbol{x}_j}{\sum_{j=1}^{m}\gamma_{ji}}$，$\Sigma_i=\dfrac{\sum_{j=1}^{m}\gamma_{ji}(\boldsymbol{x}_j-\boldsymbol{\mu}_i)(\boldsymbol{x}_j-\boldsymbol{\mu}_i)^T}{\sum_{j=1}^{m}\gamma_{ji}}$，$\alpha_i=\dfrac{1}{m}\sum_{j=1}^{m}\gamma_{ji}$。

输入： 样本集 $D=\{\boldsymbol{x}_1,\boldsymbol{x}_2,\ldots,\boldsymbol{x}_m\}$;
高斯混合成分个数 k.

过程：

初始化高斯混合分布的模型参数 $\{(\alpha_i,\boldsymbol{\mu}_i,\boldsymbol{\Sigma}_i)\mid 1\leqslant i\leqslant k\}$
repeat
  for $j=1,2,\ldots,m$ **do**
    根据式(9.30)计算 $\boldsymbol{x}_j$ 由各混合成分生成的后验概率, 即
    $\gamma_{ji}=p_{\mathcal{M}}(z_j=i\mid\boldsymbol{x}_j)\ (1\leqslant i\leqslant k)$
  end for
  for $i=1,2,\ldots,k$ **do**
    计算新均值向量: $\boldsymbol{\mu}_i'=\frac{\sum_{j=1}^{m}\gamma_{ji}\boldsymbol{x}_j}{\sum_{j=1}^{m}\gamma_{ji}}$;
    计算新协方差矩阵: $\boldsymbol{\Sigma}_i'=\frac{\sum_{j=1}^{m}\gamma_{ji}(\boldsymbol{x}_j-\boldsymbol{\mu}_i')(\boldsymbol{x}_j-\boldsymbol{\mu}_i')^{\mathrm{T}}}{\sum_{j=1}^{m}\gamma_{ji}}$;
    计算新混合系数: $\alpha_i'=\frac{\sum_{j=1}^{m}\gamma_{ji}}{m}$;
  end for
  将模型参数 $\{(\alpha_i,\boldsymbol{\mu}_i,\boldsymbol{\Sigma}_i)\mid 1\leqslant i\leqslant k\}$ 更新为 $\{(\alpha_i',\boldsymbol{\mu}_i',\boldsymbol{\Sigma}_i')\mid 1\leqslant i\leqslant k\}$
until 满足停止条件
$C_i=\varnothing\ (1\leqslant i\leqslant k)$
for $j=1,2,\ldots,m$ **do**
  根据式(9.31)确定 $\boldsymbol{x}_j$ 的簇标记 λ_j;
  将 $\boldsymbol{x}_j$ 划入相应的簇: $C_{\lambda_j}=C_{\lambda_j}\bigcup\{\boldsymbol{x}_j\}$
end for

输出： 簇划分 $\mathcal{C}=\{C_1,C_2,\ldots,C_k\}$

其中式(9.31)是指选择后验概率最大的高斯混合成分对应的簇类。

9.5 密度聚类

假设聚类结构能通过样本分布的紧密程度决定。通常情形下，密度聚类算法从样本密度的角度来考察样本之间的可连接性，并基于可连接样本不断扩展聚类簇以

获得最终结果。

DBSCAN 算法：基于一组邻域参数(ε,MinPts)来刻画样本分布的紧密程度。定义：

ε-邻域：$N_\varepsilon(\boldsymbol{x}_j) = \{\boldsymbol{x}_i \in D | dist(\boldsymbol{x}_i,\boldsymbol{x}_j) \leqslant \varepsilon\}$；

核心对象：若 $\boldsymbol{x}_j$ 的ε-邻域至少包含 MinPts 个样本，则 $\boldsymbol{x}_j$ 是一个核心对象；

密度直达：若 $\boldsymbol{x}_j$ 位于 $\boldsymbol{x}_i$ 的ε-邻域中，且 $\boldsymbol{x}_i$ 是核心对象，则称 $\boldsymbol{x}_j$ 由 $\boldsymbol{x}_i$ 密度直达；

密度可达：对 $\boldsymbol{x}_i$ 与 $\boldsymbol{x}_j$，若存在样本序列 $\boldsymbol{p}_1,\cdots,\boldsymbol{p}_n$ 其中 $\boldsymbol{p}_1=\boldsymbol{x}_i,\cdots,\boldsymbol{p}_n=\boldsymbol{x}_j$ 且 $\boldsymbol{p}_{i+1}$ 由 $\boldsymbol{p}_i$ 密度直达，则称 $\boldsymbol{x}_j$ 由 $\boldsymbol{x}_i$ 密度可达；

密度相连：对 $\boldsymbol{x}_i$ 与 $\boldsymbol{x}_j$，若存在 $\boldsymbol{x}_k$ 使得 $\boldsymbol{x}_i$ 与 $\boldsymbol{x}_j$ 均由 $\boldsymbol{x}_k$ 密度可达，则称 $\boldsymbol{x}_i$ 与 $\boldsymbol{x}_j$ 密度相连。

DBSCAN 将“簇”定义为：由密度可达关系导出的最大的密度相连的样本集合。即是给定邻域参数(ε,MinPts)，簇 $C \subset D$ 是满足以下性质的非空样本子集：

连接性：$\boldsymbol{x}_i \in C$，$\boldsymbol{x}_j \in C$，则 $\boldsymbol{x}_i$ 与 $\boldsymbol{x}_j$ 密度相连；

最大性：$\boldsymbol{x}_i \in C$，$\boldsymbol{x}_j$ 由 $\boldsymbol{x}_i$ 密度可达，则 $\boldsymbol{x}_j \in C$。

设 $\boldsymbol{x}$ 为核心对象，由 $\boldsymbol{x}$ 密度可达的所有样本组成的集合为满足连接性和最大性的簇。DBSCAN 算法先任选数据集中的一个核心对象为“种子”，再由此出发确定相应的聚类簇。

输入： 样本集 $D = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_m\}$;
邻域参数 $(\epsilon, MinPts)$.

过程：

初始化核心对象集合: $\Omega = \varnothing$
for $j = 1, 2, \ldots, m$ **do**
  确定样本 $\boldsymbol{x}_j$ 的 ϵ-邻域 $N_\epsilon(\boldsymbol{x}_j)$;
  if $|N_\epsilon(\boldsymbol{x}_j)| \geqslant MinPts$ **then**
    将样本 $\boldsymbol{x}_j$ 加入核心对象集合: $\Omega = \Omega \bigcup \{\boldsymbol{x}_j\}$
  end if
end for
初始化聚类簇数: $k = 0$
初始化未访问样本集合: $\Gamma = D$
while $\Omega \neq \varnothing$ **do**
  记录当前未访问样本集合: $\Gamma_{\text{old}} = \Gamma$;
  随机选取一个核心对象 $\boldsymbol{o} \in \Omega$, 初始化队列 $Q =< \boldsymbol{o} >$;
  $\Gamma = \Gamma \setminus \{\boldsymbol{o}\}$;
  while $Q \neq \varnothing$ **do**
    取出队列 Q 中的首个样本 $\boldsymbol{q}$;
    if $|N_\epsilon(\boldsymbol{q})| \geqslant MinPts$ **then**
      令 $\Delta = N_\epsilon(\boldsymbol{q}) \bigcap \Gamma$;
      将 Δ 中的样本加入队列 Q;
      $\Gamma = \Gamma \setminus \Delta$;
    end if
  end while
  $k = k + 1$, 生成聚类簇 $C_k = \Gamma_{\text{old}} \setminus \Gamma$;
  $\Omega = \Omega \setminus C_k$
end while

输出： 簇划分 $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$

9.6 层次聚类

试图在不同层次对数据集进行划分，从而形成树形的聚类结构。可采用“自底向上”的聚合策略，也可采用“自顶向下”的分拆策略。

AGENS 是一种自底向上聚合策略的层次聚类算法。它先将数据集中的每个样本看作一个初始聚类簇，然后在算法运行的每一步中找出距离最近的两个聚类簇进行合并，直至达到预设的聚类簇个数。实际上，每个簇是一个样本集合，只需采用关于集合的某种距离即可。

最小距离：$d_{min}(C_i,C_j)=\min \text{dist}(\boldsymbol{x}_i,\boldsymbol{x}_j)$，$\boldsymbol{x}_i \in C_i,\boldsymbol{x}_j \in C_j$。

最大距离：$d_{max}(C_i,C_j)=\max \text{dist}(\boldsymbol{x}_i,\boldsymbol{x}_j)$，$\boldsymbol{x}_i \in C_i,\boldsymbol{x}_j \in C_j$。

平均距离：$d_{avg}(C_i,C_j)=\dfrac{1}{|C_i||C_j|}\sum_{\boldsymbol{x}_i \in C_i}\sum_{\boldsymbol{x}_j \in C_j} dist(\boldsymbol{x}_i,\boldsymbol{x}_j)$。

输入： 样本集 $D=\{\boldsymbol{x}_1,\boldsymbol{x}_2,\ldots,\boldsymbol{x}_m\}$;
聚类簇距离度量函数 d;
聚类簇数 k.

过程：

```
for j = 1,2,...,m do
    C_j = {x_j}
end for
for i = 1,2,...,m do
    for j = 1,2,...,m do
        M(i,j) = d(C_i,C_j);
        M(j,i) = M(i,j)
    end for
end for
设置当前聚类簇个数: q = m
while q > k do
    找出距离最近的两个聚类簇 C_i* 和 C_j*;
    合并 C_i* 和 C_j*: C_i* = C_i* ∪ C_j*;
    for j = j*+1, j*+2, ..., q do
        将聚类簇 C_j 重编号为 C_{j-1}
    end for
    删除距离矩阵 M 的第 j* 行与第 j* 列;
    for j = 1,2,...,q-1 do
        M(i*,j) = d(C_i*,C_j);
        M(j,i*) = M(i*,j)
    end for
    q = q-1
end while
```

输出： 簇划分 $\mathcal{C}=\{C_1,C_2,\ldots,C_k\}$

当聚类簇距离分别使用 d_{min},d_{max},d_{avg} 计算时，AGNES 算法被称为单链接、全链接或均链接算法。

10 降维与度量学习

10.1 k 近邻学习

给定测试样本，基于某种距离度量找出训练集中与其最靠近的 k 个训练样本，然后基于 k 个“邻居”的信息来进行预测；可使用投票法或平均法。

是懒惰学习的著名代表，训练时间开销为零，收到样本再进行处理。相反，在训

练阶段就对样本进行学习处理的方法称为急切学习。

对 k=1 的结果进行讨论，设最近邻样本为 $\boldsymbol{z}$，则 $P(err)=1-\sum_{c\in Y}P(c\mid \boldsymbol{x})P(c\mid \boldsymbol{z})$。假设样本独立同分布且任意样本 $\boldsymbol{x}$ 和任意小正数δ，在 $\boldsymbol{x}$ 附近δ距离范围内总能找到训练样本 $\boldsymbol{z}$，则有 $P(err)\leqslant 2\times(1-P(c^*|\boldsymbol{x}))$，即是其泛化错误率不超过贝叶斯最优分类器错误率的两倍。

10.2 低维嵌入

上述讨论基于重要假设：任意测试样本 $\boldsymbol{x}$ 附近任意小距离内总能找到训练样本，称为“密采样”。然而随着属性维数上升，密采样条件所需的样本是无法达到的天文数字，且高维空间距离计算非常困难。在高维情形下出现的数据样本稀疏、距离计算困难等问题，被称为“维数灾难”。

缓解维数灾难的重要途径是降维，即通过某种数学变换将原始高维属性空间转变为低维“子空间”，在这个子空间中样本密度大幅提高，距离计算也更加容易。若要求在低维空间中保持原始空间中样本之间的距离，即得到多维缩放 MDS。

假定 m 个样本在原始空间的距离矩阵为 $D\in R^{m\times m}$，我们希望获得样本在 d'维空间的表示 $Z\in R^{d'\times m}$，且$\|\boldsymbol{z}_i-\boldsymbol{z}_j\|=dist_{ij}$。

$B=Z^TZ\in R^{m\times m}$ 为降维后的内积矩阵，$b_{ij}=\boldsymbol{z}_i^T\boldsymbol{z}_j$，则 $dist_{ij}^2=\|\boldsymbol{z}_i\|^2+\|\boldsymbol{z}_j\|^2-2\boldsymbol{z}_i\boldsymbol{z}_j=b_{ii}+b_{jj}-2b_{ij}$。

为便于讨论，令降维后的样本 Z 被中心化，即是 $\sum_{i=1}^{m}\boldsymbol{z}_i=0$。显然，矩阵 B 的行与列之和均为零，即 $\sum_{i=1}^{m}b_{ij}=\sum_{j=1}^{m}b_{ij}=0$。从而成立：

$$\sum_{i=1}^{m}dist_{ij}^2=tr(B)+mb_{jj},\quad \sum_{j=1}^{m}dist_{ij}^2=tr(B)+mb_{ii},\quad \sum_{i=1}^{m}\sum_{j=1}^{m}dist_{ij}^2=2m\cdot tr(B)。$$

令 $dist_{i\cdot}^2=\frac{1}{m}\sum_{j=1}^{m}dist_{ij}^2$，$dist_{\cdot j}^2=\frac{1}{m}\sum_{i=1}^{m}dist_{ij}^2$，$dist_{\cdot\cdot}^2=\frac{1}{m^2}\sum_{i=1}^{m}\sum_{j=1}^{m}dist_{ij}^2$，得到

$b_{ij}=-\frac{1}{2}(dist_{ij}^2-dist_{i\cdot}^2-dist_{\cdot j}^2+dist_{\cdot\cdot}^2)$。于是可以通过距离矩阵 D 计算内积矩阵 B。

对矩阵 B 做特征值分解，$B=V\Lambda V^T$，其中Λ是特征值构成的对角矩阵，且$\lambda_1\geqslant\lambda_2\geqslant\cdots\geqslant\lambda_d$，V 是特征向量矩阵。假定其中有 d*个非零特征值，它们构成对角矩阵$\Lambda^*=diag\{\lambda_1,\cdots,\lambda_{d^*}\}$，令 V*表示相应的特征向量矩阵，则 $Z=\Lambda^{*1/2}V^{*T}\in R^{d^*\times m}$。

现实应用中为了有效降维，往往仅需降维后的距离与原始空间的距离尽可能接近，而不必严格相等。此时可取 d'<<d 个最大特征值构成的对角矩阵Λ'，对应的特征向量矩阵为 V'，则 $Z=\Lambda'^{1/2}V'^T\in R^{d'\times m}$。下图是 MDS 算法的详细描述。

输入： 距离矩阵 $\mathbf{D}\in\mathbb{R}^{m\times m}$，其元素 $dist_{ij}$ 为样本 $\boldsymbol{x}_i$ 到 $\boldsymbol{x}_j$ 的距离；
低维空间维数 d'.

