分类模型

1 基础知识

1.1 机器学习分类

通俗来说，机器学习是一门讨论各种各样适用于不同问题的函数形式，以及如何使用数据来有效地获取函数参数具体值的学科。

按以下标准可以将他们进行大的分类：

- 是否在监督下训练（有监督学习，无监督学习，半监督学习，强化学习）
- 是否可以动态增量学习（在线学习，批量学习）

有监督学习任务：回归，分类	算法：线性回归，逻辑回归，k-近邻，SVM，决策树和随机森林
无监督学习任务：聚类	聚类算法：k-均值，DBSCAN，分层聚类
	异常检测和新颖性检测：单类 SVM，孤立森林
	可视化和降维：PCA，核主成分分析，局部线性嵌入 LLE，t-分布随机近邻嵌入 t-SNE
	关联规则学习：Apriori，Eclat

（1）异常检测：当新实例在训练集中很罕见时，可能被模型归类为异常。

新颖性检测：当新实例在训练集中没有出现过时，可能被模型归类为新颖。

（2）降维：例如汽车里程与其使用年限存在很大的相关性，所以降维算法会将他们合并成一个代表汽车磨损的特征，这个过程叫做特征提取。

（3）关联规则学习：例如买烧烤酱和薯片的人也倾向于买牛排，那么会将这几样商品摆放得更近一些。

（1）在线增量学习：指一个学习系统能不断地从新样本中学习新的知识，并能保存大部分以前已经学习到的知识。

（理论上说，只要能是用 SGD 进行优化的模型都是可以实现增量训练的）

（2）批量学习：使用所有的可用数据进行训练，这通常是离线完成的，如果学

习新数据则需要重新训练一遍模型。

1.2 为什么要进行特征工程

机器学习的主要挑战：选择训练算法对数据进行训练，出现的问题无外乎就是坏算法和坏数据：

- 训练数据的数量不足（需要成千上万）
- 训练数据的质量不足（测量粗糙，含噪声异常值）
- 训练数据不具代表性（如调查中国 34 个省的情况，缺没有河南省的数据）
- 无关特征（需要足够多的相关特征和较少的无关特征）

一个成功的机器学习项目，关键是提取出一组好的用来训练的特征集，这个过程叫做特征工程。

（1）特征选择（从现有特征中选择最有用的特征进行训练）

（2）特征提取（将特征进行整合，产生更有用的特征）

（3）通过收集新数据创建新特征。

1.3 分类评价指标

任务	指标	公式
二分类	查准率（尽可能选出来的都是好瓜）Precision	TP/(TP+FP)
	查全率（尽可能把好瓜全部选出来）Recall	TP/(TP+FN)
	PR 曲线	/
	F_β	/
	真正例率 TPR	TP/(TP+FN)
	假正例率 FPR	FP/(FP+TN)
	ROC 曲线（AUC 值）	/
多分类	1，化成多个二分类，由每个混淆矩阵的 TP,FP,FN,TN 计算每个的 P，R 求平均，再计算 F 值 2，化成多个二分类，由每个混淆矩阵的 TP,FP,FN,TN 求平均，计算 1 个总的 P，R，再计算 F 值	/

2 数据质量检测

2.1 数据合并

2.2 数据清洗

（1）删除重复值

（2）填补缺失值

（3）修改异常值

对于异常值的判断，主要有两种方式：当数据满足正态分布或者近似正态分布时，可以采用3σ原则进行异常值的判断；当数据满足其他分布时，可以采用箱线图进行异常值的判断。判断异常值后，可以把异常值赋值为Nan，然后按照缺失值方式进行处理；或者把异常值修改为特定的值。

在正态分布中σ代表标准差，μ代表均值，$x=\mu$即为图像的对称轴。3σ原则为：数值分布在$(\mu-\sigma,\mu+\sigma)$中的概率为0.6826，数值分布在$(\mu-2\sigma,\mu+2\sigma)$中的概率为0.9545，数值分布在$(\mu-3\sigma,\mu+3\sigma)$中的概率为0.9973，可以认为，数值几乎全部集中在$(\mu-3\sigma,\mu+3\sigma)$区间内，超出这个范围的可能性仅占不到0.3%，超出这个范围的数值视为异常值。

箱线图（Box-plot）又称为盒须图、盒式图或箱形图，是一种用作显示一组数据分散情况的统计图，因形状如箱子而得名。它主要用于反映原始数据分布的特征，还可以进行多组数据分布特征的比较。箱线图的绘制方法是：先找出一组数据的上边缘、下边缘、中位数和两个四分位数；然后，连接两个四分位数画出箱体；再将上边缘和下边缘与箱体相连接，中位数在箱体中间。利用箱线图分析时，异常值通常被定义为小于QL-1.5*IQR或大于 QU+1.5IQR（QL指下四分位数，QU指上四分位数，IQR指QU-QL）。

2.3 数据可视化

2.4 数据标准化

在进行特征工程之前，我们需要根据算法特性确定是否对特征进行缩放，特征缩放主要是对数据进行比例尺的转换，使得不同量级的特征具有相同的数值范围，这样可以使得模型训练更加稳定和有效。

（1）归一标准化：将特征数据的区间缩放到[0,1]内，且和为1

$$X_i^n=\frac{X_i^n}{sum(X^n)} \tag{1}$$

其中，X_i^n表示第i个数据元素的第n个特征数据，X^n为所有数据元素的第n个特征数据。

（2）离差标准化：将特征数据的区间缩放到[0,1]内

$$X_i^n = \frac{X_i^n - \min(X^n)}{\max(X^n) - \min(X^n)} \tag{2}$$

（3）Z-score 标准化：将特征数据缩放为均值为 0，方差为 1 的数据

$$X_i^n = \frac{X_i^n - \mu}{\sigma} \tag{3}$$

其中，X_i^n表示第i个数据元素的第n个特征数据，μ,σ分别为所有数据元素的第n个特征数据的均值和方差。

2.5 数据类别不平衡处理

分类学习算法都有一个共同的基本假设，即不同类别的训练样本数目相当，如果不同类别的训练样例数目稍有差别，通常影响不大，但若差别很大，则会对学习过程造成困扰。

由于目标变量有 x 个 0，y 个 1（划分集合后的类别数），数据集存在类别不平衡，因此需要通过几种基本措施来均衡类别。

（1）欠采样（下采样）：

对于多数类别（这里是类别 0），可以随机选择一部分样本，使得类别 0 和类别 1 的样本数量一致，然后再进行训练。这种方法的缺点是可能会丢失一些有用的信息。

（2）过采样（上采样）：

对于少数类别（这里是类别 1），可以通过复制样本或者生成合成样（SMOTE 等算法）来增加样本数量，使得类别 0 和类别 1 的样本数量一致。这种方法的缺点是可能会过拟合，因为少数类别的样本信息会被重复使用。

（3）阈值移动：

我们对新样本x进行分类时，实际在用预测出来的y与一个阈值进行比较，例如通常在$y > 0.5$时判别为正例，否则为反例。y实际代表了正例的可能性，几率$y/(1-y)$则反映了正例可能性与反例可能性的比值，阈值设置为 0.5 恰好表明了分类器认为正反例可能性相同，即分类器决策规则为：

$$\text{若 } y/(1-y) > 1, \text{则预测为正例}$$

当训练集正反例数目不同时，令 m^+, m^- 表示正反例数目，则观测几率为 m^+/m^-，由于我们通常假设训练集是真实样本总体的无偏采样，因此观察几率就代表真实几率。于是分类器的预测几率高于观察几率就判定为正例，即

$$若\ y/(1-y) > m^+/m^-,\ 则预测为正例 \qquad (4.9)$$

如果进行欠采样，则只有 25 个正例数据和 25 个反例数据进行训练，数据量太少并且过拟合风险太大，因此本文不采取这种方式。

阈值移动的思想虽然简单，但是“训练集是真实样本总体的无偏采样”这个假设往往不成立，因此本文也不采取这种方式。

本文采用**过采样方法中的 SMOTE 算法**，其基本思想是，对于每一个少数类别的样本，根据其在特征空间中的邻近情况进行样本合成。这样合成的样本既保留了原样本的一些特性，又加入了一些随机性，可以有效地扩充数据集中的少数类别样本。

（对二分类使用 OvR 策略解决多分类时，每个类都被不平等对待，不平衡的影响会相互抵消，因此不需专门处理）

3 特征工程

建立基于相关性分析与灰色关联分析的冗余特征筛选模型，建立基于随机森林分析与主成分分析的集成特征提取模型。经过**特征工程**后可以选取**独立性较好**且同时具有**显著重要性**的特征，作为分类模型的输入。

3.1 特征选择

在建立血肿扩张风险预测模型之前，我们需要对扩张风险相关因素进行探索分析，在上节中我们已经提取出来（160，73）的数据集，如果将 73 个特征数据直接作为模型的输入势必增加欠拟合的风险，而我们要选取的特征是不包含高度相关性的重复特征，是独立性较好同时具有显著重要性的特征。因此我们需要进行特征工程。

特征工程包括特征选择和特征提取，特征选择是从初始特征空间选择一个最优特征子空间的过程，特征提取是一种基于变换的方法，通过坐标变换的方式将原始的高维数据映射到低维空间，可以保留原始特征空间大部分性质并去除不相关特征和冗余特征，在一定程度上将噪声数据对分类器性能的影响降到最低。

（1）相关性分析

目前主要采用的包括 Pearson 最大相关系数分析方法、Kendall 最大相关系数分析方法、Spearman 最小关联系数，下面比较三种方法对本题的适用性。

皮尔逊（Pearson）相关系数：是一种判断两组统计整体相互之间是不是连在同一根线上，也用来判断两定距变量相互之间的直线联系。当两变量均为连续变量，且两变量的总体为正态分布以及两变量之间为线性关系时可使用 Pearson 的方法。两变量间的皮尔逊相关系数可定义为两变量间的协方差与标准差的比商：

$$r=\frac{\sum_{i=1}^{m}(X_i^n-\bar{X}^n)(Y_i^n-\bar{Y}^n)}{\sqrt{\sum_{i=1}^{m}(X_i^n-\bar{X}^n)^2}\sqrt{\sum_{i=1}^{m}(Y_i^n-\bar{Y}^n)^2}} \tag{1}$$

