



Multivariate Statistics and Machine Learning

目录

Multivariate Statistics and Machine Learning

EM算法

完全数据的对数似然函数 (Complete Data Log-Likelihood)

观测数据的对数似然函数 (Observed Data Log-Likelihood)

EM算法的关键思想

E步 (Expectation)

M步 (Maximization)

EM计算流程

初始化 (Initialisation)

E步 (期望步骤 in 第 b 次迭代)

M步 (最大化步骤 in 第 b 次迭代)

更新分配给第 k 类的样本数量 (Number of samples assigned to class k)

更新类别的先验概率 (Updated estimates of class probabilities)

更新类别的均值 (Updated estimates of group mean)

更新类别的协方差矩阵 (Updated estimates of group covariance)

K-means 与 GMM 的关系 - E 步

将软分配转为硬分配

K-means 与 GMM 的关系 - M 步

硬分配下的更新公式

例子：咖啡店顾客的分类

收敛性和不变状态 (invariant states)

收敛性

不变状态 (Invariant States)

具体公式

Supervised Learning

监督学习(Supervised Learning)的定义

监督学习的目标

关键术语

例子：糖尿病风险预测

Bayes Classifier贝叶斯分类器

贝叶斯分类器的基本概念

软分类 (Soft Classification)

Bayesian discriminant rule / Bayes classifier

后验概率 $P(y = k|x)$

硬分类 (Hard Classification)

贝叶斯判别函数 $d_k(x)$

判别函数与后验概率的关系

Normal Bayes Classifier

Quadratic Discriminant Analysis

判别函数 (Discriminant Function)

决策边界

Linear Discriminant Analysis (LDA)

判别函数

判别函数的进一步简化

线性判别和决策边界

Diagonal Discriminant Analysis (DDA)

判别函数

 EM算法

先来回顾一下log-likelihood functions for a GMM:

 完全数据的对数似然函数 (Complete Data Log-Likelihood)

首先，假设我们有数据 \mathbf{X} 和隐藏的变量 \mathbf{y} ，则完全数据的对数似然函数定义为：

$$\ell(\theta|\mathbf{X}, \mathbf{y}) = \sum_{i=1}^n \log f(x_i, y_i|\theta) = \sum_{i=1}^n \log \pi_{y_i} N_d(x_i|\mu_{y_i}, \Sigma_{y_i})$$

这个公式表示我们在已知每个数据点 x_i 的类别 y_i 时，可以直接计算的似然函数。这里 π_{y_i} 表示类别 y_i 的先验概率，而 $N_d(x_i|\mu_{y_i}, \Sigma_{y_i})$ 表示该类别下数据的高斯分布。因为我们知道每个数据点的类别，这个对数似然函数比较容易最大化。

 观测数据的对数似然函数 (Observed Data Log-Likelihood)

在实际情况中，我们并不知道每个数据点的类别 y_i 。因此，观测数据的对数似然函数为：

$$\ell(\theta|\mathbf{X}) = \sum_{i=1}^n \log f(x_i|\theta) = \sum_{i=1}^n \log \left(\sum_{k=1}^K \pi_k N_d(x_i|\mu_k, \Sigma_k) \right)$$

在这里， π_k 表示第 k 类的先验概率， $N_d(x_i|\mu_k, \Sigma_k)$ 是数据点 x_i 在第 k 类下的高斯分布。因为我们不知道每个数据点的类别，所以需要对所有可能的类别进行加权求和。这种形式的对数似然函数无法直接简化，使得最大化变得困难。

 EM算法的关键思想

EM算法的核心思想是利用已知的完全数据对数似然，通过“推断”每个数据点的类别来简化问题。EM算法包含两个步骤：期望步骤（E步）和最大化步骤（M步），通过这两个步骤的交替迭代来优化参数。

 E步 (Expectation)

在E步中，我们为每个数据点分配“软”类别。具体来说，我们计算数据点属于每个类别的概率：

$$\hat{q}_i(k) = P(y_i = k|x_i, \hat{\theta})$$

这里， $\hat{q}_i(k)$ 表示在当前参数估计 $\hat{\theta}$ 下，数据点 x_i 属于第 k 类的概率。这一步是计算期望完成数据的对数似然：

$$E_{P(y|\mathbf{X}, \hat{\theta})} (\ell(\theta|\mathbf{X}, \mathbf{y}))$$

 M步 (Maximization)

在M步中，我们最大化上一步中的期望对数似然函数，从而更新参数 θ 的估计值。这一步的目标是寻找一组新的参数，使得期望对数似然最大化。

通过交替执行E步和M步，EM算法能够逐步优化参数估计，直到收敛。

 EM计算流程

接下来我们讲一下这两步骤的具体计算流程：

初始化 (Initialisation)

在 EM 算法的开始，我们需要初始化参数 $\hat{\theta}^{(0)}$ ，或者为每个样本赋予一个初始的“软”类别分配，这些分配值通常存放在一个矩阵 $\mathbf{Q}^{(1)}$ 中。

E步 (期望步骤 in 第 b 次迭代)

在 E 步中，我们需要更新每个样本属于每个类别的概率。更新公式如下：

$$\hat{q}_i(k)^{(b+1)} = \frac{\hat{\pi}_k^{(b)} f_k(x_i | \hat{\theta}^{(b)})}{\sum_{l=1}^K \hat{\pi}_l^{(b)} f_l(x_i | \hat{\theta}^{(b)})}$$

对于 GMM，这个公式可以进一步写成：

$$\hat{q}_i(k)^{(b+1)} = \frac{\hat{\pi}_k^{(b)} N_d(x_i | \hat{\mu}_k^{(b)}, \hat{\Sigma}_k^{(b)})}{\sum_{l=1}^K \hat{\pi}_l^{(b)} N_d(x_i | \hat{\mu}_l^{(b)}, \hat{\Sigma}_l^{(b)})}$$

其中 $\hat{\pi}_k^{(b)}$ 是第 k 类的先验概率估计值， $N_d(x_i | \hat{\mu}_k^{(b)}, \hat{\Sigma}_k^{(b)})$ 是第 k 类的高斯分布概率密度函数。

在此，我们还需要计算“期望的完全数据对数似然”：

$$G(\theta | \mathbf{X}, \mathbf{Q}^{(b+1)}) = \sum_{i=1}^n \sum_{k=1}^K \hat{q}_i(k)^{(b+1)} \log(\pi_k f_k(x_i))$$