过程：

1: 根据式(10.7)~(10.9)计算 $dist_{i\cdot}^2$, $dist_{\cdot j}^2$, $dist_{\cdot\cdot}^2$;
2: 根据式(10.10)计算矩阵 $\mathbf{B}$;
3: 对矩阵 $\mathbf{B}$ 做特征值分解;
4: 取 $\tilde{\mathbf{\Lambda}}$ 为 d' 个最大特征值所构成的对角矩阵，$\tilde{\mathbf{V}}$ 为相应的特征向量矩阵.

输出： 矩阵 $\tilde{\mathbf{V}}\tilde{\mathbf{\Lambda}}^{1/2}\in\mathbb{R}^{m\times d'}$，每行是一个样本的低维坐标

下面介绍基于线性变换来进行降维的方法，即线性降维方法。
给定 d 维空间中的样本 $X=\{\boldsymbol{x}_1,\cdots,\boldsymbol{x}_m\}\in R^{d\times m}$，变换之后得到 d'≤d 维空间的中的样本 $Z=W^TX$，其中 $W\in R^{d\times d'}$是变换矩阵，Z 是样本在新空间中的表达。
变换矩阵 W 可视为 d'个 d 维向量，$\boldsymbol{z}_i=W^T\boldsymbol{x}_i$ 是第 i 个样本与这 d'个基向量分别做内积得到的 d'维属性向量。可以认为 $\boldsymbol{z}_i$ 是原属性向量 $\boldsymbol{x}_i$ 在新坐标系$\{\boldsymbol{\omega}_1,\cdots,\boldsymbol{\omega}_{d'}\}$中的坐标向量。若$\boldsymbol{\omega}_i$ 和$\boldsymbol{\omega}_j$ 正交，则 W 维正交变换。新空间中的属性是原空间中属性的线性组合。

10.3 主成分分析(PCA)

利用一个超平面对所有样本进行恰当的表达，希望具有：
最近重构性：样本点到这个超平面的距离都足够近；
最大可分性：样本点在这个超平面上的投影尽可能分开。
先考虑最近重构性。
假设数据样本$\Sigma_i\boldsymbol{x}_i=0$，投影变换后得到的新坐标系为$\{\boldsymbol{\omega}_1,\cdots,\boldsymbol{\omega}_d\}$，其中$\boldsymbol{\omega}_i$ 是标准正交基。若丢弃心坐标系中的部分坐标，使维度降低到 $d'<d$，则样本点 $\boldsymbol{x}_i$ 在低维坐标系中的投影是 $\boldsymbol{z}_i=\{z_{i1},\cdots,z_{id'}\}$，其中 $z_{ij}=\boldsymbol{\omega}_j^T\boldsymbol{x}_i$ 是 $\boldsymbol{x}_i$ 在低维坐标系下第 j 维的坐标，且 $\hat{\boldsymbol{x}}_i=\sum_{j=1}^{d'}z_{ij}\boldsymbol{\omega}_j$。考虑整个训练集样本点到投影重构样本点之间的距离：

$$\sum_{i=1}^{m}\left\|\sum_{j=1}^{d'}z_{ij}\boldsymbol{\omega}_j-\boldsymbol{x}_i\right\|_2^2=\sum_{i=1}^{m}\boldsymbol{z}_i^T\boldsymbol{z}_i-2\sum_{i=1}^{m}\boldsymbol{z}_i^TW^T\boldsymbol{x}_i+const\propto -tr\left(W^T\left(\sum_{i=1}^{m}\boldsymbol{x}_i\boldsymbol{x}_i^T\right)W\right),$$

从而得到优化问题：

$$\min_W -tr\left(W^TXX^TW\right),s.t.W^TW=I。$$

再考虑最大可分性。
样本点 $\boldsymbol{x}_i$ 在新空间中超平面上的投影是 $W^T\boldsymbol{x}_i$，希望使投影后样本点的方差最大化。投影后样本点的协方差矩阵是$\Sigma_iW^T\boldsymbol{x}_i\boldsymbol{x}_i^TW$，于是得到同样的优化问题：

$$\max_W tr\left(W^TXX^TW\right),s.t.W^TW=I。$$

这便是主成分分析的优化目标。

利用拉格朗日乘子法得到 $XX^T\boldsymbol{\omega}_i=\lambda_i\boldsymbol{\omega}_i$。于是只需对协方差 XX^T 进行特征值分解，将特征值排序$\lambda_1\geqslant\cdots\geqslant\lambda_d$，取前 d'个特征值对应的特征向量 $W^*=\{\boldsymbol{\omega}_1,\cdots,\boldsymbol{\omega}_{d'}\}$。这就是主成分分析的解。

输入： 样本集 $D=\{\boldsymbol{x}_1,\boldsymbol{x}_2,\ldots,\boldsymbol{x}_m\}$;
低维空间维数 d'.
过程：
1: 对所有样本进行中心化: $\boldsymbol{x}_i\leftarrow\boldsymbol{x}_i-\frac{1}{m}\sum_{i=1}^{m}\boldsymbol{x}_i$;
2: 计算样本的协方差矩阵 $\mathbf{XX}^{\mathrm{T}}$;
3: 对协方差矩阵 $\mathbf{XX}^{\mathrm{T}}$ 做特征值分解;
4: 取最大的 d' 个特征值所对应的特征向量 $\boldsymbol{w}_1,\boldsymbol{w}_2,\ldots,\boldsymbol{w}_{d'}$.
输出： 投影矩阵 $\mathbf{W}=(\boldsymbol{w}_1,\boldsymbol{w}_2,\ldots,\boldsymbol{w}_{d'})$.

注：现实中常通过对 X 奇异值分解来代替协方差矩阵的特征值分解。
降维后低维空间的维数 d'通常由用户事先指定，或通过在 d'值不同的低维空间中对 k 近邻分类器进行交叉验证来选取较好的 d'值。对于 PCA，还可以从重构的角

度设置一个重构阈值 t，使得 $\frac{\sum_{i=1}^{d'}\lambda_i}{\sum_{i=1}^{d}\lambda_i} \geq t$。

最小的 $d-d'$个特征值的特征向量被舍弃了，但这些舍弃往往是必要的。一是使得样本采样密度增大，二是这些信息往往与噪声有关，一定程度上可以去噪。

10.4 核化线性降维

非线性降维的一种常用方法，基于核技巧对线性降维方法进行“核化”。下面介绍核主成分分析 KPCA。

假定我们将在高维特征空间中把数据投影到由 $W=\{\boldsymbol{w}_1,\cdots,\boldsymbol{w}_d\}$确定的超平面上，则对于 $\boldsymbol{w}_j$，成立$\left(\sum_{i=1}^{m}\boldsymbol{z}_i\boldsymbol{z}_i^T\right)\boldsymbol{w}_j=\lambda_j\boldsymbol{w}_j$，其中 $\boldsymbol{z}_i$ 是样本点 $\boldsymbol{x}_i$ 在高维特征空间的像。

计算知 $\boldsymbol{w}_j=\sum_{i=1}^{m}\boldsymbol{z}_i\alpha_i^j$，其中 $\alpha_i^j=\frac{1}{\lambda_j}\boldsymbol{z}_i^T\boldsymbol{w}_j$ 是 $\boldsymbol{\alpha}_i$ 的第 j 个分量。假定 $\boldsymbol{z}_i$ 是由原始属性空间中的样本点 $\boldsymbol{x}_i$ 通过映射Φ产生，引入核函数 $\kappa(\boldsymbol{x}_i,\boldsymbol{x}_j)=\Phi(\boldsymbol{x}_i)^{\mathrm{T}}\Phi(\boldsymbol{x}_j)$，则化简后得到 $\mathrm{K}\boldsymbol{\alpha}^j=\lambda_j\boldsymbol{\alpha}^j$，其中 K 为 κ 对应的核矩阵，$\boldsymbol{\alpha}^j=(\alpha_1^j,\cdots,\alpha_m^j)$。解特征值分解问题，取 K 最大的 d'个特征值对应的特征向量。对新样本 $\boldsymbol{x}$，其投影后的第 j 维坐标为

$z_j=\boldsymbol{w}_j^{\mathrm{T}}\Phi(\boldsymbol{x})=\sum_{i-1}^{m}\alpha_i^j\boldsymbol{\phi}(\boldsymbol{x}_i)^T\boldsymbol{\phi}(\boldsymbol{x})=\sum_{i-1}^{m}\alpha_i^j\kappa(\boldsymbol{x}_i,\boldsymbol{x})$，其中 $\boldsymbol{\alpha}_i$ 已经经过规范化。为获得投影后的坐标，KPCA 需对所有样本求和，因此计算开销较大。

10.5 流形学习

流行是局部与欧氏空间同胚的空间，具有欧氏空间的性质，能用欧氏距离来进行计算。可以容易地在局部建立将为映射关系，再设法将局部映射关系推广道全局。当维数被降至二维或三维时，可以进行可视化展示。

• 等度量映射 Isomap

认为低维流形嵌入到高维空间之后，直接在高维空间中计算直线距离具有误导性。可以利用流形在局部上与欧氏空间同胚这个性质，对每个点基于欧氏距离找出其近邻点，然后就能建立一个近邻连接图，图中近邻点之间存在连接，而非近邻点之间不存在连接。于是，计算两点之间测地线距离的问题，就转变为计算近邻连接图上两点之间的最短路径问题。采用 Dijkstra 算法和 Floyd 算法就可以得到两点距离，再采用 10.2 节的 MDS 方法来获取低维空间坐标。

输入： 样本集 $D=\{\boldsymbol{x}_1,\boldsymbol{x}_2,\ldots,\boldsymbol{x}_m\}$;
近邻参数 k;
低维空间维数 d'.

过程：

```
for i = 1,2,...,m do
    确定 x_i 的 k 近邻;
    x_i 与 k 近邻点之间的距离设置为欧氏距离, 与其他点的距离设置为无穷大;
end for
调用最短路径算法计算任意两样本点之间的距离 dist(x_i, x_j);
将 dist(x_i, x_j) 作为 MDS 算法的输入;
return MDS 算法的输出
```

输出： 样本集 D 在低维空间的投影 $Z=\{\boldsymbol{z}_1,\boldsymbol{z}_2,\ldots,\boldsymbol{z}_m\}$.

Isomap 仅是得到了训练样本在低维空间的坐标，对于新样本，将其映射到低维空间的办法是训练样本的高维空间坐标作为输入、低维空间坐标作为输出，训练一个回归学习器对新样本的低维空间坐标进行预测。

对近邻图的构建通常有两种做法。一种是指定近邻点个数，例如欧氏距离最近的 k 个点为近邻点得到 k 近邻图，另一种是指定距离阈值ε，距离小于ε的点认为是近邻点，得到ε近邻图。

• 局部线性嵌入

试图保持邻域内样本之间的线性关系，即使样本点 $\boldsymbol{x}_i$ 的坐标能通过它的邻域样本 $\boldsymbol{x}_j,\boldsymbol{x}_k,\boldsymbol{x}_l$ 的坐标通过线性组合出来，LLE 希望这种关系在低维空间中得以保持。

LLE 先为每个样本 $\boldsymbol{x}_i$ 找到其近邻下标集合 Q_i，然后计算出基于 Q_i 中的样本点对 $\boldsymbol{x}_i$ 进行线性重构的系数 $\boldsymbol{w}_i$：$\min\limits_{\boldsymbol{w}_1,\cdots,\boldsymbol{w}_m}\sum\limits_{i=1}^{m}\left\|\boldsymbol{x}_i-\sum\limits_{j\in Q_j}w_{ij}\boldsymbol{x}_j\right\|_2^2, s.t.\sum\limits_{j\in Q_j}w_{ij}=1$，其中 $\boldsymbol{x}_i$ 和 $\boldsymbol{x}_j$ 均为已知，令 $C_{jk}=(\boldsymbol{x}_i-\boldsymbol{x}_j)^{\mathrm{T}}(\boldsymbol{x}_i-\boldsymbol{x}_j)$，则 w_{ij} 有闭式解 $w_{ij}=\dfrac{\sum\limits_{k\in Q_i}C_{jk}^{-1}}{\sum\limits_{l,s\in Q_i}C_{ls}^{-1}}$。LLE 在低维空间中保持 $\boldsymbol{w}_i$ 不变，于是低维空间坐标 $\boldsymbol{z}_i$ 表示为 $\min\limits_{\boldsymbol{z}_1,\cdots,\boldsymbol{z}_m}\sum\limits_{i=1}^{m}\left\|\boldsymbol{z}_i-\sum\limits_{j\in Q_j}w_{ij}\boldsymbol{z}_j\right\|_2^2$。令 $Z=\{\boldsymbol{z}_1,\cdots,\boldsymbol{z}_m\}$，$(W)_{ij}=w_{ij}$，$M=(I-W)^{\mathrm{T}}(I-W)$，则可重写为 $\min\limits_{Z}tr(ZMZ^T), s.t. ZZ^T=I$。可通过特征值分解求解：M 最小的 d' 个特征值对应的特征向量组成的矩阵即为 Z^{T}。

输入： 样本集 $D=\{\boldsymbol{x}_1,\boldsymbol{x}_2,\ldots,\boldsymbol{x}_m\}$;
近邻参数 k;
低维空间维数 d'.