肯德尔（Kendall）相关系数：是一种秩相关系数，可以处理有序的类别变量，将 n 个同类的统计对象按特定属性排序，同序对（concordant pairs）和异序对（discordant pairs）之差与总对数 $m(m-1)/2$ 的比值定义为 Kendall(肯德尔)系数。

$$\tau_B=\frac{n_c-n_d}{m(m-1)/2} \tag{2}$$

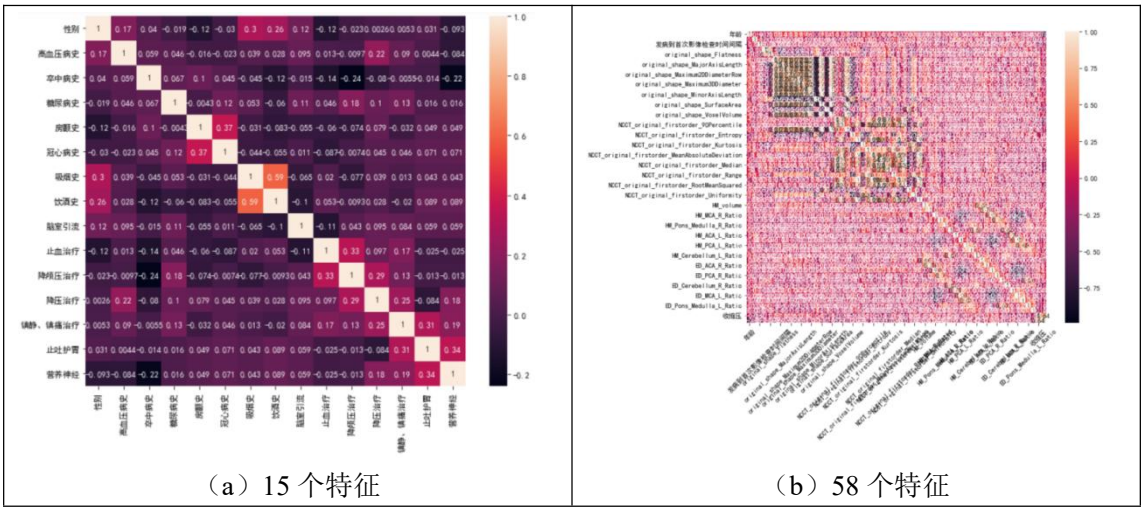
其中，n_c 表示有序对的数目，n_d 表示无序对的数目。

斯皮尔曼（Spearman）相关系数：系数被定义成等级变量之间的皮尔逊相关系数，当两变量总体不为正态分布时可以采用斯皮尔曼分析，该变量不需要关心数据如何变化，符合什么样的分布，只需要关心每个变量对应数值的位置。如果两个变量的对应值，在各
组内的排列顺位是相同或类似的，则具有显著的相关性。对于样本容量为 m 的样本，m 个原始数据被转换成等级数据，相关系数 ρ 的公式表示为：

$$\rho=1-\frac{6\sum_{i=1}^{m}d_i^2}{m(m^2-1)} \tag{3}$$

其中，d_i 表示两组数据第 i 个次序的差值。

问题 1（b）用到的数据集尺寸为（160，73），对于患者的疾病史、治疗史特征，共 15 个，其取值为 0 和 1，本文采用上一小节的斯皮尔曼相关性进行分析，对于其他连续性特征，共 58 个，本文采用皮尔逊相关性进行分析。结果如下图展示：



（a）15 个特征　　（b）58 个特征

图 4-5 对 15 个特征（斯皮尔曼）和 58 个特征（皮尔逊）的相关分析

根据表 1，我们找出相关系数大于阈值 0.8 的位置，定义两个变量为强相关变量，按遍历顺序并将其中之一剔除，即可剔除类内强相关的冗余变量。最后结果是剔除 27 个特征，保留 46 个特征。

表 4-4 皮尔逊相关系数和关联强度

相关系数	关联强度
0.8-1	极强相关
0.6-0.8	强相关
0.4-0.6	中等程度相关
0.2-0.4	弱相关
0-0.2	极弱相关或无相关

（2）灰色关联分析

在对一个系统进行研究之前，我们往往需要对系统进行因素分析，这些因素中哪些对系统来讲是主要的，哪些是次要的，哪些需要发展，哪些需要抑制，哪些是潜在的，哪些是明显的。这就需要将因素间关联程度进行量化，从而确定最主要的影响因素来达到简化问题的目的。

灰色关联分析的核心思想是通过计算参考数据列与若干个对比数据列的几何形状的相似程度来确定其关联是否紧密，假设选取参考数列和 m 个比较数列，

$$x^{(0)}=\{x^{(0)}(i)\,|\,i=1,2,...,n\},x^{(k)}=\{x^{(k)}(i)\,|\,i=1,2,...,n,k=1,2,...,m\}$$

其中 i 表示时刻。

则称

$$\xi^{(k)}(i)=\frac{\min\limits_{s}\min\limits_{t}|x^{(0)}(t)-x^{(s)}(t)|+\rho\max\limits_{s}\max\limits_{t}|x^{(0)}(t)-x^{(s)}(t)|}{|x^{(0)}(i)-x^{(k)}(i)|+\rho\max\limits_{s}\max\limits_{t}|x^{(0)}(t)-x^{(s)}(t)|} \tag{4}$$

为比较数列 $x^{(k)}$ 对参考数列 $x^{(0)}$ 在 i 时刻的关联系数，其中 $\rho\in[0,1]$ 为分辨系数，一般来讲，分辨系数越大，分辨率越高，$\min\limits_{s}\min\limits_{t}|x^{(0)}(t)-x^{(s)}(t)|$ 为两极最小差，$\max\limits_{s}\max\limits_{t}|x^{(0)}(t)-x^{(s)}(t)|$ 为两极最大差。

（4）式定义的关联系数是描述比较数列与参考数列在某时刻关联程度的一种指标，由于各个时刻都有一个关联数，那么就有 mn 个关联系数，信息显得过于分散，不便于比较，为此我们给出

$$r_k=\frac{1}{n}\sum_{i=1}^{n}\xi^{(k)}(i) \tag{5}$$

为比较数列 $x^{(k)}$ 对参考数列 $x^{(0)}$ 的关联度。

由（5）易看出，关联度是把各个时刻的关联系数集中为一个平均值，亦即把过于分散的信息集中处理。利用关联度这个概念，我们可以对各种问题进行因素分析。

由于在计算关联系数时，要求量纲相同，所以先对特征进行缩放处理。然后绘制出灰色关联度矩阵可视化图像：

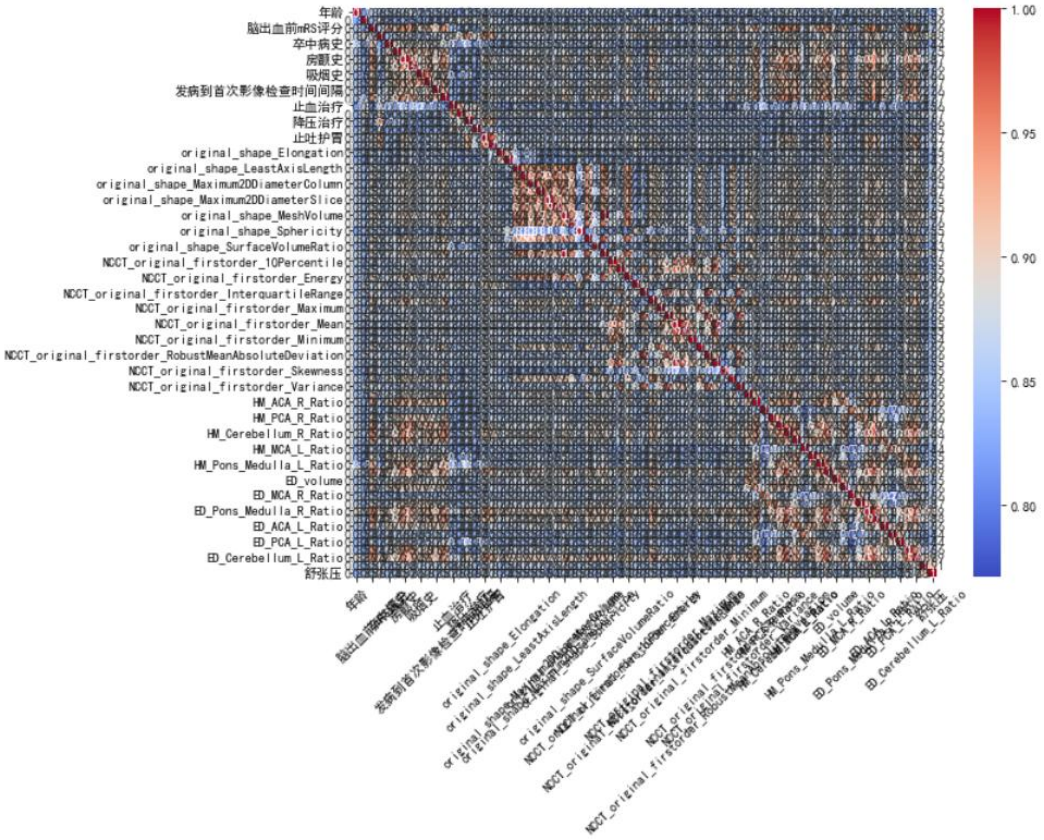


图 4-6 灰色关联度矩阵可视化图像

找出关联度大于阈值 0.95 的位置，定义两个变量为强相关变量，按遍历顺序并将其中之一剔除，即可剔除类内强相关的冗余变量。最后结果是剔除 36 个特征，保留 37 个特征。

3.2 特征提取

（1）随机森林分析

随机森林是一种集成学习方法，根据个体学习器的生成方式，目前的集成学习方法大致可以分为两大类，即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，以及个体学习器间不存在强依赖关系、可同时生成的并行化方法；前者的代表是 Boosting，后者的代表是 Bagging 和“随机森林”。

随机森林衡量每个特征相对重要性的原理：通过比较使用该特征的树节点平均（在森林中的所有树上）减少不纯度的程度来衡量该特征的重要性。更准确地说，它是一个加权平均值，其中每个节点的权值等于与其关联的训练样本的数量。

随机森林的变量选择过程如下表所示：

表 1 基于随机森林分析的特征提取过程

输入：前 100 个患者的 73 维特征向量数据； 输出：标签数据（是否发生血肿扩张）
初始化随机森林 利用标签数据对随机森林进行训练 计算特征重要度 计算特征重要度排序索引

while True:
    初始化新随机森林
    利用标签数据对随机森林进行训练
    计算特征重要度
    计算特征重要度排序索引
    计算 N 次平均特征重要度
    计算 N 次平均特征重要度排序索引
    if 平均特征重要度排序索引三轮未改变：
        return 特征重要度排序索引
根据特征重要度排序索引选择特征

我们绘制出变量重要性的条形图，

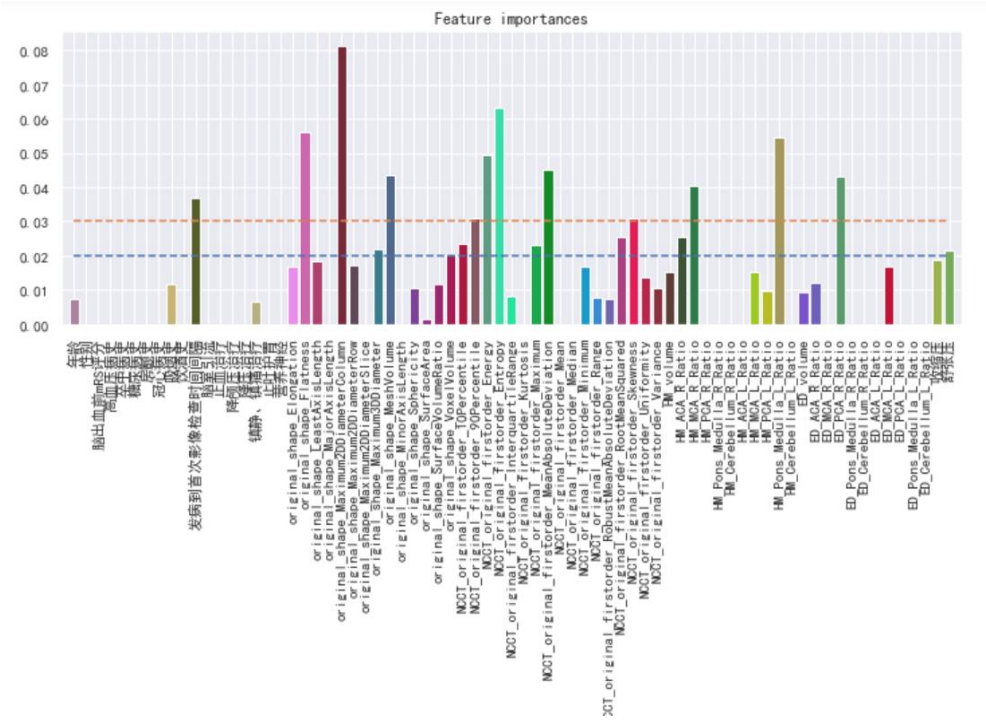


图 4-7 随机森林变量重要性

找出重要性程度小于阈值 0.03 的位置，定义非重要性变量，按遍历顺序并将其剔除，最后结果是剔除 61 个特征，保留 12 个特征，将这 12 个特征重要性按从大到小排序，绘制条形图：