其中

$$\log(\pi_k f_k(x_i)) = -\frac{1}{2}(x_i - \mu_k)^T \Sigma_k^{-1} (x_i - \mu_k) - \frac{1}{2} \log \det(\Sigma_k) + \log(\pi_k)$$

M步 (最大化步骤 in 第 b 次迭代)

在 M 步中，我们需要最大化上述期望对数似然函数，以更新参数 θ ：

$$\hat{\theta}^{(b+1)} = \arg \max_{\theta} G(\theta | \mathbf{X}, \mathbf{Q}^{(b+1)})$$

那么接下来讲一下我们在 M 步中怎么来更新每个类别的参数。

更新分配给第 k 类的样本数量 (Number of samples assigned to class k)

公式如下：

$$n_k^{(b+1)} = \sum_{i=1}^n \hat{q}_i(k)^{(b+1)}$$

这里 $n_k^{(b+1)}$ 表示在第 $(b+1)$ 次迭代中分配给第 k 类的“有效样本数量”。这个数量并不是指具体分配到第 k 类的样本个数，而是一个“软分配”的结果。通过计算每个样本属于第 k 类的概率 $\hat{q}_i(k)^{(b+1)}$ 的总和，我们得到了一个“加权”的样本数量。

可以这样理解：每个样本 x_i 都有一个属于类别 k 的概率值 $\hat{q}_i(k)^{(b+1)}$ 。我们将所有样本的这些概率值相加，就得到了该类别的“有效样本数量”。

更新类别的先验概率 (Updated estimates of class probabilities)

公式如下：

$$\hat{\pi}_k^{(b+1)} = \frac{n_k^{(b+1)}}{n}$$

这里 $\hat{\pi}_k^{(b+1)}$ 表示在第 $(b+1)$ 次迭代中第 k 类的先验概率。这个先验概率是通过将第 k 类的有效样本数量 $n_k^{(b+1)}$ 除以总样本数 n 得到的。这样做的目的是估计每一类在数据中的比例。假设 n 是固定的，那么随着迭代进行， $\hat{\pi}_k^{(b+1)}$ 会逐渐收敛到一个稳定值。

好的，我们接着来看接下来的两个更新步骤。

更新类别的均值 (Updated estimates of group mean)

公式如下：

$$\hat{\mu}_k^{(b+1)} = \frac{1}{n_k^{(b+1)}} \sum_{i=1}^n \hat{q}_i(k)^{(b+1)} x_i$$

这里 $\hat{\mu}_k^{(b+1)}$ 表示在第 $(b+1)$ 次迭代中第 k 类的均值估计。这个公式的含义是通过加权平均计算每个类别的中心位置。

具体来说， $\hat{q}_i(k)^{(b+1)}$ 是样本 x_i 属于第 k 类的概率，所以 $\hat{q}_i(k)^{(b+1)}x_i$ 就是将 x_i 的值“分配”到第 k 类。我们将所有样本的这些“分配值”加起来，再除以该类的有效样本数 $n_k^{(b+1)}$ ，就得到了第 k 类的加权均值。

可以将这个过程类比为计算一个带有权重的平均值，每个样本对均值的贡献大小取决于它属于该类的概率。

更新类别的协方差矩阵 (Updated estimates of group covariance)

公式如下：

$$\hat{\Sigma}_k^{(b+1)} = \frac{1}{n_k^{(b+1)}} \sum_{i=1}^n \hat{q}_i(k)^{(b+1)} (x_i - \hat{\mu}_k^{(b+1)}) (x_i - \hat{\mu}_k^{(b+1)})^T$$

这里 $\hat{\Sigma}_k^{(b+1)}$ 是第 $(b+1)$ 次迭代中第 k 类的协方差矩阵。这个公式表示每个样本相对于该类别均值 $\hat{\mu}_k^{(b+1)}$ 的偏差加权平方和。

解释如下：

- $(x_i - \hat{\mu}_k^{(b+1)})$ 表示样本 x_i 与类别 k 的均值的差异。
- 这个差异的外积 $(x_i - \hat{\mu}_k^{(b+1)}) (x_i - \hat{\mu}_k^{(b+1)})^T$ 反映了样本在各个维度上的变异性。
- 再用属于第 k 类的概率 $\hat{q}_i(k)^{(b+1)}$ 进行加权，得到了每个样本对协方差的贡献。
- 最后除以该类的有效样本数 $n_k^{(b+1)}$ ，得到了加权后的协方差矩阵。

这个协方差矩阵反映了该类别在不同维度上的分布形状和分布范围。

 总结一下：

- 我们通过 $n_k^{(b+1)}$ 表示第 k 类的“有效样本数量”。
- 然后，用该类的样本加权平均计算均值 $\hat{\mu}_k^{(b+1)}$ 和协方差 $\hat{\Sigma}_k^{(b+1)}$ 。

K-means 与 GMM 的关系 - E 步

在这里，我们假设一个简化模型，其中：

- 所有类别的先验概率 π_k 是相等的，即 $\pi_k = \frac{1}{K}$ 。
- 所有类别的协方差矩阵 Σ_k 都相同，并且是一个比例缩放的单位矩阵 $\sigma^2 I_d$ ，即 $\Sigma_k = \sigma^2 I_d$ 。

基于这个简化的假设，我们在 E 步中计算样本 x_i 属于第 k 类的软分配 $\hat{q}_i(k)$ 公式可以写成：

$$\hat{q}_i(k) = C \exp \left(-\frac{1}{2\sigma^2} (x_i - \hat{\mu}_k)^T (x_i - \hat{\mu}_k) \right)$$

其中 C 是一个常数，与类别 k 无关。这表明 $\hat{q}_i(k)$ 是一个与距离成反比的指数函数。距离越小， $\hat{q}_i(k)$ 值越大，说明 x_i 越可能属于第 k 类。

将软分配转为硬分配

在 K-means 中，我们使用硬分配，将每个样本 x_i 分配给距离最近的类别中心。硬分配可以通过以下公式表示：

$$\hat{y}_i = \arg \max_k \hat{q}_i(k) = \arg \min_k (x_i - \hat{\mu}_k)^T (x_i - \hat{\mu}_k)$$