过程：

```
for i = 1,2,...,m do
    确定 x_i 的 k 近邻;
    从式(10.27)求得 w_ij, j ∈ Q_i;
    对于 j ∉ Q_i, 令 w_ij = 0;
end for
从式(10.30)得到 M;
对 M 进行特征值分解;
return M 的最小 d' 个特征值对应的特征向量
```

输出： 样本集 D 在低维空间的投影 $Z=\{\boldsymbol{z}_1,\boldsymbol{z}_2,\ldots,\boldsymbol{z}_m\}$.

10.6 度量学习

降维目的：找到合适低维空间，进行学习性能比原始空间性能更好。实际上每个空间对应了在样本属性上定义的一个距离度量，寻找合适空间就是在寻找一个合适的距离度量。

对于两个 d 维样本 $\boldsymbol{x}_i$ 和 $\boldsymbol{x}_j$，它们之间的平方欧氏距离可写为 $\mathrm{dist}_{\mathrm{ed}}^2(\boldsymbol{x}_i,\boldsymbol{x}_j)=\|\boldsymbol{x}_i-\boldsymbol{x}_j\|_2^2=\mathrm{dist}_{\mathrm{ij},1}^2+\mathrm{dist}_{\mathrm{ij},2}^2+\cdots+\mathrm{dist}_{\mathrm{ij},d}^2$，其中 $\mathrm{dist}_{\mathrm{ij},k}$ 表示 $\boldsymbol{x}_i$ 与 $\boldsymbol{x}_j$ 在第 k 维上的距离。若假定不同属性重要性不同，则可引入属性权重 $\boldsymbol{w}$，得到 $\mathrm{dist}_{\mathrm{wed}}^2(\boldsymbol{x}_i,\boldsymbol{x}_j)=(\boldsymbol{x}_i-\boldsymbol{x}_j)^{\mathrm{T}}\mathrm{W}(\boldsymbol{x}_i-\boldsymbol{x}_j)$，其中 W 是对角矩阵。

再往前走一步，往往属性之间相关，可将 W 替换为半正定对称矩阵 M，得到马氏距离：$\mathrm{dist}_{\mathrm{mah}}^2(\boldsymbol{x}_i,\boldsymbol{x}_j)=(\boldsymbol{x}_i-\boldsymbol{x}_j)^{\mathrm{T}}\mathrm{M}(\boldsymbol{x}_i-\boldsymbol{x}_j)=\|\boldsymbol{x}_i-\boldsymbol{x}_j\|_{\mathrm{M}}^2$。M 称为度量矩阵，度量学习即是对 M 进行学习。

对 M 进行学习要设置一个目标，可将 M 嵌入到评价指标去。

以近邻成分分析 NCA 为例。将近邻分类器投票法换为概率投票法。对于任意样本 $\boldsymbol{x}_j$，它对 $\boldsymbol{x}_i$ 分类结果影响概率为

$$p_{ij}=\frac{\exp(-\|\boldsymbol{x}_i-\boldsymbol{x}_j\|_{\mathrm{M}}^2)}{\sum_l \exp(-\|\boldsymbol{x}_i-\boldsymbol{x}_l\|_{\mathrm{M}}^2)}。$$

$\boldsymbol{x}_j$ 对 $\boldsymbol{x}_i$ 的影响随着它们之间距离的增大而减小。若以留一法正确率的最大化为目标，则可计算 $\boldsymbol{x}_i$ 被自身之外的所有样本正确分类的概率为 $p_i=\sum_{j\in\Omega_i}p_{ij}$，其中 Ω_i 表示与 $\boldsymbol{x}_i$ 属于相同类别的样本的下标集合。于是，整个样本集上的留一法正确率为 $\sum_{i=1}^{m}p_i=\sum_{i=1}^{m}\sum_{j\in\Omega_i}p_{ij}$，NCA 的优化目标为 $\min_P 1-\sum_{i=1}^{m}\sum_{j\in\Omega_i}\frac{\exp(-\|P^T\boldsymbol{x}_i-P^T\boldsymbol{x}_j\|_2^2)}{\sum_l\exp(-\|P^T\boldsymbol{x}_i-P^T\boldsymbol{x}_l\|_2^2)}$，

其中 $\mathrm{M}=\mathrm{PP}^{\mathrm{T}}$。求解上式即可得到最大近邻分类器 LOO 正确率的距离度量矩阵 M。

我们还可以在度量学习中引入领域知识。若已知某些样本相似、某些样本不相似，则可定义“必连”约束集合和 M 与“勿连”约束集合 C，$(\boldsymbol{x}_i,\boldsymbol{x}_j)\in M$ 表示 $\boldsymbol{x}_i$ 与 $\boldsymbol{x}_j$ 相似，$(\boldsymbol{x}_i,\boldsymbol{x}_j)\in C$ 表示 $\boldsymbol{x}_i$ 与 $\boldsymbol{x}_k$ 不相似。我们希望相似的样本之间距离较小，不相似的样本之间距离较大，于是可以通过求解下面这个凸优化问题获得适当的度量矩阵 M：$\min_{\mathrm{M}}\sum_{(\boldsymbol{x}_i,\boldsymbol{x}_j)\in M}\|\boldsymbol{x}_i-\boldsymbol{x}_j\|_{\mathrm{M}}^2, s.t.\ \sum_{(\boldsymbol{x}_i,\boldsymbol{x}_j)\in C}\|\boldsymbol{x}_i-\boldsymbol{x}_j\|_{\mathrm{M}}^2\geq 1, \mathrm{M}\succeq 0$，其中约束 $\mathrm{M}\succeq 0$ 表示 M 必须是半正定的。此式要求在不相似样本间的距离不小于 1 的前提下，使相似样本间的距离尽可能小。

不同度量学习方法针对不同目标获得“好”的半正定对称距离度量矩阵 M。若 M 是一个低秩矩阵，则可通过 M 进行特征值分解，总能找到一组正交基，其正交基数目为矩阵 M 的秩 rank(M)，小于原属性数 d。于是，度量学习学得的结果可衍生出一个降维矩阵 $\mathrm{P}\in\mathrm{R}^{\mathrm{d}\times\mathrm{rank(M)}}$，能用于降维之目的。

11 特征选择与稀疏学习

11.1 子集搜索与评价

将属性称为特征，对当前学习任务有用的属性称为相关特征，否则称为无关特征。从给定的特征集合中选择特征子集的过程，称为特征选择。

能从其他特征推演出来的称为冗余特征。本章假定数据中不涉及冗余特征，并且假定初始的特征集合包含了所有重要信息。

采用先产生一个候选子集，评价出好坏，然后基于评价结果产生下一个候选子集的方法。

子集搜索：给定特征集合 $\{a_1,\cdots,a_d\}$，将每个特征看作一个候选子集，对这 d 个候选单特征子集进行评价，假定 $\{a_2\}$ 最优，于是将 $\{a_2\}$ 作为第一轮的选定集；然后，在上一轮的选定集中加入一个特征，构成包含两个特征的候选子集，假定在这 $d-1$ 个候选两特征子集中 $\{a_2,a_4\}$ 最优，且优于 $\{a_2\}$，于是将 $\{a_2,a_4\}$ 作为本轮的选定集；……假定在第 $k+1$ 轮时，最优的候选 $(k+1)$ 特征子集不如上一轮的选定基，则停止生成候选子集，并将上一轮选定的 k 特征子集作为特征选择结果。这样逐

渐增加相关特征的策略称为“前向”搜索。类似有“后向”搜索、“双向”搜索。
子集评价：给定数据集 D，假定 D 中第 i 类样本所占的比例为 p_i。为便于讨论，假定 D 中第 i 类样本属性均为离散型。对属性子集 A，假定根据其取值将 D 分成了 V 个子集$\{D^1,\cdots,D^V\}$，每个子集在 A 上取值相同，信息增益 $Gain(A)=Ent(D)-\Sigma_{v=1,\cdots,V}|D^v|/|D|\cdot Ent(D^v)$。信息增益 Gain(A)越大，意味着特征子集 A 包含的有助于分类的信息越多。于是，对每个候选特征子集，我们可基于训练数据集 D 来计算其信息增益，以此作为评价准则。

11.2 过滤式选择

先对数据集进行特征选择，再训练学习器，特征选择过程与后续学习器无关。
Relief 方法设计了一个“相关统计量”来度量特征的重要性。该统计量是一个向量，其每个分量分别对应于一个初始特征，而特征子集的重要性则是由子集中每个特征所对应的相关统计量分量之和来决定。
Relief 的关键是确定相关统计量。给定训练集$\{(\boldsymbol{x}_1,y_1),\cdots,(\boldsymbol{x}_m,y_m)\}$，对每个示例 $\boldsymbol{x}_i$，Relief 先在 $\boldsymbol{x}_i$ 的同类样本中寻找其最近邻 $\boldsymbol{x}_{i,nh}$，称为“猜中近邻”，再从 $\boldsymbol{x}_i$ 的异类样本中寻找其最近邻 $\boldsymbol{x}_{i,nm}$，称为“猜错近邻”。然后，相关统计量对应于属性 j 的分量为 $\delta^j=\sum_i -diff(x_i^j,x_{i,nh}^j)^2+diff(x_i^j,x_{i,nm}^j)^2$，其中 $diff(x_a^j,x_b^j)$ 取决于属性 j 的类型：若为离散型，则相等为 0，否则为 1；若为连续型，则取欧氏距离，注意规范到[0,1]区间。
实际上 Relief 只需在数据集的采样上而不必在整个数据集上估计相关统计量。
Relief 是为二分类问题设计的，其扩展变体 Relief-F 能处理多分类问题。假定数据集 D 中的样本来自|Y|个类别。对示例 $\boldsymbol{x}_i$，若它属于第 k 类，则 Relief-F 先在第 k 类的样本中寻找 x_i 的最近邻示例 $\boldsymbol{x}_{i,nh}$ 并将其作为猜中近邻，然后在第 k 类之外的每个类中找到一个 $\boldsymbol{x}_i$ 的最近邻示例作为猜错近邻，记为 $\boldsymbol{x}_{i,l,nm}$。于是，相关统计量对应于属性 j 的分量为 $\delta^j=\sum_i -diff(x_i^j,x_{i,nh}^j)^2+\sum_{l\neq k} p_l\times diff(x_i^j,x_{i,l,nm}^j)^2$，其中 p_l 为第 l 类样本在数据集 D 中所占的比例。

11.3 包裹式选择

输入： 数据集 D;
特征集 A;
学习算法 $\mathfrak{L}$;
停止条件控制参数 T.