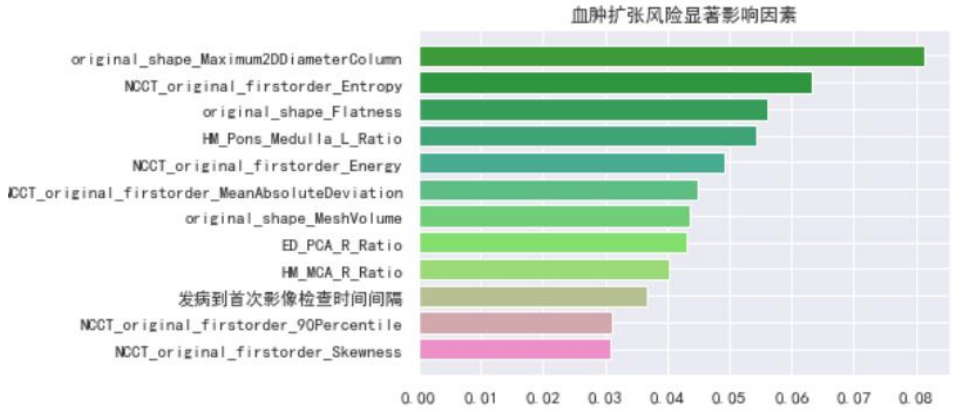


图 4-8 血肿扩张风险显著影响因素

血肿扩张风险显著影响因素前三位是：'original_shape_Maximum2DDiameterColumn'、'NCCT_original_firstorder_Entropy'、'original_shape_Flatness'。

（2）主成分分析（PCA）

主成分分析将原始数据进行线性变换、映射到低维空间中，使得各维度线性无关的表示，可用于提取数据的主要特征分量，其原理可以通过如下的一个示例进行解释：

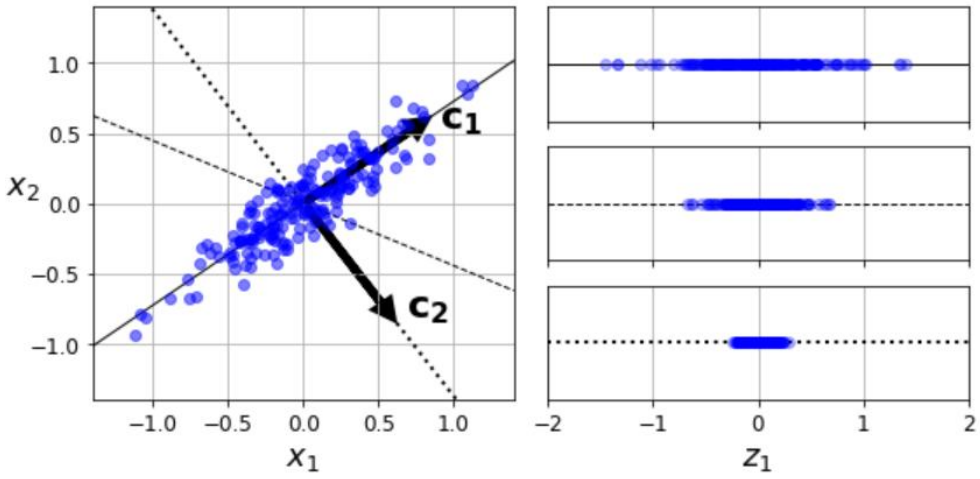


图 2 主成分分析示例解释

原理是识别最靠近数据的超平面，然后将数据投影到其上，直观看就是用实线作为一维超平面，这样投影才能保留数据最多的差异性，量化就是所有实例到投影超平面的均方误差最小。

由于在计算主成分时，要求量纲相同，所以先对特征进行缩放处理。前 30 个主成分累计贡献度为 0.9165420002888233，绘制重要程度条形图：

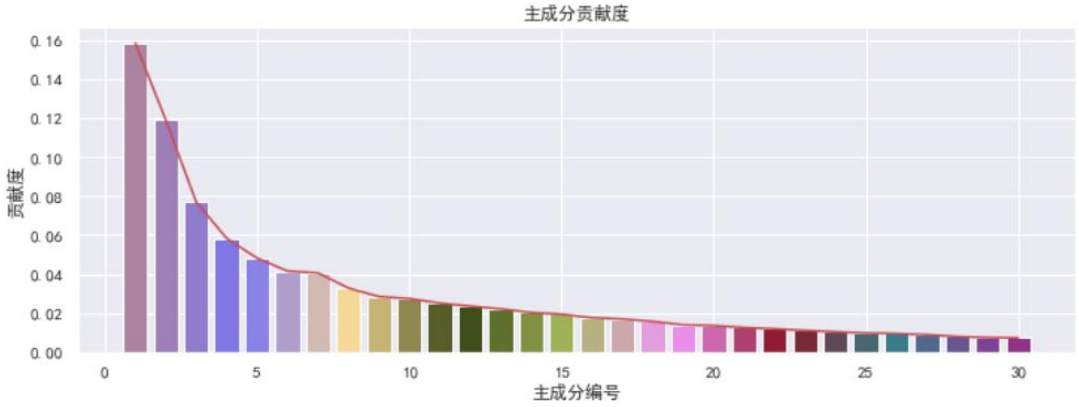


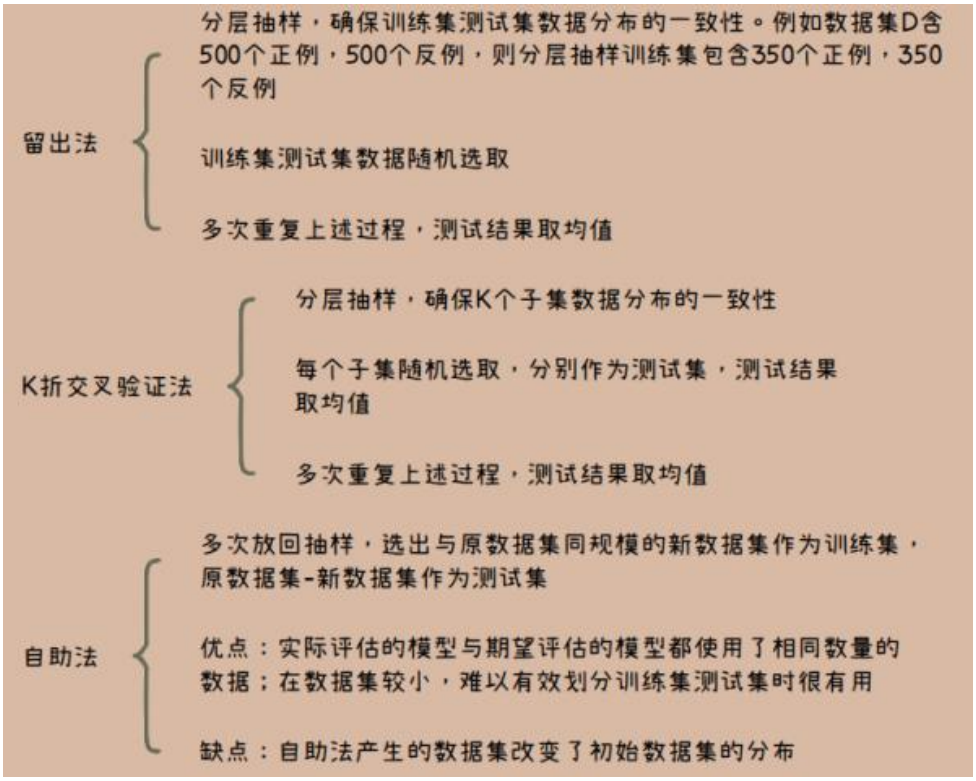
图 4-9 主成分贡献度

最后结果是保留 30 个主成分。

4 划分训练测试验证集

划分训练集（训练模型），测试集（评估模型泛化能力）的方法有留出法，

K 折交叉验证法，自助法，再可以通过训练集划分训练集和验证集。



注：10 次 10 折交叉验证等价于 100 次留出法。

5 KNN 模型

k 近邻（k-Nearest Neighbor，简称 KNN）学习是一种常用的监督学习方法，其工作机制非常简单：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，然后基于这 k 个“邻居”的信息来进行预测，对于分类任务，通常采用多数投票法，即在 K 个最近邻中多数类别为预测类别；对于回归任务，则通常是邻居的平均值。算法步骤如下：

（1）计算距离：计算测试样本与训练集中的每个样本之间的距离。

（2）找邻居：将最近的 k 个样本作为测试样本的邻居。

（3）做分类：根据 k 个邻居的主要类别对测试样本分类。

距离	定义式	说明
绝对值距离	$d_{ij}(1)=\sum_{k=1}^{p}\lvert x_{ik}-x_{jk}\rvert$	一维空间下进行的距离计算
欧式距离	$d_{ij}(2)=\sqrt{\sum_{k=1}^{p}(x_{ik}-x_{jk})^2}$	二维空间下进行的距离计算
闵可夫斯基距离	$d_{ij}(q)=[\sum_{k=1}^{p}(x_{ik}-x_{jk})^q]^{1/q},q>0$	q 维空间下进行的距离计算
切比雪夫距离	$d_{ij}(\infty)=\max_{1\le k\le p}\lvert x_{ik}-x_{jk}\rvert$	q 取无穷大时的闵可夫斯基距离

Lance 距离 $$d_{ij}(L)=\sum_{k=1}^{p}\frac{|x_{ik}-x_{jk}|}{x_{ik}+x_{jk}}$$ 削弱极端值的影响能力

优缺点分析：

优点	缺点	适合的应用场景
1，简单易懂：算法逻辑简单，易于理解和实现 2，不需要训练：没有显式学习过程，省去了训练时间 3，自适应：理论上可以对任何分布的数据进行预测 4，多功能性：可用于分类、回归甚至推荐系统	1，计算密集型：对于每个测试样本，都需要对所有训练样本计算距离，计算量大 2，存储需求高：需要保留所有训练数据 3，维数灾难：在高维空间中，距离度量可能不再有效，导致性能下降 4，对不平衡数据敏感：在类分布不平衡的数据集上，少数类可能被忽视	1，小规模数据集：在小至中等规模数据集上效果较好，计算量可控 2，基线模型：常用作基线方法，快速实现，用于性能比较 3，实时决策：由于其惰性学习特点，适合需要模型动态更新的实时决策系统 4，低维数据：适合低至中等维度的数据，在高维数据上表现不佳

总的来说，KNN 在小至中等规模、低维且类别平衡的数据集上表现较好，且可以作为许多问题的初步探索方法。

模型参数	解释	默认值
n_neighbors	查询的邻居数量	5
weights{*'uniform', 'distance'*}	权重函数，uniform 表示邻域内所有样本权重一样，distance 表示按距离的倒数进行加权	uniform
algorithm{*'auto', 'ball_tree', 'kd_tree', 'brute'*}	计算最近邻的算法，auto 表示根据传入 fit 的值选择最合适的算法，brute 表示蛮力搜索	auto
leaf_size	传递给 BallTree 或 KDTree 的叶大小。这可能会影响构造和查询的速度，以及存储树所需的内存。最佳值取决于问题的性质。	30
p	闵可夫斯基度量的功率参数。当 p = 1 使用曼哈顿距离（l1）和 p=2 欧几里得距离（l2）。对于任意 p，使用闵可夫斯基距离（l_p）。该参数应为正数。	2
metric	用于距离计算的度量	*minkowski*