这个过程表明，我们在 EM 算法的 E 步中使用“软”分配（概率）来表征一个样本属于某类的可能性，而在 K-means 中，直接选择最可能的类别来分配样本，从而实现“硬”分配。

在这个简化的情况下，EM 算法的 E 步和 K-means 的硬分配过程实际上是等价的。我们可以认为 K-means 是 EM 算法在一个受限的高斯混合模型（即固定均值相同协方差）下的特殊情况。

K-means 与 GMM 的关系 - M 步

在简化的 GMM 模型下，我们只需要更新每个类别的均值 μ_k 。具体公式如下：

$$\hat{\mu}_k = \frac{1}{n_k} \sum_{i=1}^n \hat{q}_i(k) x_i$$

这里， $\hat{q}_i(k)$ 是“软”分配概率，表示样本 x_i 属于类别 k 的概率， n_k 是类别 k 的有效样本数。

硬分配下的更新公式

在 K-means 中，我们使用硬分配，将样本完全归属于某个类别。硬分配下，更新公式变为：

$$\hat{\mu}_k = \frac{1}{n_k} \sum_{i=1}^n I_{\{\hat{y}_i=k\}} x_i = \frac{1}{n_k} \sum_{i \in G_k} x_i$$

其中：

- $I_{\{\hat{y}_i=k\}}$ 是指示函数，表示样本 x_i 是否被分配到类别 k 。
- $G_k = \{i | \hat{y}_i = k\}$ 表示所有被分配到类别 k 的样本集合。

这个公式表明， $\hat{\mu}_k$ 是类别 k 的质心（centroid），即所有属于该类别样本的平均值。

因此，K-means 可以看作是一种特殊的 EM 算法，它基于一个简化的高斯混合模型，使用硬分配来进行更新。在这种情况下，K-means 的质心更新步骤与 EM 的 M 步中的均值更新等价。

例子：咖啡店顾客的分类

假设你是一家咖啡店的老板，你的顾客分为两类：

1. 常客：经常光顾咖啡店的顾客，他们通常点单较快，花费也比较高。
2. 路人：偶尔进来消费的顾客，他们花费较低，逗留时间也短。

不过，你并不能直接知道每位顾客是常客还是路人。于是你决定根据一些数据来“推测”他们属于哪个类别，比如每位顾客的消费金额和逗留时间。

数据收集

我们收集到了一些数据，每条记录包含两个信息：

- 消费金额
- 逗留时间

例如，数据可能如下所示（每行代表一位顾客）：

顾客数据 = {(消费金额, 逗留时间)} = {(20, 30), (5, 10), (25, 35), (6, 8), (21, 28), (7, 9)}

这些数据中包含了两类顾客的信息，但我们无法直接知道每位顾客属于哪个类别。

用 EM 算法来推测顾客类别

1. 初始化

我们随机猜测两个“类别中心”（均值），比如常客的均值是（消费金额 = 22, 逗留时间 = 32），路人的均值是（消费金额 = 6, 逗留时间 = 9）。

我们还假设两类顾客的比例相等，即 $\pi_1 = \pi_2 = 0.5$ 。

2. E步：计算每位顾客属于每个类别的概率

现在，假设顾客 (20, 30) 属于常客和路人的概率分别为 \hat{q}_1 （常客）和 \hat{q}_2 （路人）。这个概率可以通过计算顾客数据点到每个类别中心的“距离”来得到。通常来说：

如果顾客(20, 30)的消费金额和逗留时间接近常客的均值(22, 32), 它更有可能是常客。

如果顾客(5, 10)的数据点更接近路人均值(6, 9), 那么它更有可能是路人。

根据这种计算, 我们可以得到每位顾客属于常客和路人的“软分配”概率。比如:

顾客数据	$\hat{q}_i(\text{常客})$	$\hat{q}_i(\text{路人})$
(20, 30)	0.8	0.2
(5, 10)	0.1	0.9
(25, 35)	0.85	0.15
(6, 8)	0.2	0.8
(21, 28)	0.75	0.25
(7, 9)	0.15	0.85

这些概率告诉我们每个顾客属于常客和路人的可能性。这里用“软分配”来表示, 不是硬性地将顾客分为某一类, 而是让他们有一定的概率属于某类。

3. M步: 更新每类的均值和比例

有了每个顾客属于每类的概率, 我们可以重新计算两类顾客的均值和比例。

更新常客的均值: 我们使用这些“软分配”概率来加权每个顾客的数据。例如, 常客的均值计算为:

$$\hat{\mu}_{\text{常客}} = \frac{1}{\text{常客的有效样本数量}} \sum_{\text{所有顾客}} \hat{q}_i(\text{常客}) \times (\text{消费金额}, \text{逗留时间})$$

这样得到的新均值会更接近“软分配”更高的顾客数据。

更新路人的均值: 用类似的方式计算路人的均值。

更新先验概率: 我们还可以通过每类的有效样本数量计算出新的 $\pi_{\text{常客}}$ 和 $\pi_{\text{路人}}$, 这反映了常客和路人在顾客中的比例。

4. 重复 E 步和 M 步

我们反复进行 E 步和 M 步的计算, 直到类别均值和先验概率收敛到稳定值。最终, 我们可以根据收敛后的均值和概率, 得到每个顾客属于常客和路人的概率。

💡 K-means 和 EM 的对比

💡 K-means 的方法: K-means 会直接将每位顾客分到距离最近的类别, 即每位顾客直接被硬性分为常客或路人。

💡 EM 的方法: EM 则允许顾客有一定的概率属于两类, 从而生成“软分配”。最终, 我们可以根据这些概率来判断每位顾客是常客还是路人。

通过这个例子, 我们可以看到 EM 算法是如何在不确定的情况下推测顾客的类别, 而 K-means 则是一个简单的硬分配方法。EM 更加灵活, 允许每个样本拥有多个类别的概率, 从而处理一些不确定性。

⚠ 收敛性和不变状态 (invariant states)

收敛性

EM 算法的一个关键特性是每次迭代都会增加观测数据的对数似然, 即:

$$\ell(\hat{\theta}^{(b+1)} | \mathbf{X}) \geq \ell(\hat{\theta}^{(b)} | \mathbf{X})$$