过程：

$E=\infty$;
$d=|A|$;
$A^*=A$;
$t=0$;
while $t<T$ **do**
  随机产生特征子集 A';
  $d'=|A'|$;
  $E'=\text{CrossValidation}(\mathfrak{L}(D^{A'}))$;
  if $(E'<E)\vee((E'=E)\wedge(d'<d))$ **then**
    $t=0$;
    $E=E'$;
    $d=d'$;
    $A^*=A'$
  else
    $t=t+1$
  end if
end while

输出： 特征子集 A^*

直接把最终将要使用的学习器的性能作为特征子集的评价准则。
LVM 是在拉斯维加斯方法框架下使用随即策略来进行子集搜索，并以最终分类器的误差为特征子集评价标准。
算法第 8 行是通过在数据集 D 上，使用交叉验证法来估计学习器的误差，注意这个误差是在仅考虑特征子集 A'时得到的，即特征子集 A'上的误差，若它比当前特征子集 A 上的误差更小，或误差项当时但 A'中包含的特征数更少，则将 A'保留下来。
需注意计算开销和时间开销均极大。

11.4 嵌入式选择与 L_1 正则化

给定数据集 $D=\{(\boldsymbol{x}_1,y_1),\cdots,(\boldsymbol{x}_m,y_m)\}$，考虑最简单的线性回归模型，以平方误差为损失函数，则优化目标为 $\min_{\boldsymbol{w}}\sum_{i=1}^{m}(y_i-\boldsymbol{w}^T\boldsymbol{x}_i)^2$。当样本特征很多，而样本数相对较少时，容易陷入过拟合。可以引入正则化项改为 $\min_{\boldsymbol{w}}\sum_{i=1}^{m}(y_i-\boldsymbol{w}^T\boldsymbol{x}_i)^2+\lambda\|\boldsymbol{w}\|_2^2$，其中正则化参数$\lambda>0$，称为“岭回归”。也可换为 L_p 范数；若 p=1，称为 LASSO 回归。L_1 范数比 L_2 范数更易于获得“稀疏”解，即求得的 $\boldsymbol{w}$ 会有更少的非零分量。
注意到 $\boldsymbol{w}$ 取得稀疏解意味着初始的 d 个特征仅有对应着 $\boldsymbol{w}$ 的非零分量的特征才会出现在最终模型中。于是，求解 L_1 范数正则化的结果是得到了仅采用一部分初始特征的模型；换言之，基于 L_1 正则化的学习方法就是一种嵌入式特征选择方法，其特征选择过程与学习器训练过程融为一体，同时完成。
L_1 正则化问题可使用近端梯度下降 PGD。
优化目标：$\min_{\boldsymbol{x}} f(\boldsymbol{x})+\lambda\|\boldsymbol{x}\|_1$，若$f(\boldsymbol{x})$可导，且$\nabla f$满足 L-Lipschitz 条件，即是$\|\nabla f(\boldsymbol{x}')-\nabla f(\boldsymbol{x})\|_2^2\leqslant L\|\boldsymbol{x}'-\boldsymbol{x}\|_2^2$，则在 $\boldsymbol{x}_k$ 附近可将$f(\boldsymbol{x})$通过二阶泰勒展式近似为 $f(\boldsymbol{x})\approx f(\boldsymbol{x}_k)+(\nabla f(\boldsymbol{x}_k),\boldsymbol{x}-\boldsymbol{x}_k)+L/2\cdot\|\boldsymbol{x}-\boldsymbol{x}_k\|_2^2=\frac{L}{2}\left\|\boldsymbol{x}-\left(\boldsymbol{x}_k-\frac{1}{L}\nabla f(\boldsymbol{x}_k)\right)\right\|_2^2+const.$。则该式的最小值在 $\boldsymbol{x}_{k+1}=\boldsymbol{x}_k-\frac{1}{L}\nabla f(\boldsymbol{x}_k):=\boldsymbol{z}$。推广到优化目标，类似得到每一步迭代应为：$\boldsymbol{x}_{k+1}=\arg\min_{\boldsymbol{x}}\frac{L}{2}\left\|\boldsymbol{x}-\left(\boldsymbol{x}_k-\frac{1}{L}\nabla f(\boldsymbol{x}_k)\right)\right\|_2^2+\lambda\|\boldsymbol{x}\|_1$。展开发现没有形如 x^ix^j 的项，即分量之间互不影响，于是有闭式解：$x_{k+1}^i=\begin{cases}z^i-\lambda/L, & \lambda/L<z^i\\ 0, & |z^i|\leq\lambda/L\\ z^i+\lambda/L, & z^i<-\lambda/L\end{cases}$。

11.5 稀疏表示与字典学习

把数据集 D 考虑成一个矩阵，每行对应于一个样本，每列对应于一个特征。特征选择所考虑的问题是特征具有“稀疏性”，即矩阵中的许多列与当前学习任务无关，通过特征选择去除这些列，则学习器仅需在较小的矩阵上进行。
考虑另一种稀疏性：D 所对应的矩阵中存在很多零元素，但这些零元素并不是以

整列、整行形式存在的。当样本具有这样的稀疏表达形式时，对学习任务来说会有不少好处，比如使问题的可分性提高，稀疏矩阵存储空间也很低。
如新华字典，大部分字并不会出现在样本文档里，即得稀疏矩阵。
若给定数据集 D 是稠密的，能否将其转化为稀疏表示形式呢？
对于一般的学习任务，可以学习出字典。为普通稠密表达的的样本找到合适的字典，将样本转化为合适的稀疏表达形式，从而使学习任务得以简化，模型复杂度得以降低，通常称为“字典学习”。

给定数据集$\{\boldsymbol{x}_1,\cdots,\boldsymbol{x}_m\}$，字典学习最简单的形式为$\min\limits_{B,\boldsymbol{\alpha}_i}\sum\limits_{i=1}^{m}\|\boldsymbol{x}_i-B\boldsymbol{\alpha}_i\|_2^2+\lambda\sum\limits_{i=1}^{m}\|\boldsymbol{\alpha}_i\|_1$，

其中$B\in R^{d\times k}$称为字典矩阵，k称为字典的词汇量，通常由用户指定，$\boldsymbol{\alpha}_i\in R^k$则是样本$\boldsymbol{x}_i\in R^d$的稀疏表示。优化目标的第一项是希望由$\boldsymbol{\alpha}_i$能很好地重构$\boldsymbol{x}_i$，第二项则是希望$\boldsymbol{\alpha}_i$尽量稀疏。
与 LASSO 相比，此优化目标显然麻烦的多，因为除了类似于 LASSO 回归中的$\boldsymbol{w}$的$\boldsymbol{\alpha}_i$，还需要学习字典矩阵 B。不过，受 LASSO 的启发，可以采用变量交替优化的策略来求解。
首先在第一步，我们固定字典 B，若将优化目标按分量展开，发现没有交叉项，于是可参照 LASSO 的解法求解下式，从而为每个样本$\boldsymbol{x}_i$找到相应的$\boldsymbol{\alpha}_i$：

$\min\limits_{\boldsymbol{\alpha}_i}\|\boldsymbol{x}_i-B\boldsymbol{\alpha}_i\|_2^2+\lambda\|\boldsymbol{\alpha}_i\|_1$。

在第二步，我们以$\boldsymbol{\alpha}_i$为初值来更新字典 B，此时改写为$\min\limits_{B}\|X-BA\|_F^2$。此式有多种求解方法，常见有基于逐列更新策略的 KSVD。令$\boldsymbol{b}_i$表示字典矩阵 B 的第i列，$\boldsymbol{\alpha}^i$表示稀疏矩阵 A 的第i行，则可重写为$\min\limits_{B}\|X-BA\|_F^2=\min\limits_{\boldsymbol{b}_i}\left\|X-\sum\limits_{j=1}^{k}\boldsymbol{b}_j\boldsymbol{\alpha}^j\right\|_F^2$

$=\min\limits_{\boldsymbol{b}_i}\left\|\left(X-\sum\limits_{j\neq i}\boldsymbol{b}_j\boldsymbol{\alpha}^j\right)-\boldsymbol{b}_i\boldsymbol{\alpha}^i\right\|_F^2=\min\limits_{\boldsymbol{b}_i}\|E_i-\boldsymbol{b}_i\boldsymbol{\alpha}^i\|_F^2$。在更新字典的第$i$列时，其他各列都是固定的，因此$E_i$是固定的，于是原则上只需对$E_i$进行奇异值分解以取得最大奇异值所对应的正交向量。直接对E_i进行奇异值分解会同时修改$\boldsymbol{b}_i$和$\boldsymbol{\alpha}^j$，从而可能破坏 A 的稀疏性。为避免发生这种情况，KSVD 对E_i和$\boldsymbol{\alpha}^i$进行专门化处理：$\boldsymbol{\alpha}^i$仅保留非零元素，E_i则仅保留$\boldsymbol{b}_i$和$\boldsymbol{\alpha}^i$的非零元素的乘积项，然后再进行奇异值分解，这样就保持了第一步所得到的稀疏性。
反复迭代上述两步，即可得到字典 B 和样本$\boldsymbol{x}_i$的稀疏表示$\boldsymbol{\alpha}_i$。

11.6 压缩感知

希望根据部分信息来恢复全部信息。
假定有长度为m的离散信号$\boldsymbol{x}$，得到长度为n的采样后信号$\boldsymbol{y}$，$n<<m$，即$\boldsymbol{y}=\Phi\boldsymbol{x}$，其中$\Phi\in R^{n\times m}$是对信号$\boldsymbol{x}$的测量矩阵，它确定了以什么频率进行采样以及如何将采样样本组成采样后的信号。
不妨假设存在某个线性变换$\psi\in R^{m\times m}$，使得$\boldsymbol{x}=\psi\boldsymbol{s}$，于是$\boldsymbol{y}=\Phi\psi\boldsymbol{s}=A\boldsymbol{s}$，我们希望根据$\boldsymbol{y}$恢复出信号$\boldsymbol{s}$。若$\boldsymbol{s}$具有稀疏性，这个问题能得到很好的解决。因为稀疏性使得未知因素的影响大为减少。此时ψ称为稀疏基，A 的作用类似于字典。
通常认为，压缩感知分为感知测量核重构恢复两个阶段。感知测量关注如何对原

始信号进行处理以获得系数样本表示，涉及傅里叶变换、小波变换等知识。重构恢复关注如何基于稀疏性从少量观测中恢复原信号。

12 计算学习理论

12.1 基础知识

通过计算来进行学习的理论，即关于机器学习的理论基础，目的是分析学习任务的困难本质，为学习算法提供理论保证，并根据分析结果指导算法设计。

给定样例集 $D=\{(\boldsymbol{x}_1,y_1),\cdots,(\boldsymbol{x}_m,y_m)\}$，本章主要讨论二分类问题。假设 X 中的所有样本服从一个隐含未知的分布 D，D 中所有样本都是独立同分布。

令 h 为 X 到 Y 的一个映射，泛化误差为 $E(h;D)=P_{x\sim D}(h(\boldsymbol{x})\neq y)$，$h$ 在 D 上的经验误差为 $\hat{E}(h;D)=\frac{1}{m}\sum_{i=1}^{m}\chi(h(\boldsymbol{x}_i)\neq y)$。由于独立同分布采样，因此 h 的经验误差期望等于其泛化误差。令ε为 $E(h)$的上限，也称为误差参数。

若 h 在数据集 D 上的经验误差为 0，则称 h 与 D 一致，否则称为不一致。对于两个映射 h_1,h_2，可以通过不合来度量它们的差别：$d(h_1,h_2)=P_{x\sim D}(h_1(\boldsymbol{x})\neq h_2(\boldsymbol{x}))$。

Jensen 不等式：对任意凸函数 $f(x)$，有 $f(E(x))\leqslant E(f(x))$。

Hoeffding 不等式：若 $x_1,\cdots,x_m$ 为 m 个独立随机变量，且满足 $0\leqslant x_i\leqslant 1$，则对任意ε>0，$P(\frac{1}{m}\sum_{i=1}^{m}x_i-\frac{1}{m}\sum_{i=1}^{m}E(x_i)\geq\varepsilon)\leq e^{-2m\varepsilon^2}$，$P(|\frac{1}{m}\sum_{i=1}^{m}x_i-\frac{1}{m}\sum_{i=1}^{m}E(x_i)|\geq\varepsilon)\leq 2e^{-2m\varepsilon^2}$。

McDiarmid 不等式：若 $x_1,\cdots,x_m$ 为 m 个独立随机变量，且对任意 $1\leqslant i\leqslant m$，函数 f 满足 $\sup_{x_1,\cdots,x_m,x_i'}|f(x_1,\cdots,x_m)-f(x_1,\cdots,x_{i-1},x_i',x_{i+1},\cdots,x_m)|\leqslant c_i$，则对任意ε>0，

$$P(f(x_1,\cdots,x_m)-E(f(x_1,\cdots,x_m)\geq\varepsilon)\leq e^{\frac{-2\varepsilon^2}{\Sigma_i c_i^2}},$$

$$P(|f(x_1,\cdots,x_m)-E(f(x_1,\cdots,x_m)|\geq\varepsilon)\leq 2e^{\frac{-2\varepsilon^2}{\Sigma_i c_i^2}}。$$

12.2 PAC 学习

概率近似正确学习理论。令 c 表示概念，这是从样本空间 X 到标记空间 Y 的映射，它决定示例 $\boldsymbol{x}$ 的真实标记 y。若对任何样例$(\boldsymbol{x},y)$有 $c(\boldsymbol{x})=y$ 成立，则称 c 为目标概念。所有我们希望学得的目标概念所构成的集合称为“概念类”，记为 C。

给定学习算法Σ，它所考虑的所有可能概念的集合称为“假设空间”，记为 H。

由于学习算法事先不知道概念类的真实存在，因此 H 和 C 通常不同。对 $h\in H$，由于并不知道它是否真是目标概念，因此称为“假设”。

若目标概念 $c\in H$，则 H 中存在假设能将所有示例与真实标记一致的方式完全分开，我们称该问题对学习算法Σ是可分的。反之，称为不可分。

给定训练集 D，我们希望基于学习算法学得的模型所对应的假设 h 尽可能接近目标概念 c。也就是说，以较大的概率学得误差满足预设上限的模型。

定义 PAC 辨识：对 $0<\varepsilon,\delta<1$，所有 $c\in C$ 和分布 D，若存在学习算法Σ，其输出假设 $h\in H$ 满足 $P(E(h)\leqslant\varepsilon)\geqslant 1-\delta$，则称学习算法Σ能从假设空间 H 中 PAC 辨识概念类 C。这样的学习算法能以较大概率$(1-\delta)$学得目标概念 c 的近似(误差≤ε)。

定义 PAC 可学习：令 m 表示从分布 D 中独立通分布采样得到的样例数目，$0<\varepsilon,\delta<1$，对所有分布 D，若存在学习算法Σ和多项式函数 $poly(\cdot,\cdot,\cdot,\cdot)$，使得对于任