6 朴素贝叶斯模型

6.1 原理介绍

朴素贝叶斯的直观思想是将测试样本的标签作为原因，测试样本的特征值作为结果，利用数据集的先验知识，按照贝叶斯公式“已知结果求原因概率”。

【贝叶斯公式（已知结果求原因公式，可以认为是对原因 A 事件先验概率的一种修正）：

$$P(A_i \mid B)=\frac{P(A_iB)}{P(B)}=\frac{P(B \mid A_i)P(A_i)}{\sum_j P(B \mid A_i)P(A_i)}$$

其中 A_i 是某个原因事件，B 是结果事件。

贝叶斯公式常被用来“已知结果求原因”，比如钥匙掉了，掉在宿舍里、掉在教室里、掉在路上的概率分别为40%、35%、25%，而掉在上述三处地方被找到的概率分别为0.8、0.3和0.1.问找到了钥匙，求它恰是在宿舍找到的概率？

假设结果事件B表示丢的钥匙被找到了，原因事件A1，A2，A3表示钥匙掉在宿舍里，掉在教室里掉在路上，则根据贝叶斯公式，问题求解的概率为：

P(A1|B)=P(A1*B)/P(B)
=P(A1)*P(B|A1)/[P(A1)*P(B|A1)+P(A2)*P(B|A2)+P(A3)*P(B|A3)]
=0.4*0.8/(0.4*0.8+0.35*0.3+0.25*0.1)
=0.32/0.45
=0.71

如果把在宿舍找到的概率0.8看作先验概率，那么0.71就是对0.8的修正，叫做后验概率，这也解释了最大后验概率估计(EM)的“后验”两字。

】

把瓜的标签当作原因，瓜的特征值当作结果，也就是求 $p(y \mid x)=\frac{p(xy)}{p(x)}=\frac{p(x \mid y)p(y)}{p(x)}$，而 $p(x \mid y)=p(x_1,x_2,...,x_n \mid y)$ 无法或很难获得（因为很多 x 的组合没有出现在样本集）。

因此“朴素”假设样本特征相互独立，则 $p(x \mid y)=p(x_1 \mid y)p(x_2 \mid y)...p(x_n \mid y)$，对每个测试样本而言，无论 y 取值如何，x 是不变的，$p(x)$ 是一样的，因此目标函数为：

$$h(x)=\max_{y \in Y} p(y)\sum_{i=1}^{n} p(x_i \mid y)$$

令 D_y 表示样本集 D 中第 y 类样本组成的集合，若有充足的独立同分布样本，则可容易地估计出类 y 先验概率：$P(y)=\frac{|D_y|}{|D|}$

对离散属性而言，令 D_{y,x_i} 表示 D_y 在第 i 个属性上取值为 x_i 的样本组成的集合，则条件概率：

对连续属性可以考虑概率密度函数，假设 $p(x_i \mid y) \sim N(\mu_{y,i},\sigma_{y,i}^2)$，其中 $\mu_{y,i},\sigma_{y,i}^2$ 分别为第 y 类样本在第 i 在属性上取值的均值和方差，则有：

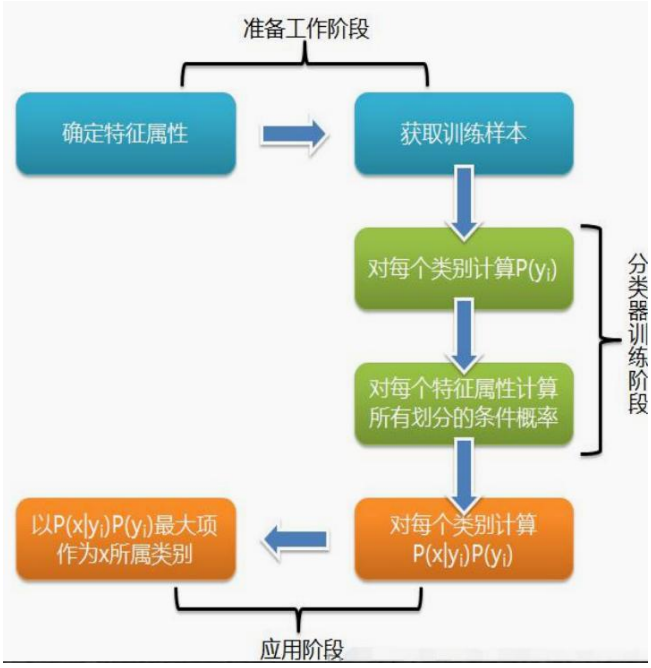
$$p(x_i \mid y) = \frac{1}{\sqrt{2\pi}\sigma_{y,i}} e^{-\frac{(x_i-\mu_{y,i})^2}{2\sigma_{y,i}^2}}$$

受样本个数限制，若某个属性值在训练集中没有与某个类同时出现过，则连乘公式必为零，其他属性取任意值都不能改变这一结论，因此需要用拉普拉斯修正：

$$P(y) = \frac{|D_y|+1}{|D|+N}, \quad p(x_i \mid y) = \frac{|D_{y,x_i}|+1}{|D_y|+N_i}$$

其中，N 表示数据集 D 中可能的类别数，N_i 表示第 i 个属性可能的取值数。

6.2 算法流程图



6.3 西瓜数据集举例

表 4.3 西瓜数据集 3.0

编号	色泽	根蒂	敲声	纹理	脐部	触感	密度	含糖率	好瓜
1	青绿	蜷缩	浊响	清晰	凹陷	硬滑	0.697	0.460	是
2	乌黑	蜷缩	沉闷	清晰	凹陷	硬滑	0.774	0.376	是
3	乌黑	蜷缩	浊响	清晰	凹陷	硬滑	0.634	0.264	是
4	青绿	蜷缩	沉闷	清晰	凹陷	硬滑	0.608	0.318	是
5	浅白	蜷缩	浊响	清晰	凹陷	硬滑	0.556	0.215	是
6	青绿	稍蜷	浊响	清晰	稍凹	软粘	0.403	0.237	是
7	乌黑	稍蜷	浊响	稍糊	稍凹	软粘	0.481	0.149	是
8	乌黑	稍蜷	浊响	清晰	稍凹	硬滑	0.437	0.211	是
9	乌黑	稍蜷	沉闷	稍糊	稍凹	硬滑	0.666	0.091	否
10	青绿	硬挺	清脆	清晰	平坦	软粘	0.243	0.267	否
11	浅白	硬挺	清脆	模糊	平坦	硬滑	0.245	0.057	否
12	浅白	蜷缩	浊响	模糊	平坦	软粘	0.343	0.099	否
13	青绿	稍蜷	浊响	稍糊	凹陷	硬滑	0.639	0.161	否
14	浅白	稍蜷	沉闷	稍糊	凹陷	硬滑	0.657	0.198	否
15	乌黑	稍蜷	浊响	清晰	稍凹	软粘	0.360	0.370	否
16	浅白	蜷缩	浊响	模糊	平坦	硬滑	0.593	0.042	否
17	青绿	蜷缩	沉闷	稍糊	稍凹	硬滑	0.719	0.103	否

（可以这样理解，拿一个西瓜，好瓜概率是 a，坏瓜概率是 b，在好瓜中色泽=青绿概率是 a1，好瓜中根蒂=蜷缩的概率是 b1，...，好瓜中含糖率=0.46 的概率是 c1，在坏瓜中色泽=青绿概率是 a2，坏瓜中根蒂=蜷缩的概率是 b2，...，坏瓜中含糖率=0.46 的概率是 c2，现在一个瓜色泽=青绿，...，问是好瓜的概率，坏瓜的概率？）

$$p(\text{好瓜}=\text{是})=\frac{8}{17}, p(\text{好瓜}=\text{否})=\frac{9}{17}$$

$$P_{\text{青绿}\mid\text{是}}=P(\text{色泽}=\text{青绿}\mid\text{好瓜}=\text{是})=3/8=0.37$$

$$P_{\text{青绿}\mid\text{否}}=P(\text{色泽}=\text{青绿}\mid\text{好瓜}=\text{否})=3/9=0.333$$

$$P_{\text{蜷缩}\mid\text{是}}=P(\text{根蒂}=\text{蜷缩}\mid\text{好瓜}=\text{是})=5/8=0.375$$

$$P_{\text{蜷缩}\mid\text{否}}=P(\text{根蒂}=\text{蜷缩}\mid\text{好瓜}=\text{否})=3/9=0.333$$

... ...

$$P_{\text{密度}:0.697\mid\text{是}}=P(\text{密度}=0.697\mid\text{好瓜}=\text{是})=\frac{1}{\sqrt{2\pi}\cdot 0.129}\exp\left(-\frac{(0.697-0.574)^2}{2\times 0.129^2}\right)=1.959$$

$$P_{\text{密度}:0.697\mid\text{否}}=P(\text{密度}=0.697\mid\text{好瓜}=\text{否})=\frac{1}{\sqrt{2\pi}\cdot 0.195}\exp\left(-\frac{(0.697-0.496)^2}{2\times 0.195^2}\right)=1.203$$

编号	色泽	根蒂	敲声	纹理	脐部	触感	密度	含糖率	好瓜
测1	青绿	蜷缩	浊响	清晰	凹陷	硬滑	0.697	0.46	?

$$h(\text{好瓜}=\text{是})=P(\text{好瓜}=\text{是})\times P_{\text{青绿}\mid\text{是}}\times P_{\text{蜷缩}\mid\text{是}}\times P_{\text{浊响}\mid\text{是}}\times P_{\text{清晰}\mid\text{是}}\times P_{\text{凹陷}\mid\text{是}}\times P_{\text{硬滑}\mid\text{是}}\times P_{\text{密度}:0.697\mid\text{是}}\times P_{\text{含糖}:0.460\mid\text{是}}=0.038$$

$$h(\text{好瓜}=\text{否})=P(\text{好瓜}=\text{否})\times P_{\text{青绿}\mid\text{否}}\times P_{\text{蜷缩}\mid\text{否}}\times P_{\text{浊响}\mid\text{否}}\times P_{\text{清晰}\mid\text{否}}\times P_{\text{凹陷}\mid\text{否}}\times P_{\text{硬滑}\mid\text{否}}\times P_{\text{密度}:0.697\mid\text{否}}\times P_{\text{含糖}:0.460\mid\text{否}}=6.80\times 10^{-5}$$

采用拉普拉斯修正后的结果：

$$\hat{P}(\text{好瓜}=\text{是})=\frac{8+1}{17+2}\approx 0.474,\quad \hat{P}(\text{好瓜}=\text{否})=\frac{9+1}{17+2}\approx 0.526$$

$$\hat{P}_{\text{青绿}\mid\text{是}}=\hat{P}(\text{色泽}=\text{青绿}\mid\text{好瓜}=\text{是})=\frac{3+1}{8+3}\approx 0.364$$

$$\hat{P}_{\text{青绿}\mid\text{否}}=\hat{P}(\text{色泽}=\text{青绿}\mid\text{好瓜}=\text{否})=\frac{3+1}{9+3}\approx 0.333$$

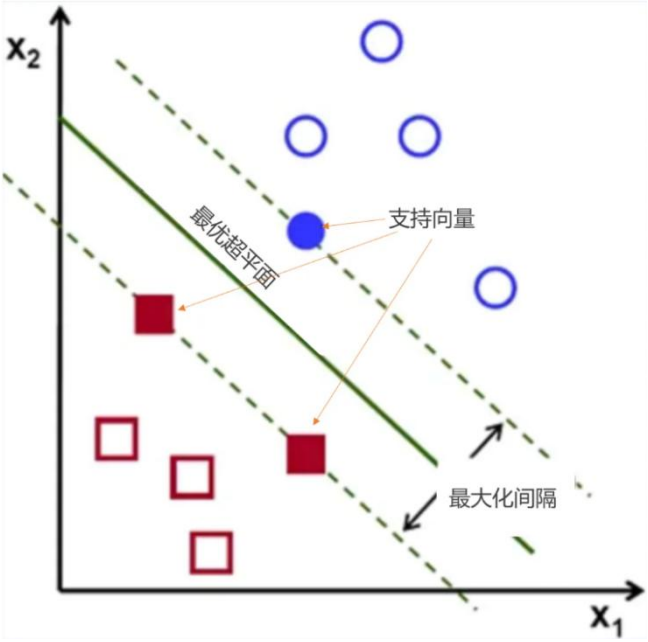
$$\hat{P}_{\text{清脆}\mid\text{是}}=\hat{P}(\text{敲声}=\text{清脆}\mid\text{好瓜}=\text{是})=\frac{0+1}{8+3}\approx 0.091$$