这意味着, 随着迭代次数的增加, 对数似然值会逐步上升, 直到达到一个局部最优值。在适当的假设下, EM 算法保证收敛到观测数据对数似然的一个局部最优解。

但是，EM 算法依赖于初始参数的选择。换句话说，不同的初始化可能导致不同的局部最优解。由于 EM 算法的收敛性并不保证全局最优解，它有可能被困在次优的解中。

💡 不变状态 (Invariant States)

在某些情况下，如果初始化选择不当，EM 算法可能会陷入一个“不变状态”。这意味着随着迭代的进行，参数更新不再发生变化，对数似然也不再增加，这样算法就处在某种平衡状态。

💡 具体公式

假设我们初始化时对每一类分配了一个均匀的“软分配”值：

$$\hat{q}_i(k) = \frac{1}{K}$$

这种初始化会导致以下结果：

1. 类大小 (Class Size) :

$$n_k = \sum_{i=1}^n \hat{q}_i(k) = \frac{n}{K}$$

每类的有效样本数都相等。

2. 类的先验概率 (Class Probabilities) :

$$\hat{\pi}_k = \frac{n_k}{n} = \frac{1}{K}$$

每一类的先验概率均为 $\frac{1}{K}$ 。

3. 组均值 (Group Mean) :

$$\hat{\mu}_k = \frac{1}{n_k} \sum_{i=1}^n \hat{q}_i(k) x_i = \bar{x}$$

这里 \bar{x} 是所有数据点的均值，意味着所有类的均值会趋于相同。

4. 组协方差 (Group Covariance) :

$$\hat{\Sigma}_k = \frac{1}{n_k} \sum_{i=1}^n \hat{q}_i(k) (x_i - \hat{\mu}_k)(x_i - \hat{\mu}_k)^T = \hat{\Sigma}$$

所有类的协方差矩阵都相同。

5. 更新软分配:

由于参数没有显著差异，更新的软分配会继续保持 $\frac{1}{K}$ ，使得 EM 算法陷入平衡，无法跳出。

这种不变状态的问题表明，EM 算法在某些初始化下可能无法有效工作。为了解决这个问题，通常会尝试多次随机初始化或使用某种启发式方法来选择初始参数，从而增加找到全局最优解的概率。



Supervised Learning

💡 监督学习(Supervised Learning)的定义

监督学习是一种从有标签的数据中学习的机器学习方法，旨在预测或分类新的数据。

可以把监督学习理解为我们教机器如何做事情

其主要是由以下数据构成：

1. 带标签的训练数据 (Labelled Training Data) :

训练数据由样本和相应的标签组成，形式为 $\{(x_1^{\text{train}}, y_1^{\text{train}}), (x_2^{\text{train}}, y_2^{\text{train}}), \dots, (x_n^{\text{train}}, y_n^{\text{train}})\}$ 。

其中， x_i^{train} 表示第 i 个样本的特征， y_i^{train} 是对应的标签。

2. 未标记的测试数据 (Unlabelled Test Data) :

- 我们希望模型能够对测试数据 x^{test} 做出预测，即找到测试数据的对应标签。

监督学习的目标

监督学习的目的是通过训练数据学习一个预测函数 $h(x)$ ，用来预测新的测试数据的标签。具体来说，我们希望通过训练数据得到函数 h ，然后对测试样本 x^{test} 做出预测：

$$\hat{y}^{\text{test}} = h(x^{\text{test}})$$

根据标签 y 的类型，监督学习可以分为两种情况：

1. 分类 (Classification) :

- 如果标签 y 是离散的（例如类别标签“猫”、“狗”等），监督学习称为分类。
- 分类的任务就是将样本划分到不同的类别中。

2. 回归 (Regression) :

- 如果标签 y 是连续的（例如温度、房价等数值），监督学习称为回归。
- 回归的任务是预测连续数值，如房价预测、温度预测等。

举个🌰 子

假设我们想开发一个系统来判断天气是否适合外出钓鱼。我们可以收集历史天气数据（例如温度、湿度、风速等）和相应的标签（“适合钓鱼”或“不适合钓鱼”）。这些数据组成了我们的训练数据。

在有新的天气数据到来时（没有标签），系统就可以使用已经学到的模型 $h(x)$ ，预测当天是否适合钓鱼，这就是分类问题。

如果我们想预测具体的钓鱼收获量（例如预计会钓多少鱼），那么这就是一个回归问题。

关键术语

1. 分类器 (Classifier) :

- 分类器是一个函数 $h(x)$ ，它根据输入的特征 x 来预测输出的类别 y 。换句话说， $h(x)$ 就是我们要学习的模型，它能够告诉我们给定数据点属于哪个类别。

2. 判别函数 (Discriminant Function) :

- 判别函数 $d_k(x)$ 是分类器用来决定类别的一个函数。对于给定的样本 x ，分类器通过比较不同类别的判别函数值来做出分类决策。例如，假设我们有多个类别，分类器会选择使 $d_k(x)$ 最大的类别：

$$h(x) = \arg \max_k d_k(x)$$

3. 决策边界 (Decision Boundary) :

- 决策边界是样本空间中将不同类别分隔开来的边界。换句话说，决策边界是 $h(x)$ 改变类别预测的那些点的集合。

- 比如，在图中，红色的线就是决策边界，表示在该边界上，样本从“健康”类别转变到“疾病阳性”类别。
- 决策边界越简单越好，过于复杂的决策边界容易导致模型对训练数据的过拟合。