意 m≥poly(1/ε,1/δ,size(***x***),size(*c*))，Σ能从假设空间 H 中 PAC 辨识概念类 C，则称概念类 C 对假设空间 H 而言是 PAC 可学习的。

定义 PAC 学习算法：若学习算法Σ使概念类 C 为 PAC 可学习的，且Σ的运行时间也是多项式函数 poly(1/ε,1/δ,size(***x***),size(*c*))，则称概念类 C 是高效 PAC 学习的，称Σ为概念类 C 的 PAC 学习算法。

定义样本复杂度：满足 PAC 学习算法Σ所需的 $m \geq \text{poly}(1/\varepsilon,1/\delta,\text{size}(\boldsymbol{x}),\text{size}(c))$中最小的 *m*，称为学习算法Σ的样本复杂度。

12.3 有限假设空间

• 可分情形

假设空间 H 中可能存在不止一个与 D 一致的等效假设，无法根据 D 来对它们的优劣做进一步区分。到底需要多少样例才能学得有效近似呢？

先估计泛化误差大于ε但在训练集上表现完美的假设出现的概率。对分布 D 上随机采样得到的样例(***x***,*y*)，成立 $P(h(\boldsymbol{x})=y)=1-P(h(\boldsymbol{x})\neq y)=1-E(h)<1-\varepsilon$。由于 D 包含 *m* 个从 D 独立同分布采样而得的样例，*h* 与 D 保持一致的概率$<(1-\varepsilon)^m$。事先不知道学习算法会输出 H 中的哪个假设，但仅需保证泛化误差大于ε，且在训练集上表现完美的所有假设出现概率之和不大于δ:$P(h\in H:E(h)>\varepsilon)<|H|e^{-m\varepsilon}$。从而 $m \geq \frac{1}{\varepsilon}(\ln|H|+\ln\frac{1}{\delta})$。由此可知，有限假设空间 H 都是可学习的，所需的样例数目如上式所示。

• 不可分情形

目标概念不存在于假设空间 H 中，任何一个假设都会在训练集上出现或多或少的错误。

由 Hoeffding 不等式： $P(|E(h)-\hat{E}(h)|\geq\varepsilon)\leq e^{-2m\varepsilon^2}$ 知：

若训练集 D 包含 *m* 个从分布 D 上独立同分布采样而得到的样例，0<ε<1，则对任意 $h\in H$，下式以至少以 1−δ的概率成立：$|E(h)-\hat{E}(h)|\leq\sqrt{\frac{\ln(2/\delta)}{2m}}$。这表明，样例数目 *m* 较大时，*h* 的经验误差是其泛化误差很好的近似。对于有限假设空间 H，我们有： $P(|E(h)-\hat{E}(h)|\leq\sqrt{\frac{\ln(2/\delta)+\ln|H|}{2m}})\geq 1-\delta$。

当 $c\notin H$ 时，学习算法无法学得目标概念 *c* 的ε近似。但是，当假设空间 H 给定时，其中必存在一个泛化误差最小的假设，找出此假设的ε近似也不失为一个较好的目标。设 H 中泛化误差最小的假设是 *h*，于是以此为目标可将 PAC 学习推广到 $c\notin H$ 的情况，则称为不可知学习。

定义不可知 PAC 学习：*m* 表示从分布 D 中独立同分布采样得到的样例数目，0<ε,δ<1，对所有分布 D，若存在学习算法Σ和多项式函数 poly(·,·,·,·)，使得对于任何的 $m\geq\text{poly}(1/\varepsilon,1/\delta,\text{size}(\boldsymbol{x}),\text{size}(c))$，Σ能从假设空间 H 中输出满足下式的假设 *h*：$P(E(h)-\min_{h'\in H}E(h')\leq\varepsilon)\geq 1-\delta$，则称假设空间 H 是不可知 PAC 可学习的。不可知 PAC 学习算法、样本复杂度类似定义。

12.4 VC 维

现实学习任务通常面临无线假设空间，需度量假设空间的复杂度。最常见的办法是考虑假设空间的“VC”维。

给定假设空间 H 和示例集 D={$\boldsymbol{x}_1,\cdots,\boldsymbol{x}_m$}，H 中每个假设 h 都能对 D 中示例赋予标记，结果为 $h|_D=\{(h(\boldsymbol{x}_1),\cdots,h(\boldsymbol{x}_m))\}$。随着 m 增大，标记可能结果数也会增大。定义空间 H 的增长函数$\Pi_H(m)=\max|\{(h(\boldsymbol{x}_1),\cdots,h(\boldsymbol{x}_m))|h\in H\}|$。这表示了假设空间 H 对 m 个示例所能赋予标记的最大可能结果数，描述了假设空间 H 的表示能力，由此反映出假设空间的复杂度。
对假设空间 H，m∈N，0＜ε＜1，任意 h∈H，成立

$$P(|E(h)-\hat{E}(h)|>\varepsilon)\geq 4\Pi_H(2m)\exp(-\frac{m\varepsilon^2}{8})。$$

对二分类问题来说，H 中所有假设对 D 中示例赋予标记的每种可能结果称为对 D 的一种对分。若假设空间 H 能实现所有对分，则称 D 能被假设空间 H 打散。
定义 VC 维：假设空间 H 的 VC 维是能被 H 大散的最大示例集的大小，即 VC(H)=max{m:$\Pi_H(m)=2^m$}。VC(H)=d 表示存在大小为 d 的示例集能被假设空间 H 打散。需要注意，定义 VC 维时与数据分布 D 无关，因此在数据分布未知时仍能计算出假设空间 H 的 VC 维。

Sauer 引理：若假设空间 H 的 VC 维为 d，则对任意 m∈N，有 $\Pi_H(m)\leq\sum_{i=0}^{d}C_m^i$。

推论：若假设空间 H 的 VC 维为 d，则对任意整数 m≥d，有 $\Pi_H(m)\leq\left(\frac{e\cdot m}{d}\right)^d$。

得到基于 **VC 维的泛化误差界**：若假设空间 H 的 VC 维为 d，则对任意 m＞d，0＜δ＜1 和 h∈H，有 $P\left(|E(h)-\hat{E}(h)|\leq\sqrt{\frac{8d\ln\frac{2em}{d}+8\ln\frac{4}{\delta}}{m}}\right)\geq 1-\delta$。可见此式只与样例数目有关，而与数据分布和样例集无关。

令 h 表示学习算法Σ输出的假设，若 h 满足 $\hat{E}(h)=\min\hat{E}(h'),h'\in H$，则称Σ维满足经验风险最小化(ERM)原则的算法。
定理：任何 VC 维有限的假设空间 H 都是（不可知）PAC 可学习的。
VC 维得到的泛化误差界往往比较松。

12.5 Rademacher 复杂度

给定训练集 D={($\boldsymbol{x}_1,y_1$),⋯,($\boldsymbol{x}_m,y_m$)}，假设 h 的经验误差为 $\hat{E}(h)=\frac{1}{m}\sum_{i=1}^{m}\chi(h(\boldsymbol{x}_i)\neq y_i)$

$=\frac{1}{2}-\frac{1}{2m}\sum_{i=1}^{m}y_ih(\boldsymbol{x}_i)$，经验误差最小的假设是 $\arg\max_{h\in H}\frac{1}{m}\sum_{i=1}^{m}y_ih(\boldsymbol{x}_i)$。现实任务样例的标记有时会受到噪声影响，即使 y_i 已经不再是 $\boldsymbol{x}_i$ 的真实标记。在此情形下，选择在训练集上表现最好的假设，有时还不如选择 H 中事先已考虑了随机噪声影响的假设。
考虑随机变量σ_i，以 0.5 概率取−1，0.5 概率取 1，称为 Rademacher 随机变量。

将上式重写为$\sup_{h\in H}\frac{1}{m}\sum_{i=1}^{m}\sigma_i h(\boldsymbol{x}_i)$，并取期望得到$\mathrm{E}_{\sigma}[\sup_{h\in H}\frac{1}{m}\sum_{i=1}^{m}\sigma_i h(\boldsymbol{x}_i)]\in[0,1]$，它体现了假设空间 H 的表达能力。例如，|H|=1，则期望为 0；若 H 能打散 D，则期望为 1。

考虑实值函数空间 F:X→R，令 Y={$\boldsymbol{x}_1,\cdots,\boldsymbol{x}_m$}，其中 $\boldsymbol{x}_i\in$X。定义函数空间 F 关于 Y 的经验 Rademacher 复杂度：$\hat{R}_Y(F)=\mathrm{E}_{\sigma}[\sup_{f\in F}\frac{1}{m}\sum_{i=1}^{m}\sigma_i f(\boldsymbol{x}_i)]$。其衡量了函数空间 F 与随机噪声在集合 Y 中的相关性。通常我们希望了解函数空间 F 在 X 上关于分布 D 的相关性，因此可对所有从 D 独立同分布采样而得的大小为 m 的集合 Y 求期望，得到$R_m(F)=\mathrm{E}_{Y\subset X:|Y|=m}[\hat{R}_Y(F)]$。

可得到关于函数空间基于 **Rademacher 复杂度的泛化误差界**：

定理：对实值函数空间 F：Z→[0,1]，根据分布 D 从 Z 中独立同分布采样得到示例集 Y={$\boldsymbol{x}_1,\cdots,\boldsymbol{x}_m$}，$\boldsymbol{x}_i\in$Z，0＜δ＜1，对任意 $f\in$F，至少 1−δ概率有：

$$\mathrm{E}[f(\boldsymbol{x})]\le\frac{1}{m}\sum_{i=1}^{m}f(\boldsymbol{x}_i)+2R_m(F)+\sqrt{\frac{\ln(1/\delta)}{2m}};$$

$$\mathrm{E}[f(\boldsymbol{x})]\le\frac{1}{m}\sum_{i=1}^{m}f(\boldsymbol{x}_i)+2R_Y(F)+3\sqrt{\frac{\ln(2/\delta)}{2m}}。$$

定理：对假设空间 H:X→{±1}，根据分布 D 从 X 中独立同分布采样得到示例集 Y={$\boldsymbol{x}_1,\cdots,\boldsymbol{x}_m$}，$\boldsymbol{x}_i\in$X，0＜δ＜1，对任意 $h\in$H，以至少 1−δ概率有：

$$E(h)\le\hat{E}(h)+R_m(H)+\sqrt{\frac{\ln(1/\delta)}{2m}};\quad E(h)\le\hat{E}(h)+\hat{R}_Y(H)+3\sqrt{\frac{\ln(2/\delta)}{2m}}。$$

可以看出 Rademacher 复杂度的泛化误差界与分布 D 和数据集 Y 有关，通常比 VC 维的泛化误差更紧。

假设空间 H 的复杂度 $R_m(H)$与增长函数$\Pi_H(m)$满足：$R_m(H)\le\sqrt{\frac{2\ln\Pi_H(m)}{m}}$，

从而可以推出 VC 维的泛化误差界：$E(h)\le\hat{E}(h)+\sqrt{\frac{2d\ln(em/d)}{m}}+\sqrt{\frac{\ln(1/\delta)}{2m}}$。

12.6 稳定性

希望获得与算法有关的分析结果。给定 D=($\boldsymbol{z}_1$=($\boldsymbol{x}_1$,y_1),⋯,$\boldsymbol{z}_m$=($\boldsymbol{x}_m$,y_m))，$\boldsymbol{x}_i\in$X 是来自分布 D 的独立同分布示例，$y_i\in\{\pm1\}$。对假设空间 H:X→{±1}和学习算法Σ，令$\Sigma_D\in$H 表示基于训练集 D 从假设空间 H 中学得的假设，考虑 D 的以下变化：

- D\i 表示移除第 i 个样例得到的集合；
- D^i 表示替换第 i 个样例得到的集合。

损失函数 $l(\Sigma_D(\boldsymbol{x}),y)$刻画了假设$\Sigma_D$的预测标记$\Sigma_D(\boldsymbol{x})$与真实标记 y 之间的差别，简记为 $l(\Sigma_D,\boldsymbol{z})$。

- 泛化损失：$l(\Sigma,D)=\mathrm{E}_{x\in X,z=(x,y)}[l(\Sigma_D,\boldsymbol{z})]$；

• 经验损失：$\hat{l}(\Sigma,\mathrm{D})=\frac{1}{m}\sum_{i=1}^{m}l(\Sigma_D,\boldsymbol{z}_i)$；

• 留一损失：$l_{loo}(\Sigma,\mathrm{D})=\frac{1}{m}\sum_{i=1}^{m}l(\Sigma_{D\backslash i},\boldsymbol{z}_i)$。

定义算法均匀稳定性：对任意 $\boldsymbol{x}\in\mathrm{X}$，$\boldsymbol{z}=(\boldsymbol{x},y)$，若学习算法Σ满足：$|l(\Sigma_{\mathrm{D}},\boldsymbol{z})-l(\Sigma_{\mathrm{D}\backslash i},\boldsymbol{z})|\leqslant\beta$，$i=1,2,\cdots,m$，则称Σ关于损失函数 l 满足β-均匀稳定性。
注：移出示例的稳定性包含替换示例的稳定性。
若损失函数 l 有界，即对所有的 D 和 $\boldsymbol{z}=(\boldsymbol{x},y)$有 $0\leqslant l(\Sigma_{\mathrm{D}},\boldsymbol{z})\leqslant\mathrm{M}$，则有：
给定从分布 D 上独立同分布采样得到的大小为 m 的示例集 D，若学习算法Σ满足关于损失函数 l 的β-均匀稳定性，且损失函数 l 的上界为 M，$0<\delta<1$，则对任意 $m\geqslant 1$，以至少 $1-\delta$概率有：

$$l(\Sigma,D)\leq\hat{l}(\Sigma,D)+2\beta+(4m\beta+M)\sqrt{\frac{\ln(1/\delta)}{2m}},$$

$$l(\Sigma,D)\leq l_{loo}(\Sigma,D)+\beta+(4m\beta+M)\sqrt{\frac{\ln(1/\delta)}{2m}}。$$