模型参数	解释	默认值
priors	类别的先验概率。如果指定，则不根据数据调整先验。	None
var_smoothing	为计算稳定性而添加到方差中的所有特征的最大方差部分。	1e-9

7 支持向量机模型【二分类】

7.1 原理介绍

SVM 通过求一个$n-1$维超平面（也叫决策边界）$\theta_0+\theta_1x^1+...+\theta_nx^n=0$，使得该超平面尽可能将样本区分开。一个重要概念是“支持向量”，它们是离超平面最近的数据点。SVM 的目标是找到这些支持向量，并确保它们之间的距离尽可能大（保证泛化性鲁棒性），这个距离称为“间隔”。



支持向量机的工作原理如下：

1. 数据转换：首先，SVM将数据点映射到高维空间，这样它们可以更容易地被一个超平面分开。

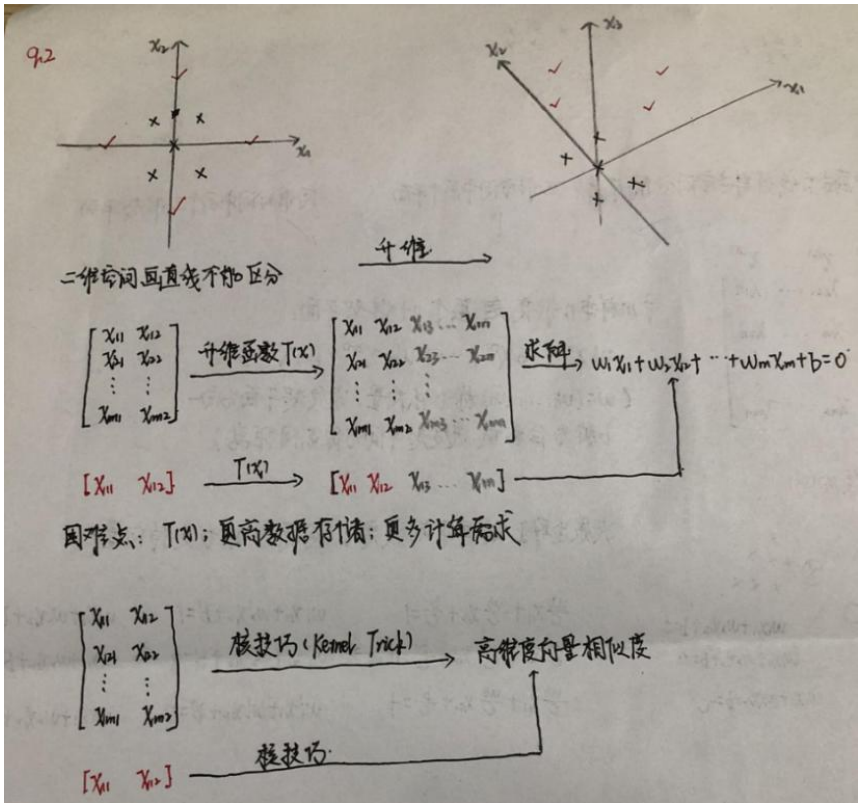
2. 超平面选择：然后，SVM尝试找到一个超平面，使得不同类别的支持向量离它最远。这个超平面的方程可以表示为：$w \cdot x + b = 0$，其中w是超平面的法向量，b是偏置。

3. 间隔最大化：SVM的目标是最大化支持向量到超平面的距离，这个距离称为“间隔”。间隔的计算公式是：$2/||w||$。

4. 分类：最后，SVM使用这个超平面来进行分类。对于新的数据点，它会根据这个超平面的位置来决定它属于哪个类别。

注意：对线性分类问题，可以直接使用 SVM 找到分割超平面；对于非线性分类问题，需要借助核函数来处理，核函数将数据映射到高维空间，以便在高维空间中找到一个线性超平面来分隔数据。常用的核函数包括线性核、多项式核和高斯核等。

7.1.1 使用核函数升维比单纯的升维好处



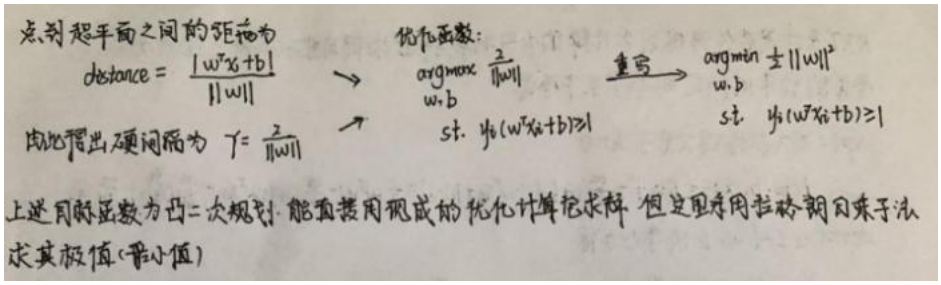
令 $\phi(x)$ 表示将 x 映射后的特征向量，则超平面为 $\theta^T\phi(x)=0$，

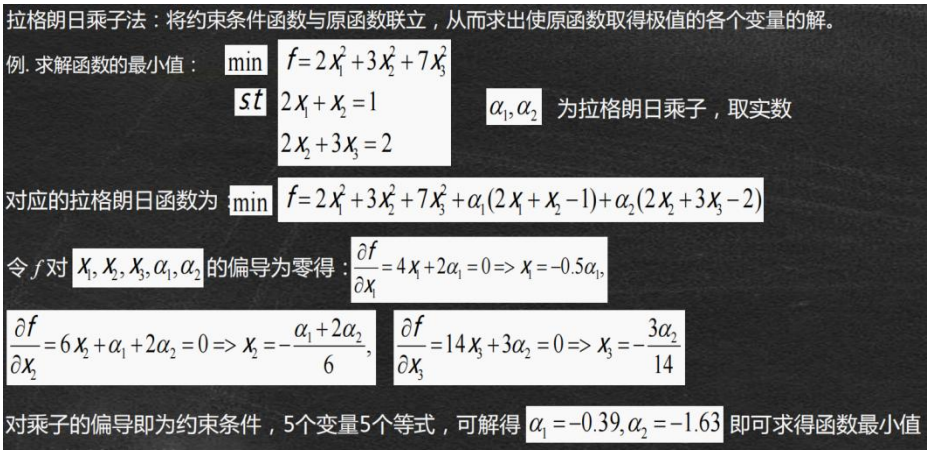
高维空间中的对偶问题为 $\max W(\alpha)=L(w,b,\alpha)=\sum_{i=1}^{m}\alpha_i-\frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j y_i y_j\phi(x_i)^T\phi(x_j)$

$\phi(x_i)^T\phi(x_j)$ 是样本 x_i,x_j 映射到高维空间后的内积，由于空间维数高，甚至是无穷维，直接计算是困难的，为了避开这个障碍，可以设想这样一个函数：$K(x_i,x_j)=\phi(x_i)^T\phi(x_j)$，即高维空间的内积等于原始空间中通过 K 计算的结果。K 即是核函数，其选择成为 SVM 中最大的变数。

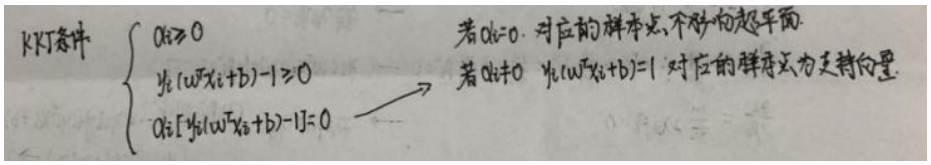
名称	表达式	参数
线性核	$\kappa(\boldsymbol{x}_i,\boldsymbol{x}_j)=\boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{x}_j$	
多项式核	$\kappa(\boldsymbol{x}_i,\boldsymbol{x}_j)=(\boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{x}_j)^d$	$d\geqslant 1$ 为多项式的次数
高斯核	$\kappa(\boldsymbol{x}_i,\boldsymbol{x}_j)=\exp\left(-\frac{\|\boldsymbol{x}_i-\boldsymbol{x}_j\|^2}{2\sigma^2}\right)$	$\sigma>0$ 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, $\beta>0,\theta<0$