咱们还是举个🌰 子：

例子：糖尿病风险预测

假设我们是一家诊所，想要根据患者的体重指数（BMI）和年龄来预测他们是否有糖尿病的风险。我们将这些患者分为两类：

1. 低风险（健康）：没有糖尿病风险。

2. 高风险（糖尿病阳性）：有糖尿病风险。

我们采集了一些数据，包括每位患者的 BMI、年龄，以及他们的糖尿病风险状态（“低风险”或“高风险”）。

训练数据

假设我们有以下训练数据（BMI 和年龄的值）：

BMI	年龄	状态
22	25	低风险
24	30	低风险
35	40	高风险
40	45	高风险
30	28	低风险
38	50	高风险

其中：

- BMI：体重指数。
- 年龄：患者年龄。
- 状态：是否有糖尿病风险。

构建分类器 $h(x)$

我们想要构建一个分类器 $h(x)$ ，输入 BMI 和年龄后，能够预测患者属于“低风险”还是“高风险”。

判别函数 $d_k(x)$

为了实现分类器 $h(x)$ ，我们可以定义两类的判别函数 $d_{\text{低风险}}(x)$ 和 $d_{\text{高风险}}(x)$ 。这两个函数可以基于 BMI 和年龄来计算每个类别的得分，比如用一个简单的线性判别函数：

$$d_{\text{低风险}}(x) = w_1 \times \text{BMI} + w_2 \times \text{年龄} + b_1$$

$$d_{\text{高风险}}(x) = w_3 \times \text{BMI} + w_4 \times \text{年龄} + b_2$$

其中， w_1, w_2, w_3, w_4 和 b_1, b_2 是参数，表示每个特征在分类决策中的权重和偏置。

预测类别

给定一个新的患者数据 $x = (\text{BMI}, \text{年龄})$ ，分类器 $h(x)$ 比较两个判别函数的值：

$$h(x) = \arg \max \{d_{\text{低风险}}(x), d_{\text{高风险}}(x)\}$$

如果 $d_{\text{低风险}}(x) > d_{\text{高风险}}(x)$ ，那么分类器会预测该患者是“低风险”；否则，预测为“高风险”。

决策边界

决策边界是两个判别函数相等的点集：

$$d_{\text{低风险}}(x) = d_{\text{高风险}}(x)$$

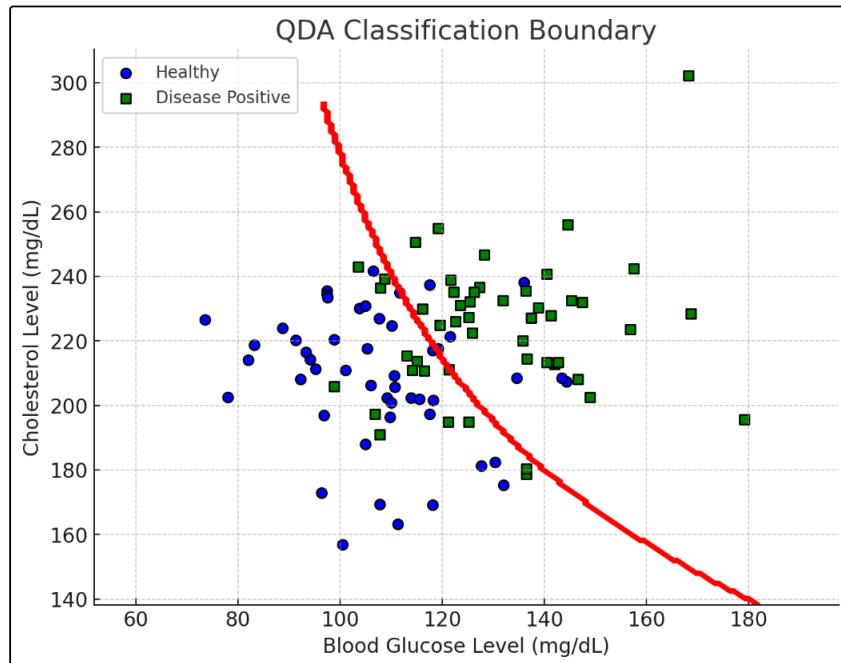
这条边界将样本空间划分为两部分，一边是“低风险”，另一边是“高风险”。在我们的二维图中，决策边界可能是一条直线，将 BMI-年龄平面分成两块区域，分别对应“低风险”和“高风险”。

例如，如果决策边界是一条直线，我们可以将患者的 BMI 和年龄坐标点放到图中：

- 如果在决策边界的“低风险”区域，则分类为低风险。
- 如果在“高风险”区域，则分类为高风险。

所以：

- 分类器 $h(\mathbf{x})$ 用于预测患者是否属于“低风险”或“高风险”。
- 判别函数 $d_k(\mathbf{x})$ 用于对每个类别计算得分。
- 决策边界则将“低风险”和“高风险”分开，是模型做出预测的依据。



- 横轴表示血糖水平，纵轴表示胆固醇水平。
- 蓝点表示“健康”类别，绿色方块表示“疾病阳性”类别。
- 红色曲线为决策边界，它区分了这两类样本。
- 在决策边界上，判别函数的值相等，即 $d_1(\mathbf{x}) = d_2(\mathbf{x})$ 。

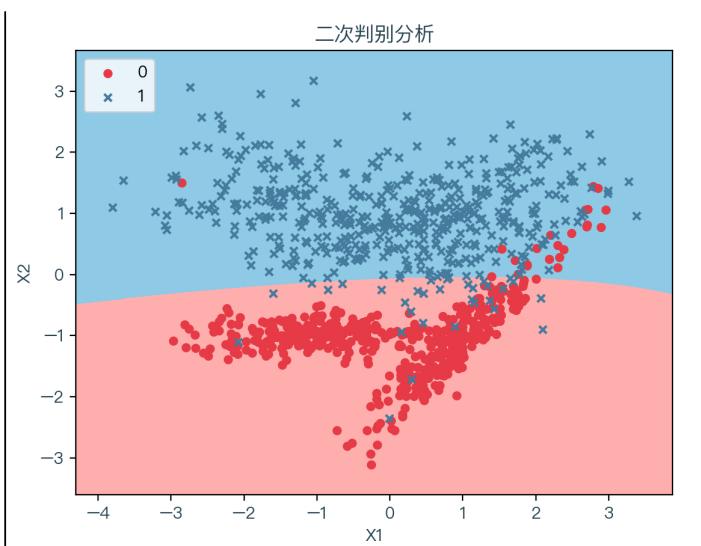
通过定义分类器、判别函数和决策边界，我们可以理解监督学习模型如何根据输入特征进行类别划分，并且简单的决策边界能够帮助模型避免过拟合，提升泛化能力。