这给出了基于稳定性分析推导出的学习算法Σ学得假设的泛化误差界。首先，必须假设 $\beta\sqrt{m}\to 0$，即经验损失收敛于泛化损失。便于计算设 $\beta=1/m$，代入得到

$$l(\Sigma,D)\leq\hat{l}(\Sigma,D)+\frac{2}{m}+(4+M)\sqrt{\frac{\ln(1/\delta)}{2m}}。$$

对损失函数 l，若学习算法Σ所输出的假设满足经验损失最小化，则称算法满足 ERM 原则。
定理：若学习算法Σ是 ERM 且稳定的，则假设空间 H 可学习。

13 半监督学习

13.1 未标记样本

学习器不依赖外界交互、自动地利用未标记样本来提升学习能力，称为半监督学习。要利用未标记样本，需要做一些假设。聚类假设：假设数据存在簇结构，同一个簇的样本属于同一个类别。流形假设：假设数据分布在一个流形结构上，邻近样本拥有相似的输出值。
可进一步分为纯半监督学习（假定训练数据中的未标记样本并非待遇测的数据）和直推学习（假定学习过程中所考虑的未标记样本恰是待预测数据，学习目的是在这些未标记样本上获得最优泛化性能）。

13.2 生成式方法

假设所有数据都是由同一个潜在的模型“生成”的，未标记数据可看作模型的缺失参数，可基于 EM 算法进行极大似然估计求解。
给定样本 $\boldsymbol{x}$，真实类别标记为 y。假设样本由高斯混合模型生成，且每个类别对应一个高斯混合成分，即 $p(\boldsymbol{x})=\sum_{i=1}^{N}\alpha_i p(\boldsymbol{x}\mid\boldsymbol{\mu}_i,\Sigma_i)$。设 $f(\boldsymbol{x})$表示模型 f 对 $\boldsymbol{x}$ 的预测

标记，Θ表示样本 $\boldsymbol{x}$ 隶属的高斯混合成份。由后验概率最大化，知

$$f(\boldsymbol{x})=\arg\max_{y\in Y}\sum_{i=1}^{n}p(y=j,\Theta=i\mid\boldsymbol{x})=\arg\max_{y\in Y}\sum_{i=1}^{n}p(y=j\mid\Theta=i,\boldsymbol{x})p(\Theta=i\mid\boldsymbol{x}),$$

其中 $p(\Theta=i\mid\boldsymbol{x})=\dfrac{\alpha_i\cdot p(\boldsymbol{x}\mid\mu_i,\Sigma_i)}{\sum_{i=1}^{N}\alpha_i\cdot p(\boldsymbol{x}\mid\mu_i,\Sigma_i)}$。

给定有标记样本集 $D_l=\{(\boldsymbol{x}_1,y_1),\cdots,(\boldsymbol{x}_l,y_l)\}$和未标记样本集 $D_u=\{\boldsymbol{x}_{l+1},\cdots,\boldsymbol{x}_{l+u}\}$。假设所有样本独立同分布，且由同一个高斯混合模型生成，则 $D_l\cup D_u$ 的对数似然是

$$LL(D_l\bigcup D_u)=\sum_{(\boldsymbol{x}_i,y_i)\in D_l}\ln\left(\sum_{i=1}^{N}\alpha_i p(\boldsymbol{x}_j\mid\boldsymbol{\mu}_i,\Sigma_i)p(y_j\mid\Theta=i,\boldsymbol{x}_j)\right)+\sum_{\boldsymbol{x}_j\in D_u}\ln\left(\sum_{i=1}^{N}\alpha_i p(\boldsymbol{x}_j\mid\boldsymbol{\mu}_i,\Sigma_i)\right)$$

可用 EM 算法求解，更新式为：

• E 步：$\gamma_{ji}=\dfrac{\alpha_i p(\boldsymbol{x}_j\mid\boldsymbol{\mu}_i,\Sigma_i)}{\sum_{i=1}^{N}\alpha_i p(\boldsymbol{x}_j\mid\boldsymbol{\mu}_i,\Sigma_i)}$；

• M 步：基于γ_{ji}更新模型参数，其中 l_i 表示第 i 类的有标记样本的数目，

$$\boldsymbol{\mu}_i=\frac{1}{\sum_{\boldsymbol{x}_j\in D_u}\gamma_{ji}+l_i}\left(\sum_{\boldsymbol{x}_j\in D_u}\gamma_{ji}\boldsymbol{x}_j+\sum_{(\boldsymbol{x}_j,y_j)\in D_l\cap y_j=i}\boldsymbol{x}_j\right),$$

$$\Sigma_i=\frac{1}{\sum_{\boldsymbol{x}_j\in D_u}\gamma_{ji}+l_i}\left(\sum_{\boldsymbol{x}_j\in D_u}\gamma_{ji}(\boldsymbol{x}_j-\boldsymbol{\mu}_i)(\boldsymbol{x}_j-\boldsymbol{\mu}_i)^T+\sum_{(\boldsymbol{x}_j,y_j)\in D_l\cap y_j=i}(\boldsymbol{x}_j-\boldsymbol{\mu}_i)(\boldsymbol{x}_j-\boldsymbol{\mu}_i)^T\right),$$

$$\alpha_i=\frac{1}{l+u}\left(\sum_{\boldsymbol{x}_j\in D_u}\gamma_{ji}+l_i\right)。$$

此类方法关键：模型假设必须准确，生成式模型必须与真实数据分布吻合，否则利用未标记数据反倒会降低泛化性能。

13.3 半监督 SVM(S3VM)

试图找到能将两类有标记样本分开，且穿过数据低密度区域的划分超平面。最著名的是 TSVM（Transductive Support Vector Machine）。它试图考虑对未标记样本进行各种可能的标记指派，即尝试将每个未标记样本分别作为正例或反例，然后在所有这些结果中寻求一个在所有样本上间隔最大化的划分超平面。

给定 $D_l=\{(\boldsymbol{x}_1,y_1),\cdots,(\boldsymbol{x}_l,y_l)\}$和 $D_u=\{\boldsymbol{x}_{l+1},\cdots,\boldsymbol{x}_{l+u}\}$，其中 $y_i\in\{\pm1\}$，TSVM 的学习目标是为 D_u 中的样本给出预测标记 $\hat{\boldsymbol{y}}=(y_{l+1},\cdots,y_{l+u})$，使得

$$\min_{\boldsymbol{\omega},b,\hat{\boldsymbol{y}},\boldsymbol{\xi}}\frac{1}{2}\|\boldsymbol{\omega}\|_2^2+C_l\sum_{i=1}^{l}\xi_i+C_u\sum_{i=l+1}^{l+u}\xi_i\ \ s.t.\begin{cases}y_i(\boldsymbol{\omega}^T\boldsymbol{x}_i+b)\geq1-\xi_i,i=1,\cdots,l\\ \hat{y}_i(\boldsymbol{\omega}^T\boldsymbol{x}_i+b)\geq1-\xi_i,i=l+1,\cdots,l+u\end{cases}\quad with\ \ \xi_i\geq0$$

若尝试未标记样本的各种标记指派是一个穷举过程，仅当未标记样本很少时才有

可能直接求解。必须采用更高效的优化策略。

TSVM 采用局部搜索法来迭代地寻找近似解。先利用有标记样本学得一个 SVM，然后利用这个 SVM 对未标记数据进行标记指派（赋予“伪标记”）。此时 $\hat{\boldsymbol{y}}$ 已知，得到标准 SVM 问题，再求解出新的划分超平面和松弛变量（注意 C_u 应当设置比 C_l 小，让有标记样本所起作用更大）。接下来，TSVM 找出两个标记指派为异类且很有可能发生错误的未标记样本，交换它们的标记，再重新基于优化问题求解出划分超平面和松弛变量；再找出两个标记指派为异类且很有可能发生错误的未标记样本……标记指派调整完成后，逐步增大 C_u 以提高未标记样本对优化目标的影响，进行下一轮标记指派调整，直至 C_u=C_l。此时求解得到的 SVM 给未标记样本提供了标记，还能对未见的示例进行预测。

对未标记样本进行样本指派及调整的过程中，有可能出现类别不平衡问题，这是可以将优化目标中的 C_u 项拆分成 C_u^+和 C_u^-项，分别对应基于伪标记而当作正、反例使用的未标记样本，并在初始化时满足 $C_u^+u_+$=$C_u^-u_-$。

输入： 有标记样本集 $D_l=\{(\boldsymbol{x}_1,y_1),(\boldsymbol{x}_2,y_2),\ldots,(\boldsymbol{x}_l,y_l)\}$;
未标记样本集 $D_u=\{\boldsymbol{x}_{l+1},\boldsymbol{x}_{l+2},\ldots,\boldsymbol{x}_{l+u}\}$;
折中参数 C_l, C_u.

过程：

用 D_l 训练一个 SVM_l;
用 SVM_l 对 D_u 中样本进行预测, 得到 $\hat{\boldsymbol{y}}=(\hat{y}_{l+1},\hat{y}_{l+2},\ldots,\hat{y}_{l+u})$;
初始化 $C_u \ll C_l$;
while $C_u < C_l$ **do**
  基于 $D_l, D_u, \hat{\boldsymbol{y}}, C_l, C_u$ 求解式(13.9), 得到 $(\boldsymbol{w},b)$, $\boldsymbol{\xi}$;
  while $\exists\{i,j \mid (\hat{y}_i\hat{y}_j<0)\wedge(\xi_i>0)\wedge(\xi_j>0)\wedge(\xi_i+\xi_j>2)\}$ **do**
    $\hat{y}_i=-\hat{y}_i$;
    $\hat{y}_j=-\hat{y}_j$;
    基于 $D_l, D_u, \hat{\boldsymbol{y}}, C_l, C_u$ 重新求解式(13.9), 得到 $(\boldsymbol{w},b)$, $\boldsymbol{\xi}$
  end while
  $C_u=\min\{2C_u, C_l\}$
end while

输出： 未标记样本的预测结果: $\hat{\boldsymbol{y}}=(\hat{y}_{l+1},\hat{y}_{l+2},\ldots,\hat{y}_{l+u})$

13.4 图半监督学习

给定一个数据集，可将其映射为一个图，每个样本对应于图中一个结点。若两个样本间相似度很高，则对应的结点之间存在一条边，边的强度正比于样本之间的相似度。可将有标记样本所对应的结点想象为染过色，而未标记样本所对应的结点尚未染色。半监督学习就对应于“颜色”在图上扩散或传播过程。由于一个图对应于一个矩阵，这能使得我们能基于矩阵运算来进行半监督学习算法的推导。给定 D_l=$\{(\boldsymbol{x}_1,y_1),\cdots,(\boldsymbol{x}_l,y_l)\}$和 D_u=$\{\boldsymbol{x}_{l+1},\cdots,\boldsymbol{x}_{l+u}\}$。先基于 $D_l\cup D_u$构建一个图 G=(V,E)，其中边集 E 表示为一个亲和矩阵，常定义 $(\mathrm{W})_{ij}=\begin{cases}\exp\left(-\dfrac{\|\boldsymbol{x}_i-\boldsymbol{x}_j\|_2^2}{2\sigma^2}\right), if\ i\neq j\\ 0, otherwise\end{cases}$。

假定从图 G=(V,E)将学得一个实值函数 f:V→R，对应分类规则为 y_i=sgn($f(\boldsymbol{x}_i)$)。直观上看，相似的样本应具有形似的标记，于是可定义关于 f 的能量函数：

$$E(f)=\frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}(\mathrm{W})_{ij}(f(\boldsymbol{x}_i)-f(\boldsymbol{x}_j))^2=\boldsymbol{f}^T(\mathrm{D}-\mathrm{W})\boldsymbol{f}$$

。其中 $\boldsymbol{f}$=($f(\boldsymbol{x}_1),\cdots,f(\boldsymbol{x}_m)$)$^{\mathrm{T}}$，D=diag($d_1,\cdots,d_m$)是对角阵，其对角元素 d_i 是矩阵 W 的第 i 行元素之和。

具有最小能量的函数 f 在标记样本上满足 $f(\boldsymbol{x}_i)=y_i$，在未标记样本上满足 $\triangle \boldsymbol{f}=0$，其中 $\triangle=D-W$ 为拉普拉斯矩阵，采用分块矩阵 $W=\begin{bmatrix} W_{ll} & W_{lu} \\ W_{ul} & W_{uu} \end{bmatrix}$，$D=\begin{bmatrix} D_{ll} & 0_{lu} \\ 0_{ul} & D_{uu} \end{bmatrix}$，则上式可重写为 $E(f)=(\boldsymbol{f}_l^T \boldsymbol{f}_u^T)\left(\begin{bmatrix} D_{ll} & 0_{lu} \\ 0_{ul} & D_{uu} \end{bmatrix}-\begin{bmatrix} W_{ll} & W_{lu} \\ W_{ul} & W_{uu} \end{bmatrix}\right)\begin{pmatrix} \boldsymbol{f}_l \\ \boldsymbol{f}_u \end{pmatrix}$。由 $\frac{\partial E(f)}{\partial \boldsymbol{f}_u}=0$ 可知 $\boldsymbol{f}_u=(D_{uu}-W_{uu})^{-1}W_{ul}\boldsymbol{f}_l$。令 $P=D^{-1}W$，则重写为 $\boldsymbol{f}_u=(I-P_{uu})^{-1}P_{ul}\boldsymbol{f}_l$。将 $\boldsymbol{f}_l$ 代入即可。

下面来看一个适用于多分类问题的标记传播方法。定义一个 $m\times|Y|$ 的非负标记矩阵 $F=(F_1^T,\cdots,F_m^T)^T$，其第 i 行元素 F_i 为示例 $\boldsymbol{x}_i$ 的标记向量，相应的分类规则为 $y_i=\arg\max(F)_{ij}$。将 F 初始化为 $F(0)=(Y)_{ij}=\begin{cases}1, if\ (1\le i\le l)\bigcap(y_i=j)\\ 0, otherwise\end{cases}$，基于 W 构造一个标记传播矩阵 $S=D^{-\frac{1}{2}}WD^{-\frac{1}{2}}$，有迭代计算式 $F(t+1)=\alpha SF(t)+(1-\alpha)Y, \alpha\in(0,1)$，基于此至收敛可得 $F^*=\lim F(t)=(1-\alpha)(I-\alpha S)^{-1}Y$，从而获得 D_u 中样本的标记。

输入： 有标记样本集 $D_l=\{(\boldsymbol{x}_1,y_1),(\boldsymbol{x}_2,y_2),\ldots,(\boldsymbol{x}_l,y_l)\}$;
未标记样本集 $D_u=\{\boldsymbol{x}_{l+1},\boldsymbol{x}_{l+2},\ldots,\boldsymbol{x}_{l+u}\}$;
构图参数 σ;
折中参数 α.