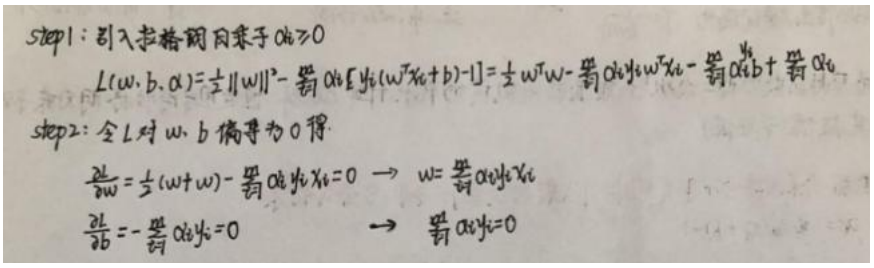
7.1.2 拉格朗日乘子法对凸二次规划求解



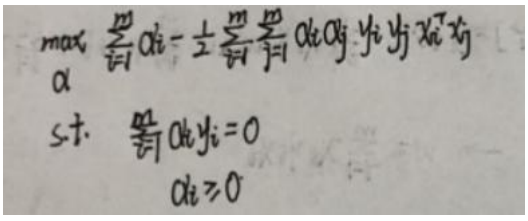


KKT 条件将拉格朗日乘子法所处理涉及等式的约束优化问题推广至不等式。

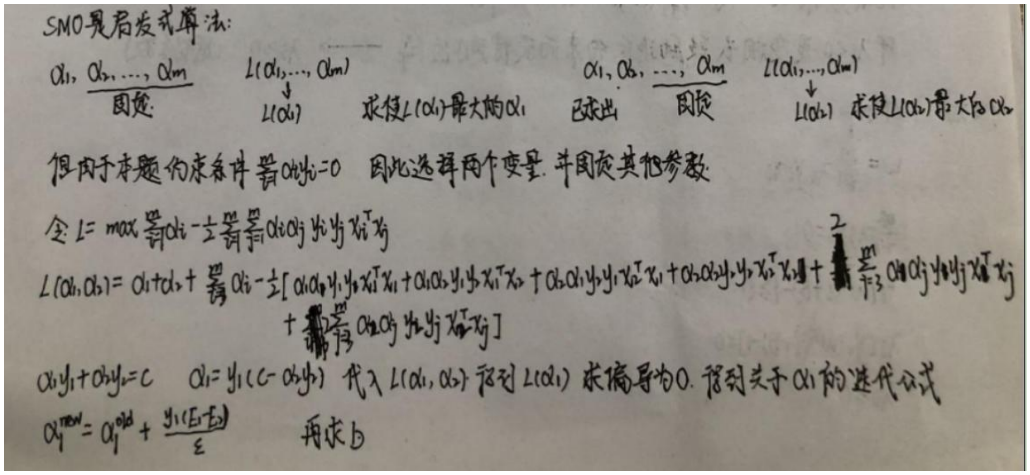




写出对偶问题：

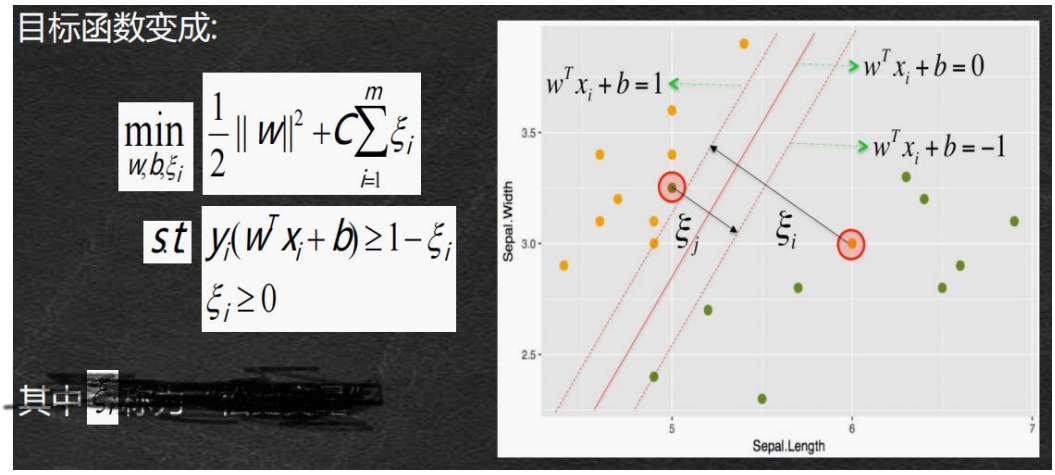


对偶问题本身为组合优化问题（改变任何一个 alpha 目标函数值均会变化），且随着训练样本数（m）增大，计算开销也会激增，所以需用更高效的算法。



7.1.3 软间隔处理

很难找到一个合适的核函数，将原始样本映射到高维空间后完全线性可分。有错误可以接受，但要量化出来，这样后面才知道怎么减少。



其中，C 表示对 ξ 的惩罚度，当其为 $+\infty$ 时，软间隔即是硬间隔。

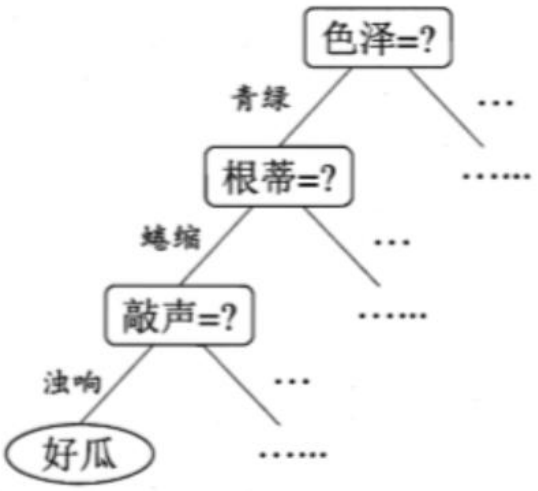
模型参数	解释	默认值
C	正则化参数，强度与 C 成反比，惩罚是平方 l2 惩罚	1
kernel{*'linear', 'poly', 'rbf', 'sigmoid', 'precomputed'*}	核函数类型	rbf
degree	多项式核函数的次数，必须是非负数，被所有其他内核忽略。	3
gamma{*'scale', 'auto'} or float*	"rbf"、"poly"和"sigmoid"的核系数,scale 表示 1 / (n_features * X.var())，auto 表示 1 / n_features	auto
probability	是否启用概率估计。这必须在调用 fit 之前启用，这会降低该方法的速度，因为它在内部使用 5 重交叉验证，并且 predict_proba 可能与 predict 不一致	False

random_state	随机种子	None

8 决策树模型

8.1 原理介绍

决策树（Decision tree）是一种非参数的有监督学习算法，非参数即对变量的类型不作要求，可以是连续变量也可以是分类变量。决策树通过一系列的特征和标签中总结出决策规则，用于解决分类问题和回归问题，最后 以树状的结构进行呈现。如下图所示：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，其他每个结点对应于一个属性测试。每个结点包含的样本集合根据属性测试的结果划分到子结点中，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判定的测试序列。决策树学习的目的是产生一棵泛化能力强，即处理未见示例强的决策树。使用决策树进行决策的过程就是从根节点开始，测试待分类项中相应的特征属性，并按照其值选择输出分支，直到到达叶子节点，将叶子节点存放的类别作为决策结果。



西瓜问题的一棵决策树

决策树的构建过程为：

（1）特征选择：选取有较强分类能力的特征。

（2）决策树生成：典型的算法有 CART，ID3 和 C4.5， 它们生成决策树过程相似，CART 是采用基尼指数作为特征选择度量，ID3 是采用信息增益，而 C4.5

采用信息增益比率。

（3）决策树剪枝：剪枝原因是决策树生成算法生成的树对训练数据的预测很准确， 但是对于未知数据分类很差， 这就产生了过拟合的现象。

8.1.1 特征选择

在决策树中，评价是否是最佳节点的方法是看节点的纯度（impurity），纯度越高说明这个节点中相同样本的占比越大，拟合效果越好。

（1）基尼指数(Gini index)【CART 决策树（1984，二叉树）采用该标准】

假设当前样本集合 D 中第 k 类样本所占的比例为 $p_k(k=1,2,...,n)$， n 为 D 中所含的类别个数，则 D 的基尼值定义为：

$$Gini(D)=1-\sum_{k=1}^{n}p_k^2$$

基尼值越小，则 D 的纯度越高。

假设特征 a 有 b 个不同的取值 $\{a^1,a^2,...,a^b\}$，用 a 对样本集 D 进行划分时会产生 b 个分支（或者说 b 个小集合）$\{D^1,D^2,...,D^b\}$，用特征 a 对样本集 D 划分后得到的基尼指数(Gini index)定义为：

$$Gini\_index(D,a)=\sum_{i=1}^{b}\frac{|D^i|}{|D|}Gini(D^i)$$

于是，在特征属性集合 A 中，选择那个使得划分后基尼指数最小的特征作为最优划分特征，即 $a_*=\arg\min_{a\in A} Gini\_index(D,a)$。

（2）信息增益(information gain)【ID3 决策树（1986，多叉树）采用该标准】

假设当前样本集合 D 中第 k 类样本所占的比例为 $p_k(k=1,2,...,n)$， n 为 D 中所含的类别个数，则 D 的信息熵(information entropy)定义为：

$$Ent(D)=-\sum_{\substack{k=1\\p_k\neq 0}}^{n}p_k\log_2(p_k)$$

信息熵越小，则 D 的纯度越高。

假设特征 a 有 b 个不同的取值 $\{a^1,a^2,...,a^b\}$，用 a 对样本集 D 进行划分时会产生 b 个分支（或者说 b 个小集合）$\{D^1,D^2,...,D^b\}$，用特征 a 对样本集 D 划分后得

到的信息增益定义为：

$$Gain(D,a)=Ent(D)-\sum_{i=1}^{b}\frac{|D^i|}{|D|}Ent(D^i)$$

一般而言，信息增益越大，则意味着使用特征 a 来进行划分所获得的“纯度提升”越大。于是，在特征属性集合 A 中，选择那个使得划分后信息增益最大的特征作为最优划分特征，即 $a_*=\arg\max_{a\in A} Gain(D,a)$ 。

（3）增益率(gain ratio)【C4.5 决策树（1993，多叉树）采用该标准】

如果将序号也作为特征，那么计算出来的信息增益远大于其他特征（1 个序号对应 1 个特征，全纯），实际上，信息增益对较多取值的特征有所偏好，为减少这种偏好可能带来的不利影响，使用增益率来选择最优划分特征：

$$Gain\_ratio(D,a)=\frac{Gain(D,a)}{IV(a)}$$

其中，$IV(a)=-\sum_{i=1}^{b}\frac{|D^i|}{|D|}\log_2\frac{|D^i|}{|D|}$，称为特征 a 的固有值，特征 a 的取值越多，$IV(a)$ 的值越大。

于是，在特征属性集合 A 中，选择那个使得划分后增益率最大的特征作为最优划分特征，即 $a_*=\arg\max_{a\in A} Gain\_ratio(D,a)$ 。

需要注意的是，增益率准则可能对较少取值的特征有所偏好，因此在实际中先使用一个启发式：先从候选特征中找出信息增益高于平均水平的特征，再从中选择增益率最高的。

8.1.2 剪枝

分支过多会造成训练样本学得“太好”，以至于把训练样本本身的一些特点当作所有数据都具有的一些性质而导致过拟合，因此需要剪枝。

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

后剪枝：先从训练集生成一棵完整的决策树，然后自底向上地对非叶节点进行考察，若将该节点对应的子树替换为叶子节点能带来决策树泛化性能提升，则将该子树替换为叶节点。

具体是将训练集预留一部分数据作为“验证集”来进行性能评估。

预剪枝基于贪心思想，禁止了某些分支的展开，欠拟合风险大，但时间开销小；有些分支当前划分虽不能提升泛化性能，但后续划分却可能提高，这个问题没有解决；

后剪枝保留更多分支，欠拟合风险小，但时间开销大。

8.1.3 连续与缺失值

（1）连续特征

上述讨论了基于离散特征来生成决策树，但现实学习任务中常用到连续特征，由于连续属性的可取值数目不再有限，因此不能直接根据连续特征的可取值来对节点进行划分，此时连续特征离散化技术可以排上用场，最简单的策略是采用二分法对连续属性进行处理，这是 C4.5 决策树算法中采用的机制。

给定样本集合 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 $\{a^1, a^2, ..., a^n\}$。基于划分点 t 可将 D 分为子集 D_t^- 和 D_t^+，其中 D_t^- 包含那些在特征 a 上取值不大于 t 的样本，而 D_t^+ 包含那些在特征 a 上取值大于 t 的样本。显然对相邻特征 a^i 与 a^{i+1} 来说，t 在区间 $[a^i, a^{i+1})$ 中取任意值所产生的划分结果相同，因此对连续特征 a 可以考察包含 $n-1$ 个元素的候选划分点集合

$$T_a = \{\frac{a^i + a^{i+1}}{2} | 1 \le i \le n-1\}$$

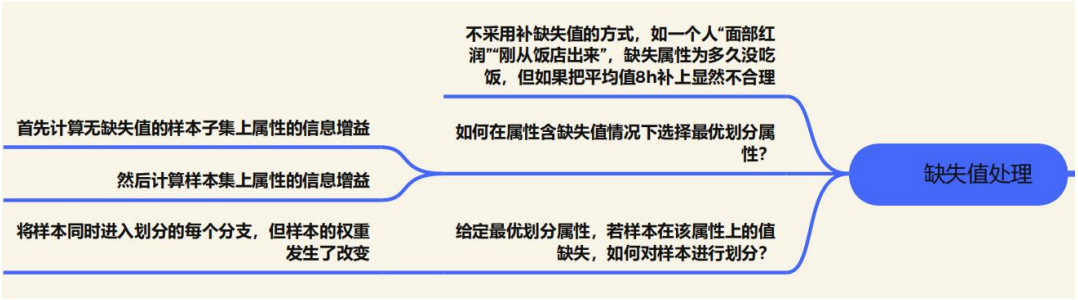
然后选取最优的划分点进行样本集合的划分，例如信息增益公式改进为：

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

注意：1 可将划分点设为该特征在训练集中出现的不大于中位点的最大值，从而使得最终决策树使用的划分点都在训练集中出现过。

2 与离散属性不同，若当前节点划分特征为连续特征，该特征还可以作为其后代节点的划分特征。如在父节点上使用“密度<=0.381”不会禁止在子节点上使用“密度<=0.294”。