在监督学习中，分类问题可以通过不同的方法来解决。这些方法可以分为两大类：概率方法和非概率方法。

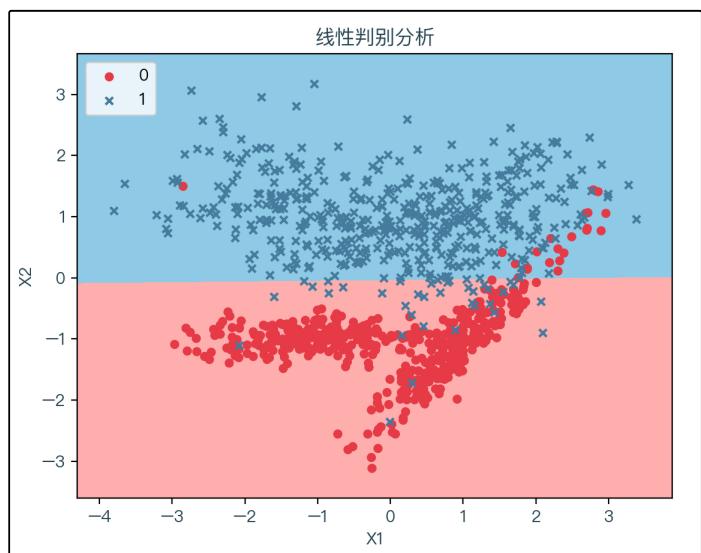
1. 概率方法 (Probabilistic Approach)

概率方法主要基于统计模型，通过估计数据属于每个类别的概率来进行分类。这类方法通常假设数据符合某种概率分布，并通过这些假设来推断样本属于不同类别的可能性。常见的概率方法包括：

- QDA (Quadratic Discriminant Analysis): 二次判别分析，假设每个类别的数据具有不同的协方差矩阵，允许决策边界为曲线。(后文会讲)



- **LDA (Linear Discriminant Analysis):** 线性判别分析, 假设每个类别的数据具有相同的协方差矩阵, 因此决策边界是直线。



- **DDA (Diagonal Discriminant Analysis):** 对角判别分析, 这是一种更简单的判别分析, 仅使用对角线上的协方差。
- **Naive Bayes Classification:** 朴素贝叶斯分类, 假设特征之间是条件独立的, 通过贝叶斯定理计算每个类别的概率。
- **Logistic Regression:** 逻辑回归, 虽然被称为回归, 但其实是一个分类方法, 尤其适用于二分类问题。

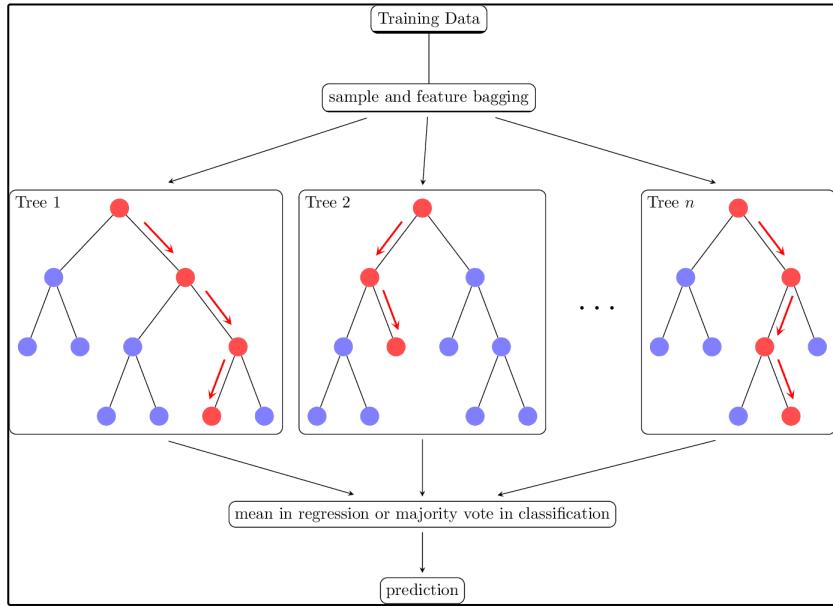


2. 非概率方法 (Non-Probabilistic Approach)

非概率方法并不直接估计类别的概率, 而是基于距离或边界的概念来进行分类。它们通常更为灵活, 可以在高维数据和非线性分类任务中表现良好。常见的非概率方法包括:

- **K-NN (K-Nearest Neighbors):** K 近邻算法, 通过找到样本点的 K 个最近邻来决定其类别。

- **SVM (Support Vector Machine)**: 支持向量机，通过找到一个最大化类别间隔的决策边界来进行分类，适合处理高维和非线性数据。
- **Random Forest**: 随机森林，基于多个决策树构建的集成方法，通过投票来确定最终类别。



- **Neural Networks**: 神经网络，通过模拟人脑神经元的工作方式来进行分类，可以处理复杂的非线性关系，适用于图像、语音等复杂数据的分类任务。

Bayes Classifier 贝叶斯分类器

贝叶斯分类器的基本概念

贝叶斯分类器的核心是**贝叶斯判别规则**，它使用概率来估计样本属于某个类别的可能性。具体来说，贝叶斯分类器基于混合模型，将数据分为 K 个类别，并在每个类别下进行如下假设：

1. 类别分布 (Group Distribution) :

- 每个类别 k 都有自己的分布 F_k ，其参数记为 θ_k 。

2. 类别密度 (Group Density) :

- 在给定类别 k 的情况下，样本的概率密度为 $f_k(\mathbf{x}) = f(\mathbf{x}|y=k)$ 。

3. 类别的先验概率 (Prior Probability) :

- 每个类别 k 都有一个先验概率 $\pi_k = P(y=k)$ ，表示样本在未观察特征时属于该类别的概率。所有类别的先验概率之和为 1，即 $\sum_{k=1}^K \pi_k = 1$ 。

4. 边际密度 (Marginal Density) :

- 样本的总概率密度可以通过类别密度的加权和表示：

$$f(\mathbf{x}) = \sum_{k=1}^K \pi_k f_k(\mathbf{x})$$

后验概率 (Posterior Probability)

贝叶斯分类器的核心是计算每个类别的后验概率，即在观察到样本 \mathbf{x} 后，该样本属于类别 k 的概率：

$$P(y=k|\mathbf{x}) = \frac{\pi_k f_k(\mathbf{x})}{f(\mathbf{x})}$$

后验概率衡量了在给定特征 \mathbf{x} 的条件下，样本属于类别 k 的可能性。

软分类 (Soft Classification)

