第一章导言

1.1 表示约定

n 表示样本中不同的数据点，即**因变量个数**

p 表示样本中的可以用于预测变量的数量，即**自变量个数**

一个 n 行 p 列的矩阵用于表示因变量相对于自变量的多次观测值：

$$X = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1p} \\ x_{21} & x_{22} & \cdots & x_{2p} \\ \vdots & \vdots & & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{np} \end{bmatrix}$$

> n 次观测，每次观测包含 p 个参数

1.2 本书结构

1 章导言
2 ~ 6 章线性统计
7 ~ 13 章非线性统计

第二章统计学习

2.1 统计学的一般概念

可约误差和不可约误差：可约误差是预测函数中可以消去的误差，通过优化模型可以使之减小；不可约误差是始终存在的误差，永远不可能为零。不可约误差记为 ϵ

> 可以认为可约误差对应偏差，不可约误差对应方差

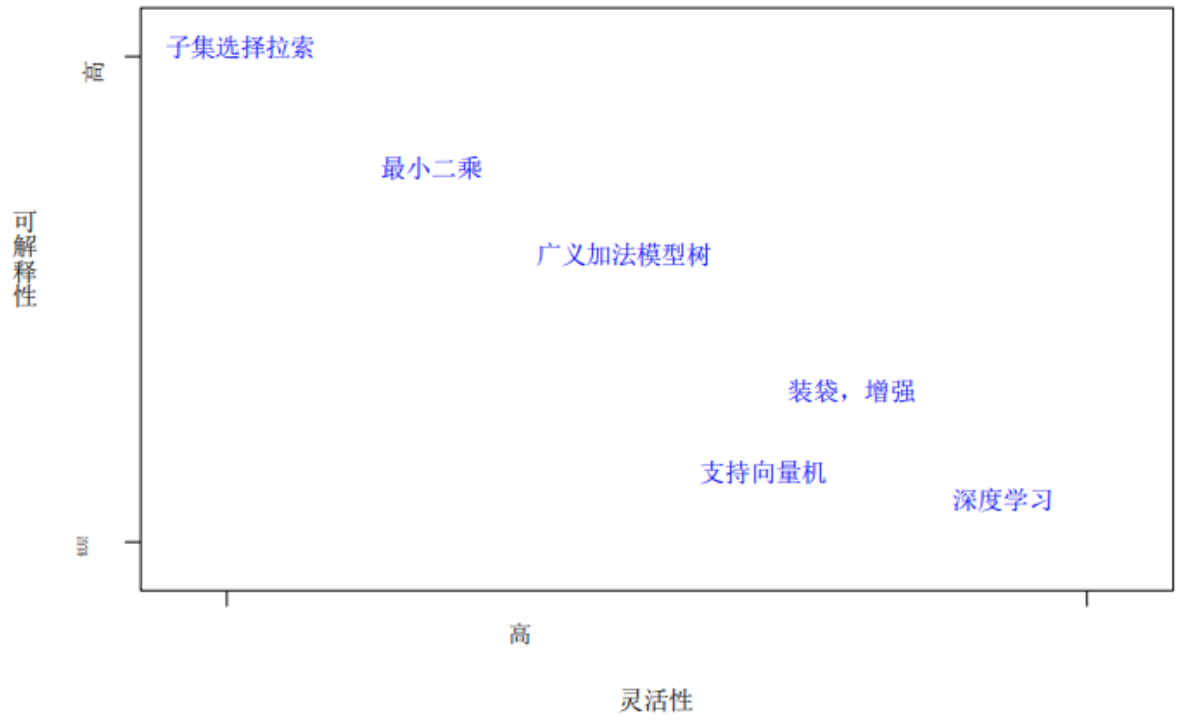
参数化方法和非参数化方法：参数化方法需要找到一系列参数，求出参数后得到估计函数；非参数化方法则是直接给出估计函数，使之完全拟合已有的数据