（2）缺失特征



8.2 算法伪代码

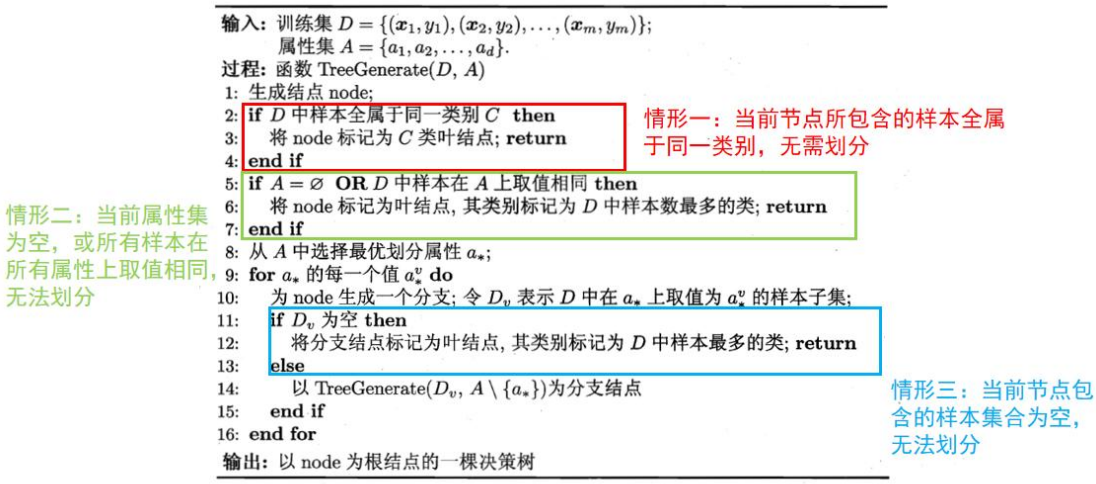


图 4.2 决策树学习基本算法

决策树的基本流程基本上可以分为四步：计算全部特征的不纯度，选出不纯度最低的特征来分支，在第一个特征的分支下，再次计算全部特征的不纯度，如此反复循环。当没有指标可用或者不纯度最优时，决策树停止增长。

决策树的生成是一个递归过程，在决策树基本算法中，有三种情形会导致递归返回：（1）当前节点所包含的样本全部属于同一类别，无需划分；（2）当前属性集为空，或是所有样本在所有属性上取值相同，无法划分；（3）当前节点包含的样本集合为空，不能划分。

8.3 西瓜数据集举例

表 4.3 西瓜数据集 3.0

编号	色泽	根蒂	敲声	纹理	脐部	触感	密度	含糖率	好瓜
1	青绿	蜷缩	浊响	清晰	凹陷	硬滑	0.697	0.460	是
2	乌黑	蜷缩	沉闷	清晰	凹陷	硬滑	0.774	0.376	是
3	乌黑	蜷缩	浊响	清晰	凹陷	硬滑	0.634	0.264	是
4	青绿	蜷缩	沉闷	清晰	凹陷	硬滑	0.608	0.318	是
5	浅白	蜷缩	浊响	清晰	凹陷	硬滑	0.556	0.215	是
6	青绿	稍蜷	浊响	清晰	稍凹	软粘	0.403	0.237	是
7	乌黑	稍蜷	浊响	稍糊	稍凹	软粘	0.481	0.149	是
8	乌黑	稍蜷	浊响	清晰	稍凹	硬滑	0.437	0.211	是
9	乌黑	稍蜷	沉闷	稍糊	稍凹	硬滑	0.666	0.091	否
10	青绿	硬挺	清脆	清晰	平坦	软粘	0.243	0.267	否
11	浅白	硬挺	清脆	模糊	平坦	硬滑	0.245	0.057	否
12	浅白	蜷缩	浊响	模糊	平坦	软粘	0.343	0.099	否
13	青绿	稍蜷	浊响	稍糊	凹陷	硬滑	0.639	0.161	否
14	浅白	稍蜷	沉闷	稍糊	凹陷	硬滑	0.657	0.198	否
15	乌黑	稍蜷	浊响	清晰	稍凹	软粘	0.360	0.370	否
16	浅白	蜷缩	浊响	模糊	平坦	硬滑	0.593	0.042	否
17	青绿	蜷缩	沉闷	稍糊	稍凹	硬滑	0.719	0.103	否

$$Ent(D) = -(\frac{8}{17}\log_2\frac{8}{17} + \frac{9}{17}\log_2\frac{9}{17}) = 0.998$$

$$Ent(D^1(\text{色泽} = \text{青绿})) = -(\frac{3}{6}\log_2\frac{3}{6} + \frac{3}{6}\log_2\frac{3}{6}) = 1$$

$$Ent(D^2(\text{色泽} = \text{乌黑})) = -(\frac{4}{6}\log_2\frac{4}{6} + \frac{2}{6}\log_2\frac{2}{6}) = 0.918$$

$$Ent(D^3(\text{色泽} = \text{浅白})) = -(\frac{1}{5}\log_2\frac{1}{5} + \frac{4}{5}\log_2\frac{4}{5}) = 0.722$$

$$Gain(D, \text{色泽}) = 0.998 - (\frac{6}{17}\times 1 + \frac{6}{17}\times 0.918 + \frac{5}{17}\times 0.722) = 0.109$$

类似的，可以计算出其他属性的信息增益：

$Gain(D, \text{根蒂}) = 0.143$， $Gain(D, \text{敲声}) = 0.141$， $Gain(D, \text{纹理}) = 0.381$，

$Gain(D, \text{脐部}) = 0.143$， $Gain(D, \text{触感}) = 0.006$，选择特征纹理。

之后，该结点包含的样例集合 D^1 中有编号为 {1,2,3,4,5,6,8,10,15} 的 9 个样例，可用属性集合为{色泽，根蒂，敲声，脐部，触感}，基于 D^1 计算出各特征的信息增益：

$Gain(D^1, \text{色泽}) = 0.043$， $Gain(D^1, \text{根蒂}) = 0.458$， $Gain(D^1, \text{敲声}) = 0.331$，

$Gain(D^1, \text{脐部}) = 0.458$， $Gain(D^1, \text{触感}) = 0.458$，任选一个。

8.4 参数调整

决策树的不稳定性：决策树对训练集的变动（旋转，减小）非常敏感，或许可以通过随机森林对许多树进行平均预测来限制这种不稳定性。

模型参数	解释	默认值
criterion*{"gini", "entropy", "log_loss"}*	衡量节点纯度的函数	gini
splitter*{"best", "random"}*	用于在每个节点选择拆分的策略。支持的策略是"最佳"选择最佳拆分和"随机"选择最佳随机拆分。	best
max_depth	树的最大深度。如果没有，则扩展节点，直到所有叶子都是纯的，或者直到所有叶子包含少于 min_samples_split 样本。	None
min_samples_split	分割内部节点所需的最小样本数	2
min_samples_leaf	叶节点上所需的最小样本数	1
max_leaf_nodes	最大叶子节点数量	None
min_impurity_decrease	用来限制信息增益参数,信息增益是父子节点信息熵的差值,若设置 impurtiy_decrease=0.2,则会在父子节点信息增益<0.2 时停止生长	0
random_state	随机种子	None

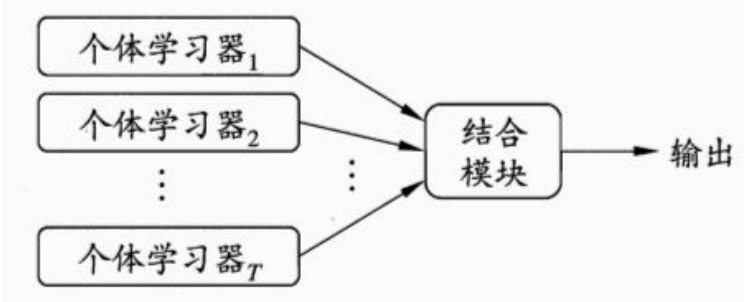
9 集成学习模型

9.1 原理介绍

9.1.1 集成学习的核心和分类

决策树有过拟合的风险，剪枝策略也只能缓解，无法避免。后来的研究者通过集成（ensemble）思想，开发了以随机森林（random forest）和梯度提升决策树 （gradient boosted decision tree）为代表的集成学习模型，这两者都是以决策树为基础的。

集成学习是指合并多个机器学习模型（或者说基学习器）来构建更强大模型的方法，应用越来越广泛。不只是决策树可以集成，很多不同类型的学习器都可以作为基础来进行集成，比如可以集成 logistic 回归和决策树，复杂点的，还可以集成神经网络与随机森林。



集成学习示意图

在一般经验中，如果把好坏不等的东西掺到一起，那么通常结果会是比最坏的要好一些，比最好的要坏一些。集成学习把多个学习器结合起来,如何能获得比最好的单一学习器更好的性能呢？

如下图所示，对号表示分类正确，叉号表示分类错误，这个简单的例子显示出：要获得好的集成，个体学习器应“好而不同”，即个体学习器要有一定的“准确性”，即学习器不能太坏，并且要有“多样性”(diversity),即学习器间具有差异。事实上，个体学习器的“准确性”和 “多样性”本身就存在冲突.一般的，准确性很高之后，要增加多样性就需牺牲准确性.事实上，**如何产生并结合“好而不同”的个体学习器，恰是集成学习研究的核心。**

	测试例1	测试例2	测试例3
h_1	√	√	×
h_2	×	√	√
h_3	√	×	√
集成	√	√	√

(a) 集成提升性能

	测试例1	测试例2	测试例3
h_1	√	√	×
h_2	√	√	×
h_3	√	√	×
集成	√	√	×

(b) 集成不起作用

	测试例1	测试例2	测试例3
h_1	√	×	×
h_2	×	√	×
h_3	×	×	√
集成	×	×	×

(c) 集成起负作用

图 8.2 集成个体应“好而不同”(h_i 表示第 i 个分类器)

根据个体学习器的生成方式，目前的集成学习方法大致可分为两大类，即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，以及个体学习器间不存在强依赖关系、可同时生成的并行化方法；前者的代表是 Boosting,后者的代表是 Bagging 和“随机森林" (Random Forest)。

9.1.2 Boosting【串行】组代表算法：AdaBoost，GBDT，XGB

Boosting 族算法的工作机制类似：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（调整其权重），然后基于调整后的样本分布来训练下一个基学习器；如此重复进行，直至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”实施，即在训练过程的每一轮中，根据样本分布为每个训练样本重新赋予一个权重。对无法接受带权样本的基学习算法，则可通过“重采样法”来处理，即在每一轮学习中，根据样本分布对训练集重新进行采样，再用重采样而得的样本集对基学习器进行训练。

需注意的是，Boosting 算法在训练的每一轮都要检查当前生成的基学习器是否满足基本条件（精度大于 0.5，检查当前基分类器是否是比随机猜测好）：

“重赋权法”一旦条件不满足，则当前基学习器即被抛弃,且学习过程停止.在此种情形下,初始设置的学习轮数 T 也许还远未达到，可能导致最终集成中只包含很少的基学习器而性能不佳.