贝叶斯分类器的分类结果可以表示为一个后验概率的向量，即软分类：

$$h(\mathbf{x}^{\text{test}}) = (P(y=1|\mathbf{x}^{\text{test}}), \dots, P(y=K|\mathbf{x}^{\text{test}}))^T$$

这表示每个类别的概率分布，分类结果并不是将样本硬性分到某个类别，而是给出每个类别的可能性。这种方式在一些需要概率输出的场景中（比如医疗诊断中不同疾病的危险概率）非常有用。

贝叶斯分类器通过先验概率和类条件概率来计算后验概率，基于后验概率进行软分类。贝叶斯分类器的一个优势是能够给出分类的概率结果，而不仅仅是类别标签。

Bayesian discriminant rule / Bayes classifier

后验概率 $P(y=k|\mathbf{x})$

后验概率表示在给定输入 \mathbf{x} 后，样本属于类别 k 的概率：

$$P(y=k|\mathbf{x}) = \frac{\pi_k f_k(\mathbf{x})}{f(\mathbf{x})}$$

其中：

- π_k 是类别 k 的先验概率。
- $f_k(\mathbf{x})$ 是类别 k 下样本 \mathbf{x} 的条件概率密度。
- $f(\mathbf{x})$ 是 \mathbf{x} 的边际密度（将所有类别的概率密度综合起来）。

硬分类 (Hard Classification)

在硬分类中，分类器会选择使后验概率最大的类别。具体来说，对于测试样本 \mathbf{x}^{test} ，分类结果为：

$$h(\mathbf{x}^{\text{test}}) = \arg \max_k P(y=k|\mathbf{x}^{\text{test}})$$

可以进一步简化为：

$$h(\mathbf{x}^{\text{test}}) = \arg \max_k (\pi_k f_k(\mathbf{x}^{\text{test}}))$$

贝叶斯判别函数 $d_k(\mathbf{x})$

为了简化计算，我们引入贝叶斯判别函数 $d_k(\mathbf{x})$ ：

$$d_k(\mathbf{x}) = \log(\pi_k f_k(\mathbf{x})) = \log \pi_k + \log f_k(\mathbf{x})$$

这样，我们可以将分类问题改写为：

$$h(\mathbf{x}^{\text{test}}) = \arg \max_k d_k(\mathbf{x}^{\text{test}})$$

这个定义让我们避免直接计算 $\pi_k f_k(\mathbf{x})$ ，并有助于数值稳定性。

判别函数与后验概率的关系

判别函数 $d_k(\mathbf{x})$ 可以映射回后验概率。后验概率可以表示为：

$$P(y=k|\mathbf{x}) = \frac{\exp(d_k(\mathbf{x}))}{\sum_{l=1}^K \exp(d_l(\mathbf{x}))} = \frac{\exp(d_k(\mathbf{x}) - d_{\max}(\mathbf{x}))}{\sum_{l=1}^K \exp(d_l(\mathbf{x}) - d_{\max}(\mathbf{x}))}$$

其中， $d_{\max}(\mathbf{x})$ 是所有判别函数值中的最大值。通过减去 $d_{\max}(\mathbf{x})$ ，我们可以避免溢出问题，确保数值稳定性。

- 后验概率计算了样本属于每个类别的可能性。

- 硬分类通过选择后验概率最大的类别来进行分类。

- 判别函数帮助我们简化了计算，并避免了直接计算概率密度时的数值不稳定性。

Normal Bayes Classifier

Quadratic Discriminant Analysis

在这里，我们介绍二次判别分析（Quadratic Discriminant Analysis, QDA），这是贝叶斯分类器的一个特殊情况，适用于类别分布都是多元正态分布的情况。

在 QDA 中，我们假设每个类别 k 的数据分布为多元正态分布，密度函数为：

$$f_k(\mathbf{x}) = N_d(\mathbf{x} | \mu_k, \Sigma_k)$$

这里：

- μ_k 是类别 k 的均值向量。
- Σ_k 是类别 k 的协方差矩阵。

判别函数 (Discriminant Function)

对于 QDA，贝叶斯判别函数 $d_k^{QDA}(\mathbf{x})$ 表示为：

$$d_k^{QDA}(\mathbf{x}) = \log \pi_k + \log f_k(\mathbf{x})$$

进一步展开为：

$$d_k^{QDA}(\mathbf{x}) = -\frac{1}{2}(\mathbf{x} - \mu_k)^T \Sigma_k^{-1}(\mathbf{x} - \mu_k) - \frac{1}{2} \log \det(\Sigma_k) + \log \pi_k$$

这里：

- π_k 是类别 k 的先验概率。
- $-\frac{1}{2}(\mathbf{x} - \mu_k)^T \Sigma_k^{-1}(\mathbf{x} - \mu_k)$ 是 \mathbf{x} 相对于类别均值 μ_k 的二次形式，考虑了样本在各个维度上的距离。
- $-\frac{1}{2} \log \det(\Sigma_k)$ 是协方差矩阵的行列式项，反映了类别 k 的数据分布范围。

决策边界

在 QDA 中，判别函数 $d_k^{QDA}(\mathbf{x})$ 在 \mathbf{x} 上是二次形式。因此，决策边界（即 $d_k^{QDA}(\mathbf{x}) = d_l^{QDA}(\mathbf{x})$ ）可以是以下两种情况：

1. 线性边界：当不同类别的协方差矩阵相等，即 $\Sigma_k = \Sigma_l$ 时，决策边界是线性的。

2. 二次边界：当不同类别的协方差矩阵不相等，即 $\Sigma_k \neq \Sigma_l$ 时，决策边界是二次的。

- QDA 假设每个类别的数据分布为多元正态分布，且不同类别的协方差矩阵可以不同。
- 判别函数是 \mathbf{x} 的二次形式，因此决策边界可以是线性或二次的，取决于类别协方差矩阵是否相同。
- QDA 比较适合处理类别之间有明显分布差异的数据。

Linear Discriminant Analysis (LDA)

LDA 是 QDA 的特例，假设所有组的协方差矩阵是相同的，即 $\Sigma_k = \Sigma$ ，不随类别 k 而变化。这种假设使得 LDA 更加简化，并且其决策边界是线性的。

判别函数

首先，我们从 QDA 的判别函数开始：

$$d_k^{QDA}(\mathbf{x}) = -\frac{1}{2}(\mathbf{x} - \mu_k)^T \Sigma_k^{-1}(\mathbf{x} - \mu_k) - \frac{1}{2} \log \det(\Sigma_k) + \log \pi_k$$