> 参数数量少的估计函数，估计误差比较大；而复杂的模型则可能出现过拟合的问题

监督学习和无监督学习：监督学习包含训练集和测试样本，典型的有**回归分析**；而无监督学习没有测试样本，需要在数据集中寻找规律，典型的是**聚类**

回归与分类问题：变量可以采用数值的是定量问题，否则是定性问题。前者采用回归分析，后者采用分类。

2.2 统计方法的可解释性和灵活性分布



2.3 统计模型准确性的评估

最近邻方法 (Nearest neighbor)： 考虑一个比较小的数据集，若在某点上没有对应数据，则可用附近点的值作平均来估计该点值大小

> 此方法在高维度时效果不好，这被称为**高维度诅咒 (curse of dimensionality)**

均方误差：均方误差 (MSE) 可以用于衡量预测值在多大程度上接近于观测的真实值，它通过计算预测值与实际观测值之间差异的平方的平均值来衡量预测的准确度。MSE 越小，说明模型的拟合效果越好。

$$MSE = \frac{\sum_{i=1}^{n}(y_i - \hat{f}(x_i))^2}{n}$$

训练均方误差：用于训练的数据集与测试模型之间误差平方的平均值，记为训练均方误差 (tranning MSE)

测试均方误差：训练模型完成后，用一组用于检测模型训练效果的数据集计算 test MSE，根据

$$Ave(y_0 - \hat{f}(x_0))^2$$

比较哪个模型估计得最精确

> 训练均方误差和测试均方误差使用的数据集不同。前者使用的是训练数据集，后者使用的是测试数据集

自由度：自由度可以用于描述统计模型的灵活性。随着模型灵活性的增加，训练 MSE 会降低，但测试 MSE 可能不会降低。当一个模型的训练均方误差很小但测试均方误差比较大时，我们认为这个模型出现了**过拟合**

> 过拟合特指灵活性较差的模型会产生较小的测试均方误差的情况。模型通过不断地拟合训练数据集，可以对训练数据集产生很小的均方误差

期望测试均方误差：用大量的训练集重复估计 f，并在 x_0 处分别检验，可以得到期望测试均方误差 (expected test MSE)。对测试集中 x_0 的所有可能值取平均，可以计算期望测试均方误差。

$$E(y_0 - \hat{f}(x_0))^2 = Var(\hat{f}(x_0)) + [Bias(\hat{f}(x_0))]^2 + Var(\epsilon)$$

> $Var(\hat{f}(x_0))^2$ 表示不同数据集在 x_0 处给出的 $\hat{f}(x_0)$ 的变异程度，$[Bias(\hat{f}(x_0))]^2$ 表示模型与实际值偏差的程度

方差：方差 (variance) 可以用来表示模型对检验数据集的拟合程度。

> 方差是相对于检验数据集而言的。高方差意味着模型对测试数据集的变动非常敏感，泛化能力差

偏差：偏差 (bias) 是指用一个简单得多的模型去近似一个可能非常复杂的实际问题所引入的误差。普遍来讲，更高自由度的模型具有更小的偏差

> 偏差指的是模型预测值与真实值之间的差距，高偏差意味着模型对测试数据集和训练数据集的拟合程度都不好

方差和偏差的相对变化率，决定了 test MSE 是增大还是减小。因此，test MSE 随着自由度的变化曲线是一个向下的抛物线

方差-偏差权衡：方差-偏差权衡 (bias-variance trade-off) 表明，在实际的模型选择中，我们需要对模型的方差和偏差找到一个平衡点。过高的偏差可能导致模型无法识别数据特征，过高的方差则可能导致模型对噪声敏感

2.4 统计模型分类体系

误差率：通过计算

$$\frac{1}{n}\sum_{i=1}^{n} I(y_i \neq \hat{y}_i)$$

可以得到模型的误差率 (error rate)，从而判断模型的准确程度。这个数据表示了分类的错误比率。

训练误差：分类器对训练数据的误差率被称为训练误差

测试误差：分类器对测试数据的误差率被称为测试误差

$$Ave(I(y_0 \neq \hat{y_0}))$$

2.4.1 贝叶斯分类器

贝叶斯分类器 (Bayes Classifier) 是一种基于贝叶斯定理的统计分类方法。

记 Y 为类别，X 为特征，则根据**预测向量**x_0 的值将其分配给类中概率最大的第 j 类，概率可表示为：

$$Pr(Y=j|X=x_0)$$

在一个二元分类中，当 $Pr(Y=j|X=x_0)>0.5$ 时，我们认为它是属于分类 j 的；在 $Pr(Y=j|X=x_0)=0.5$ 处，形成了**贝叶斯决策边界 (Bayes decision boundary)**