“重采样法”，则可获得“重启动”机会以避免训练过程过早停止，即在抛弃不满足条件的当前基学习器之后，可根据当前分布重新对训练样本进行采样，再基于新的采样结果重新训练出基学习器，从而使得学习过程可以持续到预设的 T 轮完成。

（1）XGB（Extreme Gradient Boosting）

旨在提供高效、灵活和可扩展的梯度提升框架，在性能和准确性方面具有很大的优势，具备以下特色：

1. 基于梯度提升决策树（GBDT）：XGBoost 使用梯度提升算法来训练决策树模型，这是一种逐步迭代的方法，每次迭代都根据前一次迭代的结果来优化模型。通过串行训练决策树，每棵树都修正上一棵树的残差，从而逐步提升模型的准确性。

2. 正则化：XGBoost 提供了正则化技术来控制模型的复杂度，避免过拟合。正则化方法包括 L1 和 L2 正则化，使得模型能够更好地泛化到新的数据。

3. 特征工程的自动处理：XGBoost 能够自动处理缺失值和缺失特征，并找出最佳分割点。此外，它还能处理高维特征，通过自动学习特征的重要性和组合方式，能够生成更有代表性的特征。

4. 并行训练：XGBoost 具有良好的可扩展性，支持并行训练。它可以同时在多个核心上训练多棵树，大大缩短了训练时间。

5. 损失函数的灵活性：XGBoost 可以根据任务的不同选择不同的损失函数。例如，对于回归问题，可以使用平方损失函数；对于分类问题，可以使用逻辑损失函数。

6. 特征重要性评估：XGBoost 能够估计每个特征对于模型的重要性，并提供各种工具来可视化和分析特征的重要性。这对于特征选择和模型解释非常有用。

总体而言，XGBoost 凭借优化的性能、高准确性和可解释性，曾成为许多机器学习任务中的首选算法。它在各种数据集和问题上都表现出色，并且经过了广泛的应用和验证。

模型参数	解释	默认值
n_estimators	树的数量	gini
max_depth	树的深度	best
learning_rate	学习率控制每个回归树（boosting round）对最终预测结果的贡献程度。较小的学习率可以使模型更加稳定，但可能需要增加更多的回归树才能达到理想的性能。	None

9.1.3 Bagging【并行】组算法

工作机制：通过自助法采样出 T 个含 m 个训练样本的采样集，然后基于每个采样集训练出一个基学习器，再将这些基学习器进行**结合（怎么结合看后面）**。

9.1.4 随机森林【并行】

工作机制：以决策树为基学习器构建 Bagging 集成的基础上，进一步在决策树的训练过程中引入了随机属性选择。具体来说，传统决策树在选择划分属性时

是在当前结点的属性集合(假定 d 个属性)中选择一个最优属性；而在 RF 中，对基决策树的每个结点，**先从该结点的属性集合中随机选择一个包含 k 个属性的子集，然后再从这个子集中选择一个最优属性用于划分**。这里的参数 k 控制了随机性的引入程度：若令 k = d）则基决策树的构建传统决策树相同；若令 k = 1，则是随机选择一个属性用于划分；一般情况下，推荐值 k = log2d。

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，这就使得最终集成的泛化性能可通过个体学习器之间差异度的增加而进一步提升。

模型参数	解释	默认值
n_estimators	树的个数	100
criterion*{"gini", "entropy", "log_loss"}*	衡量节点纯度的函数	gini
max_depth	树的最大深度。如果没有，则扩展节点，直到所有叶子都是纯的，或者直到所有叶子包含少于 min_samples_split 样本。	None
min_samples_split	分割内部节点所需的最小样本数	2
min_samples_leaf	叶节点上所需的最小样本数	1
max_leaf_nodes	最大叶子节点数量	None
min_impurity_decrease	用来限制信息增益参数,信息增益是父子节点信息熵的差值,若设置 impurtiy_decrease=0.2,则会在父子节点信息增益<0.2 时停止生长	0
random_state	随机种子	None

9.2 基学习器结合策略

学习器结合可能会从三个方面带来好处：首先，从统计的方面来看，由于学习任务的假设空间往往很大，可能**有多个假设在训练集上达到同等性能**，此时若使用单学习器可能因误选而导致泛化性能不佳，结合多个学习器则会减小这一风险；第二，从计算的方面来看，**学习算法往往会陷入局部极小**，有的局部极小点所对应的泛化性能可能很糟糕，而通过多次运行之后进行结合，可降低陷入糟糕局部极小点的风险；第三，从表示的方面来看，某些学习任务的**真实假设可能不在当前学习算法所考虑的假设空间中**，此时若使用单学习器则肯定无效，而通过结合多个学习器，由于相应的假设空间有所扩大，有可能学得更好的近似。

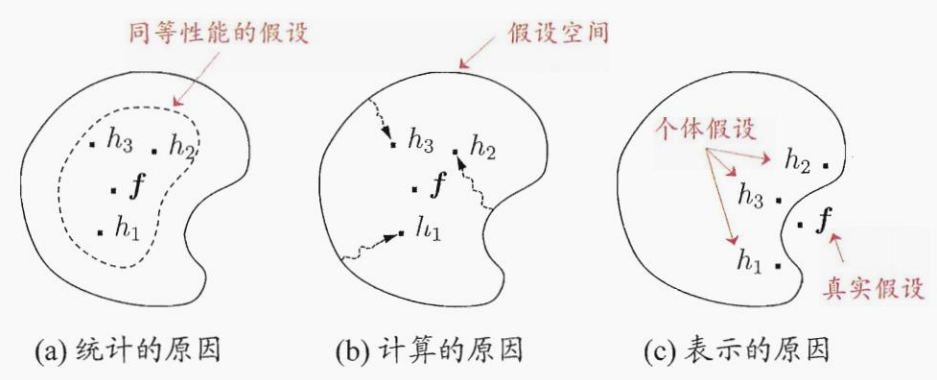


图 8.8 学习器结合可能从三个方面带来好处 [Dietterich, 2000]

（1）平均法

• 简单平均法

$$H(x)=\sum_{i=1}^{T}h_i(x)$$

其中，$h_i(x)$ 为第 i 个分类器对实例 x 的输出。

• 加权平均法

$$H(x)=\sum_{i=1}^{T}\omega_i h_i(x)$$

其中，$\omega_i \geq 0$ 且 $\sum_{i=1}^{T}\omega_i=1$，文献研究使用非负权重才能确保集成性能优于单一最佳个体学习器。

加权平均法的**权重一般是从训练数据中学习**而得，现实任务中的训练样本通常不充分或存在噪声，这将使得学出的权重不完全可靠。尤其是对规模比较大的集成来说，要学习的权重比较多，较容易导致过拟合。因此，实验和应用均显示出，加权平均法未必一定优于简单平均法。一般而言，在个体学习器性能相差较大时宜使用加权平均法，而在个体学习器性能相近时宜使用简单平均法。

（2）投票法

少数服从多数。

（3）Stacking 集成

当训练数据很多时，一种更为强大的结合策略是使用“学习法”，即通过另一个学习器来进行结合。这时候把以前的基学习器叫做初级学习器，把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，然后利用初级学习器**“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，而初始样本的标记仍被当作样例标记。

次级学习器的输入属性表示和次级学习算法对 Stacking 集成的泛化性能有很大影响。有研究表明，将初级学习器的输出类概率作为次级学习器的输入属性，用多响应线性回归（Multi-response Linear Regression,简称 MLR）作为次级学习算法效果较好。

10 对数几率回归

线性回归模型的向量化形式为：

$$\hat{y} = h_\theta(x) = \theta \cdot x$$

在此等式中：

- θ 是模型的参数向量，其中包含偏置项 θ_0 和特征权重 θ_1 至 θ_n。
- x 是实例的特征向量，包含从 $x^{(0)}$ 到 $x^{(n)}$， $x^{(0)}$ 始终等于 1。
- $\theta \cdot x$ 是向量 θ 和 x 的点积，它等于 $\theta_0 x^{(0)} + \theta_1 x^{(1)} + \theta_2 x^{(2)} + ... + \theta_n x^{(n)}$。
- h_θ 是假设函数，使用模型参数 θ。

对数几率回归的向量化形式为：

$$p = \sigma(y) = \frac{1}{1+e^{-y}}, y = \theta \cdot x$$

在此等式中：

- σ 为 sigmoid 函数，可以将任何实数映射到 0 和 1 之间。

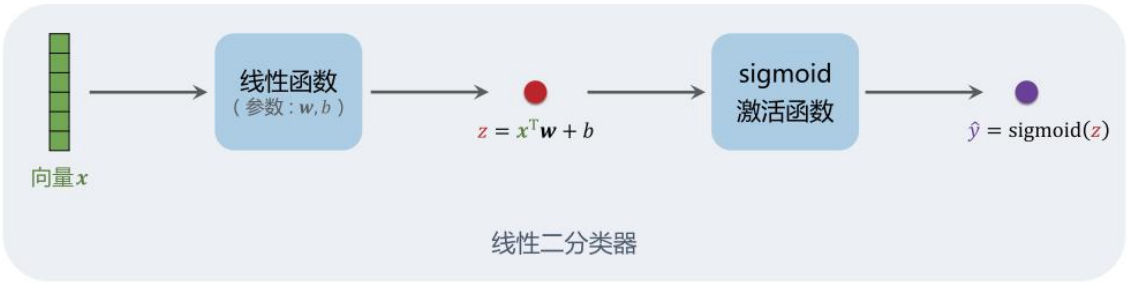


图 1-1 线性 sigmoid 分类器的结构. 输入是向量 $\boldsymbol{x} \in \mathbb{R}^d$，输出是介于 0 和 1 之间的标量

使用交叉熵作为模型的损失函数：

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}[y_i \ln(p_i) + (1-y_i)\ln(1-p_i)]$$

这个函数没有闭式解，经典的数值优化算法如梯度下降法，牛顿法等都可求得其最优解。

11 Softmax 回归

首先，给定一个示例 x，计算出每个类 k 的分数 $s_k(x)$

$$s_k(x) = x^T \theta^{(k)}$$

每个类都有属于自己的 $\theta^{(k)}$，（逻辑回归共用 1 个 θ，得到分数后，再用 sigmoid 函数得到概率）所有这些向量通常作为行存储在参数矩阵 Θ 中。

然后，对每个类 k 的分数应用 softmax 函数，求出实例 x 属于类 k 的概率：

$$p_k = \sigma(s_k(x)) = \frac{e^{s_k(x)}}{\sum_{j=1}^{K} e^{s_j(x)}}$$

在此等式中：

- K 是类数。
- $s(x)$ 是一个向量，其中包含实例 x 的每个类的分数。
- $\sigma(s_k(x))$ 是实例 x 属于类 k 的估计概率，给定该实例每个类的分数。

就像逻辑回归分类器一样，softmax 回归分类器预测具有最高估计概率的类。

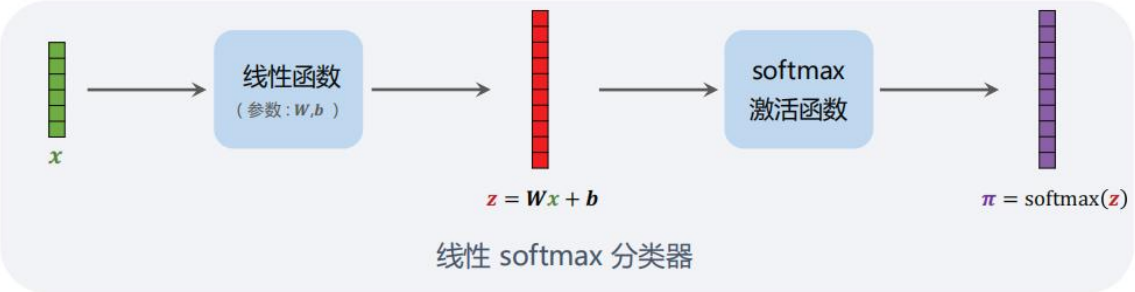


图 1-6　线性 softmax 分类器的结构．输入是向量 $\boldsymbol{x} \in \mathbb{R}^d$，输出是 $\boldsymbol{\pi} \in \mathbb{R}^k$

使用交叉熵作为模型的损失函数：

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}\sum_{k=1}^{K} y_k^{(i)} \ln(p_k^i)$$

这个损失函数是有道理的，如果示例 i 属于类 k，当它预测为类 k 时损失接近 0，当预测不属于类 k 时损失接近无穷。

这个函数没有闭式解，经典的数值优化算法如梯度下降法，牛顿法等都可求得其最优解。