过程：
1: 基于式(13.11)和参数 σ 得到 $\mathbf{W}$;
2: 基于 $\mathbf{W}$ 构造标记传播矩阵 $\mathbf{S}=\mathbf{D}^{-\frac{1}{2}}\mathbf{W}\mathbf{D}^{-\frac{1}{2}}$;
3: 根据式(13.18)初始化 $\mathbf{F}(0)$;
4: $t=0$;
5: **repeat**
6: $\quad\mathbf{F}(t+1)=\alpha\mathbf{S}\mathbf{F}(t)+(1-\alpha)\mathbf{Y}$;
7: $\quad t=t+1$
8: **until** 迭代收敛至 $\mathbf{F}^*$
9: **for** $i=l+1,l+2,\ldots,l+u$ **do**
10: $\quad y_i=\arg\max_{1\leqslant j\leqslant|\mathcal{Y}|}(\mathbf{F}^*)_{ij}$
11: **end for**

输出： 未标记样本的预测结果: $\hat{\boldsymbol{y}}=(\hat{y}_{l+1},\hat{y}_{l+2},\ldots,\hat{y}_{l+u})$

这对应正则化框架 $\min_F \frac{1}{2}\left(\sum_{i,j=1}^{m}(W)_{ij}\left\|\frac{1}{\sqrt{d_i}}F_i-\frac{1}{\sqrt{d_j}}F_j\right\|^2\right)+\mu\sum_{i=1}^{l}\|F_i-Y_i\|^2, \mu>0$。由于未标记样本往往很多，为了缓解过拟合，可在式中引入针对未标记样本的 L_2 范数项 $\mu\sum_{i=l+1}^{l+u}\|F_i\|^2$。特别地，若 $\mu=\frac{1-\alpha}{\alpha}$，此式最优解对应图中的收敛解 F*。

算法缺点：存储开销大，难以判知新样本在图中的位置（可以重新构图标记传播，也可以将 D_l 和 D_u 合并训练一个学习器）。

13.5 基于分歧的方法

使用多学习器，学习器之间的分歧对未标记数据的利用至关重要。

多视图数据：拥有多个属性集，每个属性集构成了一个视图，表示为$([\boldsymbol{x}_1,\boldsymbol{x}_2],y)$。假设不同视图具有相容性，即其所包含的关于输出空间 Y 的信息是一致的：令 Y_1 表示从信息 A 判别的标记空间，Y_2 表示从信息 B 判别的标记空间，则成立 $Y=Y_1=Y_2$。

协同训练：利用多视图的相容互补性。假设数据拥有两个充分且条件独立视图，“充分”是指每个视图都包含足以产生最优学习器的信息，“条件独立”则是指在给定类别标记条件下两个视图独立。在此情形下，可这样利用未标记数据：首先在每个视图上基于有标记样本分别训练出一个分类器，然后让每个分类器去挑选自己最有把握的未标记样本赋予伪标记，并将伪标记样本提供给另一个分类器作为新增的有标记样本用于训练更新……这个“相互学习、共同进步”的过程不断迭代，直至两个分类器都不再发生变化，或达到预先设定的迭代轮数。

输入： 有标记样本集 $D_l=\{(\langle \boldsymbol{x}_1^1,\boldsymbol{x}_1^2\rangle,y_1),\ldots,(\langle \boldsymbol{x}_l^1,\boldsymbol{x}_l^2\rangle,y_l)\}$;
未标记样本集 $D_u=\{\langle \boldsymbol{x}_{l+1}^1,\boldsymbol{x}_{l+1}^2\rangle,\ldots,\langle \boldsymbol{x}_{l+u}^1,\boldsymbol{x}_{l+u}^2\rangle\}$;
缓冲池大小 s;
每轮挑选的正例数 p;
每轮挑选的反例数 n;
基学习算法 $\mathfrak{L}$;
学习轮数 T.

过程：

从 D_u 中随机抽取 s 个样本构成缓冲池 D_s;
$D_u=D_u\setminus D_s$;
for $j=1,2$ **do**
  $D_l^j=\{(\boldsymbol{x}_i^j,y_i)\mid(\langle \boldsymbol{x}_i^j,\boldsymbol{x}_i^{3-j}\rangle,y_i)\in D_l\}$;
end for
for $t=1,2,\ldots,T$ **do**
  for $j=1,2$ **do**
    $h_j\leftarrow\mathfrak{L}(D_l^j)$;
    考察 h_j 在 $D_s^j=\{\boldsymbol{x}_i^j\mid\langle \boldsymbol{x}_i^j,\boldsymbol{x}_i^{3-j}\rangle\in D_s\}$ 上的分类置信度，挑选 p 个正例置信度最高的样本 $D_p\subset D_s$、n 个反例置信度最高的样本 $D_n\subset D_s$;
    由 D_p^j 生成伪标记正例 $\tilde{D}_p^{3-j}=\{(\boldsymbol{x}_i^{3-j},+1)\mid\boldsymbol{x}_i^j\in D_p^j\}$;
    由 D_n^j 生成伪标记反例 $\tilde{D}_n^{3-j}=\{(\boldsymbol{x}_i^{3-j},-1)\mid\boldsymbol{x}_i^j\in D_n^j\}$;
    $D_s=D_s\setminus(D_p\bigcup D_n)$;
  end for
  if h_1,h_2 均未发生改变 **then**
    break
  else
    for $j=1,2$ **do**
      $D_l^j=D_l^j\bigcup\left(\tilde{D}_p^j\bigcup\tilde{D}_n^j\right)$;
    end for
    从 D_u 中随机抽取 $2p+2n$ 个样本加入 D_s
  end if
end for

输出： 分类器 h_1, h_2

也可以使用不同学习算法、不同数据采样、不同参数设置来协同训练。此类算法无需数据拥有多视图，仅需弱学习器之间有差异，即可通过相互提供伪标记来提升泛化性能。

13.6 半监督聚类

聚类任务中获得的监督信息大致有两种类型。第一种是“必连”与“勿连”约束，前者是指样本必属于同一个簇，后者是指样本必不属于同一个簇；第二种类型的监督信息则是少量的有标记样本。

约束 k 均值算法是利用第一类监督信息的代表。

输入： 样本集 $D=\{\boldsymbol{x}_1,\boldsymbol{x}_2,\ldots,\boldsymbol{x}_m\}$;
必连约束集合 $\mathcal{M}$;
勿连约束集合 $\mathcal{C}$;
聚类簇数 k.

过程：

从 D 中随机选取 k 个样本作为初始均值向量$\{\boldsymbol{\mu}_1,\boldsymbol{\mu}_2,\ldots,\boldsymbol{\mu}_k\}$;
repeat
  $C_j=\varnothing\ (1\leqslant j\leqslant k)$;
  for $i=1,2,\ldots,m$ **do**
    计算样本 $\boldsymbol{x}_i$ 与各均值向量 $\boldsymbol{\mu}_j\ (1\leqslant j\leqslant k)$ 的距离: $d_{ij}=\|\boldsymbol{x}_i-\boldsymbol{\mu}_j\|_2$;
    $\mathcal{K}=\{1,2,\ldots,k\}$;
    is_merged=false;
    while ¬ is_merged **do**
      基于 $\mathcal{K}$ 找出与样本 $\boldsymbol{x}_i$ 距离最近的簇: $r=\arg\min_{j\in\mathcal{K}} d_{ij}$;
      检测将 $\boldsymbol{x}_i$ 划入聚类簇 C_r 是否会违背 $\mathcal{M}$ 与 $\mathcal{C}$ 中的约束;
      if ¬ is_voilated **then**
        $C_r=C_r\bigcup\{\boldsymbol{x}_i\}$;
        is_merged=true
      else
        $\mathcal{K}=\mathcal{K}\setminus\{r\}$;
        if $\mathcal{K}=\varnothing$ **then**
          break并返回错误提示
        end if
      end if
    end while
  end for
  for $j=1,2,\ldots,k$ **do**
    $\boldsymbol{\mu}_j=\frac{1}{|C_j|}\sum_{\boldsymbol{x}\in C_j}\boldsymbol{x}$;
  end for
until 均值向量均未更新

输出： 簇划分 $\{C_1,C_2,\ldots,C_k\}$

考虑第二种监督信息。给定样本集 D=$\{\boldsymbol{x}_1,\cdots,\boldsymbol{x}_m\}$，有少量的有标记样本为 S=∪$S_j$，其中非空集合 S_j 为隶属于第 j 个聚类簇的样本。可以直接将他们作为“种子”，用它们作为“种子”来初始化 k 均值算法的 k 个聚类中心，并且在聚类簇迭代更新过程中不改变种子样本的簇隶属关系，得到约束种子 k 均值算法。

输入： 样本集 $D=\{\boldsymbol{x}_1,\boldsymbol{x}_2,\ldots,\boldsymbol{x}_m\}$;
必连约束集合 $\mathcal{M}$;
勿连约束集合 $\mathcal{C}$;
聚类簇数 k.

过程：

从 D 中随机选取 k 个样本作为初始均值向量$\{\boldsymbol{\mu}_1,\boldsymbol{\mu}_2,\ldots,\boldsymbol{\mu}_k\}$;
repeat
  $C_j=\varnothing\ (1\leqslant j\leqslant k)$;
  for $i=1,2,\ldots,m$ **do**
    计算样本 $\boldsymbol{x}_i$ 与各均值向量 $\boldsymbol{\mu}_j\ (1\leqslant j\leqslant k)$ 的距离: $d_{ij}=\|\boldsymbol{x}_i-\boldsymbol{\mu}_j\|_2$;
    $\mathcal{K}=\{1,2,\ldots,k\}$;
    is_merged=false;
    while ¬ is_merged **do**
      基于 $\mathcal{K}$ 找出与样本 $\boldsymbol{x}_i$ 距离最近的簇: $r=\arg\min_{j\in\mathcal{K}} d_{ij}$;
      检测将 $\boldsymbol{x}_i$ 划入聚类簇 C_r 是否会违背 $\mathcal{M}$ 与 $\mathcal{C}$ 中的约束;
      if ¬ is_voilated **then**
        $C_r=C_r\bigcup\{\boldsymbol{x}_i\}$;
        is_merged=true
      else
        $\mathcal{K}=\mathcal{K}\setminus\{r\}$;
        if $\mathcal{K}=\varnothing$ **then**
          break并返回错误提示
        end if
      end if
    end while
  end for
  for $j=1,2,\ldots,k$ **do**
    $\boldsymbol{\mu}_j=\frac{1}{|C_j|}\sum_{\boldsymbol{x}\in C_j}\boldsymbol{x}$;
  end for
until 均值向量均未更新

输出： 簇划分 $\{C_1,C_2,\ldots,C_k\}$

14 概率图模型

14.1 隐马尔可夫模型

概率图模型是一类用图来表达变量相关关系的概率模型。最常见的是用一个结点表示一个或一组随机变量，结点之间的边表示变量间的概率相关关系。根据边的性质不同，可以将概率图模型分为两类：有向无环图（贝叶斯网），无向图（马尔可夫网）。

隐马尔可夫模型是结构最简单的动态贝叶斯网，主要用于时序数据建模。

隐马尔可夫模型中的变量分为两组，一组是状态变量$\{y_1,\cdots,y_n\}$，其中$y_i\in$Y 表示第 i 时刻的系统状态，通常假设为隐变量。第二组是观测变量$\{x_1,\cdots,x_n\}$，其中$x_i\in$X 表示第 i 时刻的观测值。在隐马尔可夫模型中，系统通常在多个状态之间转换，因此状态变量y_i的取值范围 Y 通常是有 N 个可能取值的离散空间。便于讨论仅考虑离散型观测变量，并假定取值范围 X=$\{o_1,\cdots,o_M\}$。