贝叶斯分类器产生的最低可能的错误率，被称为**贝叶斯错误率 (Bayes error rate)**：

$$1-max_jPr(Y=j|X=x_0)$$

> 因为贝叶斯分类器总是选择贝叶斯概率最大的分类，因此实际上非此分类的变量都认为是发生了错误。$max_j(Pr(Y=j|X=x_0)$ 表示的是 $X=x_0$ 时，X 最有可能得分类

根据上式，可以计算总体贝叶斯错误率：

$$1-E(max_jPr(Y=j|X=x_0))$$

> 贝叶斯误差率类似于不可约误差
> 贝叶斯分类是测试其他分类性能的金标准

2.4.2 K 近邻

K 近邻 (K-Nearest Neighbors KNN) 是一种通过测量不同特征值之间的距离来进行预测的分类方法

选取一个合适的 K 值，对于一个观测变量，考虑训练数据与其最近的 K 个点，记为 N_0，然后将第 j 类的条件概率估计为 N_0 中响应值为 j 的点的分数：

$$Pr(Y=j|X=x_0)=\frac{1}{K}\sum_{i\in N_0}I(y_i=j)$$

最后选取 x_0 在不同分类中的一个最大的概率，作为 x_0 的分类

当 K 值越大时，K 近邻的灵活性下降；当 K 值越小时，K 近邻的灵活性上升。过高的灵活性可能出现过拟合，因此当 K 值上升时，实验误差呈现U型特征

> 例如，x_0 临近的 K 个值中，黄色点最多，则考虑 x_0 也是黄色点

第三章线性回归

线性回归的特点：

1. 历史悠久。许多现代方法是其延伸
2. 相比于其他现代方法，显得有些笨拙 (dull)
3. 依旧非常常用

参数的研究包含：

1. 参数之间是否有关，相关性多大?
2. 每个参数和因变量的关系如何?
3. 我们能对未来作出多精确的预测?
4. 参数之间的关系是线性的吗?
5. 参数之间是否存在协同作用?

> **协同作用 (synergy)** 在统计中称为**相互作用 (interaction)**

3.1 简单线性回归

简单线性回归:
简单线性回归 (simple linear regression) 是一种用线性回归方式预测单个变量 X 和响应变量 Y 之间关系的统计方法。二者之间的关系可用如下方程表示:

$$Y \approx \beta_0 + \beta_1 X$$

我们有时会说我们在 X (或 Y 到 X 上) 上对 Y 进行回归来描述

> $\approx$ 表示“大约就像”

在线性回归模型中，我们需要确定 β_0 和 β_1 两个参数，他们分别称为**截距 (intercept)** 和 **斜率 (slope)**，也能称为**模型系数 (coefficient)** 或 **参数 (parameter)**

获取参数后，可以得到:

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x$$

3.1.1 参数估计

在实践中，β_0 和 β_1 是未知的，因此需要用:

$$(x_1, y_1), (x_2, y_2), \cdots, (x_n, y_n)$$

n 对测量值来估计参数

我们需要找到这样一条直线，使其尽可能地接近我们的 n 对数据。对于线性回归，最常用的方法是**最小二乘法 (least squares)**

考虑每个预测值与实际值的差 $e_i = y_i - \hat{y}_i$，则定义**残差平方和 (residual sum of squares RSS)**:

$$RSS = e_1^2 + e_2^2 + \cdots + e_n^2 = \sum_{i=1}^{n} e_i^2$$

或等价于:

$$RSS = (y_1 - \hat{\beta}_0 - \hat{\beta}_1 x_1)^2 + (y_2 - \hat{\beta}_0 - \hat{\beta}_1 x_2)^2 + \cdots + (y_n - \hat{\beta}_0 + \hat{\beta}_1 x_n)^2$$