在 LDA 中，由于 $\Sigma_k = \Sigma$ 对所有类别 k 都相同，我们可以将其简化为：

$$d_k^{LDA}(\mathbf{x}) = -\frac{1}{2}(\mathbf{x} - \mu_k)^T \Sigma^{-1}(\mathbf{x} - \mu_k) + \log \pi_k$$

其中常数项被省略，因为它不依赖于 k ，对分类结果没有影响。

判别函数的进一步简化

为了进一步简化，去掉不依赖于类别 k 的项，得到：

$$d_k^{LDA}(\mathbf{x}) = \mu_k^T \Sigma^{-1} \mathbf{x} - \frac{1}{2} \mu_k^T \Sigma^{-1} \mu_k + \log \pi_k$$

此公式可以写成线性函数的形式：

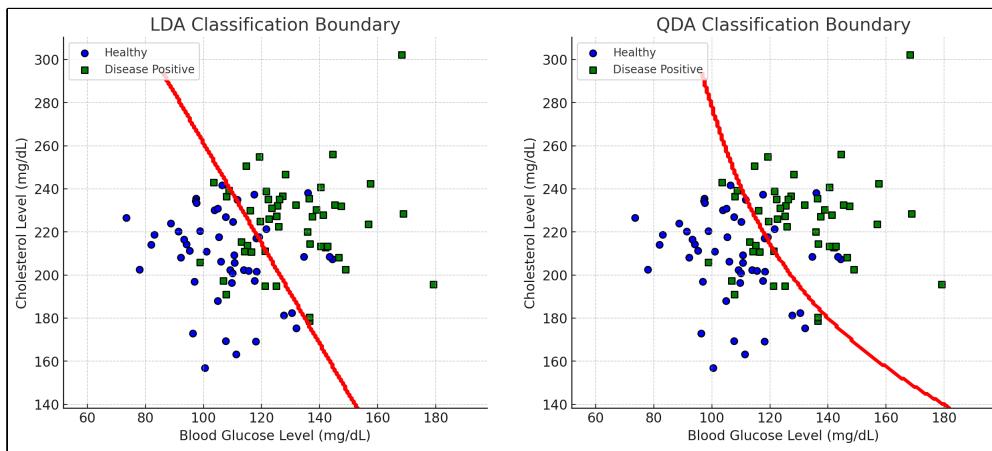
$$d_k^{LDA}(\mathbf{x}) = \mathbf{b}_k^T \mathbf{x} + a_k$$

其中， $\mathbf{b}_k = \Sigma^{-1} \mu_k$ 和 $a_k = -\frac{1}{2} \mu_k^T \Sigma^{-1} \mu_k + \log \pi_k$ 。

线性判别和决策边界

LDA 的判别函数是关于 \mathbf{x} 的线性函数，因此决策边界 $d_k^{LDA}(\mathbf{x}) - d_l^{LDA}(\mathbf{x}) = 0$ 也是线性的。这意味着不同类别之间的分界线是一个超平面。在图像中，线性决策边界通常表现为一条直线或平面，将样本空间分割成不同的区域，用于分类。

这种线性特性使得 LDA 特别适用于数据分布大致服从正态分布且协方差相同的情况。在这种情况下，LDA 能够通过线性分界准确地分类样本，并且计算效率高。



从右侧的图表中可以看到：

- LDA 决策边界（左图）是一条直线，因为它假设所有类别共享同一协方差矩阵。
- QDA 决策边界（右图）则是一个曲线，因为它允许不同类别有不同的协方差矩阵。

- LDA 适用于类别之间有较强的线性可分性，且假设不同类别的协方差相同。
- QDA 更灵活，适用于不同类别有不同的分布形状（不同的协方差矩阵）。

Diagonal Discriminant Analysis (DDA)

DDA 是在线性判别分析 (LDA) 的基础上进一步简化的方法，假设协方差矩阵是对角的。这意味着我们假定每个特征之间是独立的，协方差矩阵 Σ 被表示为一个对角矩阵 V ，其中每个特征维度的方差位于对角线位置上，而非对角线元素为零：

$$\Sigma = V = \begin{pmatrix} \sigma_1^2 & 0 & \dots & 0 \\ 0 & \sigma_2^2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \sigma_d^2 \end{pmatrix}$$

判别函数

在 DDA 中，我们的判别函数形式如下：

$$d_k^{DDA}(\mathbf{x}) = \mu_k^T V^{-1} \mathbf{x} - \frac{1}{2} \mu_k^T V^{-1} \mu_k + \log \pi_k$$

进一步展开判别函数：

$$d_k^{DDA}(\mathbf{x}) = \sum_{j=1}^d \frac{\mu_{k,j} x_j - \mu_{k,j}^2 / 2}{\sigma_j^2} + \log \pi_k$$

由于 DDA 是一种简化的 LDA，它的决策边界也是线性的。这种方法在假设特征之间是独立时非常有效，且计算简单。DDA 在高维、样本量较小的情况下特别有用，因为它对参数的需求是线性扩展的，这减少了过拟合的风险。

Naive Bayes Classifier

Naive Bayes（朴素贝叶斯）分类器是一种简单但非常有效的分类算法，它基于贝叶斯定理并假设各个特征之间是独立的。这种独立性假设使得计算变得容易，因此得名“Naive”。

假设独立性

尽管假设特征之间完全独立可能看似过于简化，但在很多实际情况下，这种假设仍然能够很好地工作。特别是在维度 d 较大、样本数 n 较少的情况下，Naive Bayes 分类器可以提供稳定且有效的结果。

特性和优势

- **简单高效：**由于假设独立性，Naive Bayes 分类器的参数数量与特征维度 d 成线性关系，因此计算量低。
 - **降低过拟合风险：**相比需要更多参数的 QDA 和 LDA，Naive Bayes 分类器更不容易过拟合。
 - **保证协方差矩阵正定：**协方差矩阵 $\hat{\Sigma}$ 可以简化为一个对角矩阵 V ，保证了正定性。
- DDA 可以视为一种 Naive Bayes 分类器，当维度 d 较大时（即样本少而特征多），DDA 和 Naive Bayes 都是优先的选择。