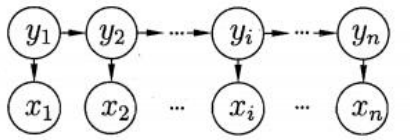


图 14.1 隐马尔可夫模型的图结构

图 14.1 的箭头表示了变量间的依赖关系。这是一个 Markov chain：系统下一时刻的状态仅有当前状态决定，不依赖于以往的任何状态。基于这种依赖关系，所有变量的联合概率分布为 $P(x_1,y_1,\cdots,x_n,y_n)=P(y_1)P(x_1\mid y_1)\prod_{i=2}^{n}P(y_i\mid y_{i-1})P(x_i\mid y_i)$。

参数为：1° 状态转移概率：模型在各个状态间转换的概率，记为矩阵 A；
2° 输出观测概率：模型根据当前状态获得各个观测值的概率，记为矩阵 B；
3° 初始状态概率：模型在初始时刻各状态出现的概率，记为向量π。

给定隐马尔可夫模型λ=[A,B,π]，它按如下过程产生观测序列$\{x_1,\cdots,x_n\}$：
1° 设置 t=1，并根据初始状态概率π选择初始状态 y_1；
2° 根据状态 y_t 和输出观测概率 B 选择观测变量取值 x_t；
3° 根据状态 y_t 和状态转移矩阵 A 转移模型状态，即确定 y_{t+1}；
4° 若 $t<n$，设置 $t=t+1$，并转到第 2 步，否则停止。

在实际应用中，人们常关注隐马尔可夫模型的三个基本问题：
• 给定模型，如何有效计算其产生观测序列的概率？（如何评估模型与观测序列之间的匹配程度？）
• 给定模型和观测序列，如何找到与此观测序列最匹配的状态序列？（如何根据观测序列推断出隐藏的模型状态？）
• 给定观测序列，如何调整模型参数使得序列出现的概率最大？（如何训练模型使其能最好地描述观测数据？）

例子：语音识别任务中，观测值为语音信号，隐藏状态为文字，即第二个问题；如何根据训练样本学得最优的模型参数，又是第三个问题。

14.2 马尔可夫随机场

马尔可夫网，无向图模型。图中每个结点表示一个或一组变量，结点之间的边表示两个变量间的依赖关系。有一组势函数，定义在变量子集上的非负实函数，主要用于定义概率分布函数。

对于图中结点的一个子集，若其中任意两结点间都有边连接，则称该结点子集为一个团；如若在一个团中加入另外任何一个结点都不再形成团，则称为极大团。在马尔科夫随机场中，多个变量间的联合概率能基于团分解为多个因子的乘积，每个因子仅与一个团相关。具体来说，对于 n 个变量 $\boldsymbol{x}=\{x_1,\cdots,x_n\}$，所有团构成的集合为 C，与团 Q∈C 对应的变量集合记为 $\boldsymbol{x}_Q$，则联合概率 P($\boldsymbol{x}$)定义为 P($\boldsymbol{x}$)=$\frac{1}{Z}\prod_{Q\in C}\phi_Q(\boldsymbol{x}_Q)$。其中 ϕ_Q 为与团 Q 对应的势函数，用于对团 Q 中的变量关系进行建模；$Z=\sum_x\prod_{Q\in C}\phi_Q(\boldsymbol{x}_Q)$ 是规范化因子。

注意到任意团都会被极大团包含，于是联合概率也可定义为 P($\boldsymbol{x}$)=$\frac{1}{Z^*}\prod_{Q\in C^*}\phi_Q(\boldsymbol{x}_Q)$，其中 $Z^*=\sum_x\prod_{Q\in C^*}\phi_Q(\boldsymbol{x}_Q)$ 是规范化因子，C*是所有极大团构成的集合。

若从结点集 A 中的结点到 B 中的结点都必须经过结点集 C 中的结点，则称结点集 A 和 B 被结点集 C 分离，C 称为分离集。对马尔可夫随机场，有“全局马尔可夫性”：给定两个变量子集的分离集，则这两个变量子集条件独立。推论：
- 局部马尔可夫性：给定某变量的邻接变量，则该变量条件独立于其他变量；
- 成对马尔可夫性：给定所有其他变量，两个非邻接变量条件独立。

现在来考察势函数，其作用是定量刻画变量集 $\boldsymbol{x}_Q$ 中变量之间的相关关系，应该是非负函数，且在所偏好的变量取值上有较大函数值。

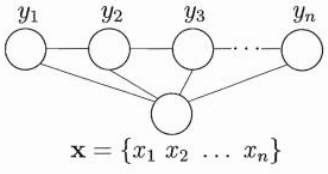
为了满足非负性，指数函数常被用于定义势函数 $\phi_Q(\boldsymbol{x}_Q)=e^{-H_Q(\boldsymbol{x}_Q)}$，其中 $H_Q(\boldsymbol{x}_Q)$ 是定义在变量 $\boldsymbol{x}_Q$ 上的实值函数，常见形式为 $H_Q(\boldsymbol{x}_Q)=\sum_{u,v\in Q,u\neq v}\alpha_{uv}x_ux_v+\sum_{v\in Q}\beta_vx_v$。

14.3 条件随机场

一种判别式无向图模型，对条件分布进行模型。

试图对多个变量在给定观测之后的条件概率进行建模。令 $\boldsymbol{x}=\{x_1,\cdots,x_n\}$ 为观测序列，$\boldsymbol{y}=\{y_1,\cdots,y_n\}$ 为与之相应的标记序列，则条件随机场的目标是构建条件概率模型 P($\boldsymbol{y}|\boldsymbol{x}$)。令 G=<V,E>表示结点与标记变量 $\boldsymbol{y}$ 中元素一一对应的无向图，y_v 表示与结点 v 对应的标记变量，$n(v)$表示结点 v 的邻接结点，若图 G 的每个变量 y_v 都满足马尔可夫性，即 P($y_v|\boldsymbol{x},\boldsymbol{y}_{V\setminus\{v\}}$)=P($y_v|\boldsymbol{x},\boldsymbol{y}_{n(v)}$)，则($\boldsymbol{y},\boldsymbol{x}$)构成一个条件随机场。

常用链式条件随机场。



与马尔可夫随机场定义联合概率的方式类似，条件随机场使用势函数和图结构上的团来定义条件概率 P($\boldsymbol{y}|\boldsymbol{x}$)。给定观测序列 $\boldsymbol{x}$，链式条件随机场主要包含两种关于标价变量的团，即单个标记变量 y_i 以及相邻的标记变量 $\{y_{i-1},y_i\}$。从而 $P(\boldsymbol{y}\mid\boldsymbol{x})=\frac{1}{Z}\exp\left(\sum_j\sum_{i=1}^{n-1}\lambda_jt_j(y_{i+1},y_i,\boldsymbol{x},i)+\sum_k\sum_{i=1}^{n}\mu_ks_k(y_i,\boldsymbol{x},i)\right)$，其中 $t_j(y_{i+1},y_i,\boldsymbol{x},i)$是定义在观测序列的两个相邻标记位置上的转移特征函数，用于刻画相邻标记变量之间的相关关系以及观测序列对它们的影响，$s_k(y_i,\boldsymbol{x},i)$是定义在观测序列的标记位置 i 上的状态特征函数，用于刻画观测序列对标记变量的影响。

14.4 学习与推断

可将参数视为待推测变量，下面只讨论概率图模型的推断方法。

假设图模型所对应的变量集 $\boldsymbol{x}=\{x_1,\cdots,x_N\}$能分为 $\boldsymbol{x}_E$ 和 $\boldsymbol{x}_F$ 两个不相交的变量集，推断问题的目标就是计算边集概率 $P(\boldsymbol{x}_F)$或条件概率 $P(\boldsymbol{x}_F|\boldsymbol{x}_E)$。由于联合概率可基于概率图模型获得，因此推断问题的关键是如何高效地计算边际分布 $P(\boldsymbol{x}_E)$。大致分为两类：一类是精确推断方法，一类是近似推断方法。本节介绍前者。

• 变量消去

动态规划算法，利用图模型描述的条件独立性来削减计算目标概率值所需的计算量。直接计算边际分布。缺点是若要计算多个，会造成大量冗余。

• 信念传播

记 $m_{ij}(x_j)$是中间结果，i 表示对 i 相加，j 表示剩下的其他变量。

$m_{ij}(x_j)=\sum_{x_i}\psi(x_i,x_j)\prod_{k\in n(i)\setminus j}m_{ki}(x_i)$，n(i)表示 x_i 的邻接结点。 $P(x_i)\propto\prod_{k\in n(i)}m_{ki}(x_i)$。

如果图结构没有环，则信念传播算法只需经过两个步骤：

1. 指定根结点，从所有叶结点开始向根节点传递消息，直到根结点收到所有邻接结点的消息；
2. 从根结点开始向叶结点传递消息，直到所有叶结点均收到消息。

这样可以获得所有的 $m_{ij}(x_j)$，从而计算出 $P(x_i)$。

14.5 近似推断

• MCMC 采样

构造平稳分布为 p 的马尔科夫链。设 T(x'|x)为 x 到 x'的转移概率，则若在某个时刻马尔科夫链满足平稳条件 $p(x_t)T(x_{t-1}|x_t)=p(x_{t-1})T(x_t|x_{t-1})$，则 p(x)是平稳分布。

MH 算法：拒绝采样逼近平稳分布。接受率为 $A(x^*|x_{t-1})=\min(1,\frac{p(x^*)T(x_{t-1}|x^*)}{p(x_{t-1})T(x^*|x_{t-1})})$

输入： 先验概率 $Q(\mathbf{x}^*\mid\mathbf{x}^{t-1})$.
过程：
1: 初始化 $\mathbf{x}^0$;
2: **for** $t=1,2,\ldots$ **do**
3: 根据 $Q(\mathbf{x}^*\mid\mathbf{x}^{t-1})$ 采样出候选样本 $\mathbf{x}^*$;
4: 根据均匀分布从 $(0,1)$ 范围内采样出阈值 u;
5: **if** $u\leqslant A(\mathbf{x}^*\mid\mathbf{x}^{t-1})$ **then**
6: $\mathbf{x}^t=\mathbf{x}^*$
7: **else**
8: $\mathbf{x}^t=\mathbf{x}^{t-1}$
9: **end if**
10: **end for**
11: **return** $\mathbf{x}^1,\mathbf{x}^2,\ldots$
输出： 采样出的一个样本序列 $\mathbf{x}^1,\mathbf{x}^2,\ldots$

吉布斯采样：随机选取某变量 x_i，根据 $\boldsymbol{x}$ 中除 x_i 变量现有取值，计算 $p(x_i|\cdots)$，根据这个概率 x_i 进行采样，代替原来值。

• 变分推断

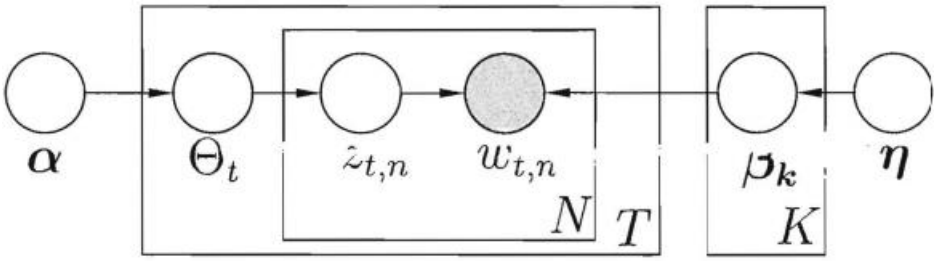
用简单分布逼近复杂分布。概率密度函数 $p(\boldsymbol{x}|\Theta)=\prod_{i=1}^{N}\sum_{z}p(x_i,z\mid\Theta)$，任务是由观

察到的 $\boldsymbol{x}$ 来估计隐变量 z 和分布参数变量Θ。采用 EM 算法。实际上 E 步计算 p(z|$\boldsymbol{x}$,Θ)往往很负责，可使用变分推断。

假设 z 服从分布 $q(z)=\prod_{i=1}^{M} q_i(z_i)$，即复杂的 z 可以拆成相互独立的多变量 z_i，可以令 q_i 分布相对简单（假设为指数族分布）。$q_j(z_j)=\frac{\exp(E_{i\neq j}[\ln p(x,z)])}{\int \exp(E_{i\neq j}[\ln p(x,z)])dz_j}$。恰当地分割独立变量子集 z_j 并选择 q_i 服从的分布，$E_{i\neq j}[\ln p(x,z)]$往往有闭式解，这样就可以高效地对隐变量 z 进行推断。

14.6 话题模型

• 隐狄利克雷分配



其中$\boldsymbol{\alpha}$是狄利克雷分布参数，Θ是话题分布，$z_{t,n}$ 是某个词的话题指派，β_k 是某话题所对应的词频。

LDA 生成步骤：

1° 根据参数为α的狄利克雷分布采样一个话题分布Θ；

2° 按以下步骤生成文档中的 N 个词：

a) 根据Θ进行话题指派，得到文档中词 n 的话题 $z_{t,n}$；

b) 根据指派的话题所对应的词频分布β_k 随机采样生成词。