最小二乘法选择这样一对 β_0, β_1，使得 RSS 最小，可以证明此时:

$$\hat{\beta}_1 = \frac{\sum_{i=1}^{n}(x_i - \overline{x})(y_i - \overline{y})}{\sum_{i=1}^{n}(x_i - \overline{x})^2}$$

$$\hat{\beta}_0 = \overline{y} - \hat{\beta}_1\overline{x}$$

其中的 $\overline{x}$ 和 $\overline{y}$ 分别表示样本平均数

3.1.2 评估系数估计值的准确性

对于均值为 0 的随机误差项 ϵ，我们在线性回归模型中考虑之，则：

$$Y = \beta_0 + \beta_1 X + \epsilon$$

在上面的式子中，β_0 是截距，也就是 $X = 0$ 时，Y 的期望值。β_1 是斜率，也就是 Y 的平均增加与 X 的一个单位增加有关

在总体中，根据总体得到的回归直线是**总体回归线 (population regression line)**，这条直线不会改变。根据样本和最小二乘法得到的直线被称为**最小二乘直线 (least squares line)**，会随着观测数据改变

> 在实际中，总体回归线一般无法测量，只能用最小二乘直线估计

无偏估计 (unbiased estimation) 指的是估计量的均值等于被估计量的真实值，也就是说，在大量的观测下的估计值的均值会趋近于真实值。与之对应的**有偏估计 (biased estimation)** 则估计量的均值不等于估计值的真实值，即存在系统误差

考虑样本均值 μ 的方差：

$$Var(\hat{\mu}) = SE(\hat{\mu})^2 = \frac{\sigma^2}{n}$$

其中 SE 表示标准差，σ 是总体的标准差。这个式子表明，观测值越多，样本均值的方差越小，也就是样本误差越小。

> **总体方差 (Population Variance)** 是所有数据与其均值差的平方的均值，通常不可得，只能获取样本方差
> 样本均值的方差不是样本的方差，是每个样本与样本均值差的平方的平均数

计算 $\hat{\beta}_0$ 和 $\hat{\beta}_1$ 的方差：

$$SE(\hat{\beta}_0)^2 = \sigma^2[\frac{1}{n} + \frac{\overline{x}^2}{\sum_{i=1}^{n}(x_i - x)^2}]$$

$$SE(\hat{\beta}_1)^2 = \frac{\sigma^2}{\sum_{i=1}^{n}(x_i - \overline{x})^2}$$

此处 σ^2 是总体的方差 (误差项的方差)，也就是残差平方和 RSS 的平均数

> 从上面的式子可以发现，x 的分布越分散，也就是 x 在横轴上的数据范围越大，估计参数的误差就越小

上述公式的假设条件是：

1. 误差项 ϵ 服从正态分布，且对每个观测值具有常数方差
2. 误差项之间相互独立
3. X 与 ϵ 无关

一般来说，σ^2 不能获取，但可以估计：

$$RSE = \sqrt{\frac{RSS}{n-2}}$$

这个估计量被称为**残差标准误差 (residual standard error)**

置信区间 (confidence interval) 是一个取值范围，使得在某个概率下，该范围将包含参数的真实未知值。置信区间计算：

$$x \pm 2SE(\hat{x})$$

标准误差也可用于**假设检验 (hypothesis test)**，假设检验通常涉及**零假设 (null hypothesis)**：

$$H_0 : There\ is\ no\ relationship\ between\ X\ and\ Y$$

区别于**备选假设 (alternative hypothesis)**：

$$H_a : There\ is\ some\ relationship\ between\ X\ and\ Y$$

在数学上，这对应于测试：

$$H_0 : \beta_1 = 0$$

相对于：

$$H_\alpha : \beta_1 \neq 0$$

实践中常常采用 t-检验。例如，对于线性回归中的 $\hat{\beta}_1$ 计算 t：

$$t = \frac{\hat{\beta}_1 - 0}{SE(\hat{\beta}_1)}$$

衡量了 $\hat{\beta}_1$ 远离 0 的标准差个数。若 X 和 Y 之间不存在关系，我们期望上式具有 $n-2$ 个自由度的t分布。

> t 分布具有钟型，在 $n > 30$ 时接近正态分布

根据 t 分布，我们很容易计算绝对值大于等于 $|t|$ 的任意数的概率，我们称之为**p 值 (p - value)**。当 p 值很小时，我们认为很大概率观测变量和响应变量有联系，此时拒绝零假设。

> 拒绝原假设的 p 临界值是 5%或 1%，当 $n = 30$ 时，分别对应 t 统计量约为 2 和 2.75

3.1.3 评估模型准确性

3.1.3.1 残差标准误差

计算残差标准误差 (RSE)：

$$RSE = \sqrt{\frac{1}{n-2}RSS} = \sqrt{\frac{1}{n-2}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

> RSS 就是残差平方和

RSE 是模型对数据不拟合程度的一种测量，可以估计平均预测数据相对于真实数据的偏移程度

3.1.3.2 R^2 统计量

由于 RSE 的大小不好度量，我们考虑使用 R^2 **统计量 (R^2 Statistic)**：

$$R^2 = \frac{TSS - RSS}{TSS} = 1 - \frac{RSS}{TSS}$$

其中 $TSS = \sum_{i=1}^{n}(y_i - \overline{y})^2$，也即响应变量相对于均值差的平方的和 (total sum of squares)。

> TSS 满足 $TSS = ESS + RSS$，其中 $ESS = \sum_{i=1}^{n}(\hat{y}_i - \overline{y})^2$ 表示模型解释的变异性，$RSS = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2$ 表示未能解释的变异性。因此 R^2 用来表示模型对变异性的解释了多少

R^2 标明了预测变量相对于观测变量的变动而变动的比例；通常 R^2 高的模型拟合程度较高

> 不同情景下要求的 R^2 大小不同。已知高线性相关的模型时，期望一个高 R^2 值；已知不相关的两个变量，则期望一个低的 R^2 值

相关性 (correlation) 也是用于衡量两个变量之间相关性的指标：

$$Cor(X, Y) = \frac{\sum_{i=1}^{n}(x_i - \overline{x})^2(y_i - \overline{y})^2}{\sqrt{\sum_{i=1}^{n}(x_i - \overline{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \overline{y})^2}}$$

也称为**皮尔逊相关系数 (Pearson correlation coefficient)**，为正值时表示正相关，且越大表示相关性越强

> 在只有一个自变量和因变量时，$R^2 = r^2$，此时二者的计算公式相同。R^2 可以适用于多个自变量的模型，侧重于解释多个变量的影响；r^2 则侧重两个变量的相关性

3.2 多元线性回归

考虑线性回归涉及的多个变量，这些变量有可能相互影响 (在一元线性回归中无法展现)，用一个多元方程：

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p + \epsilon$$

其中包含 p 个变量，且每个变量的影响程度用 β_j 来度量

3.2.1 估计线性回归的参数

获得估计参数 $\hat{\beta}_j$ 后，我们有：

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x_1 + \hat{\beta}_2 x_2 + \cdots + \hat{\beta}_p x_p$$

选择最优的 $\hat{\beta}_j$，使得：

$$\begin{aligned} RSS &= \sum_{j=1}^{n} (y_i - \hat{y}_i)^2 \\ &= \sum_{i=1}^{n} (y_i - \hat{\beta}_0 - \hat{\beta}_1 x_1 - \cdots - \hat{\beta}_p x_p) \end{aligned}$$

能获得最小值

由于相关性，在简单线性回归中包含相关性的两个变量，可能在多元线性回归中表现出无相关性。这是因为自变量之间可能存在相互影响，简单线性回归忽略了这种影响关系 (可以从相互性矩阵中得到)

3.2.2 多元线性回归关心的问题

3.2.2.1 预测变量和响应变量之间是否存在关系

与简单线性回归类似，进行零假设：

$$H_0 : \beta_1 = \beta_2 = \cdots = \beta_p = 0$$

备选假设：

$$H_a : \text{at least one } \beta_j \; is \text{ non-zero}$$

计算**F 统计量 (F-statistic)**：

$$F = \frac{(TSS - RSS)/p}{RSS/(n-p-1)}$$

其中 $TSS = \sum (y_i - \overline{y})^2$ 且 $RSS = \sum (y_i - \hat{y}_i)^2$

> F 检验的目的是构建一个满足 F 分布的 F 统计量，比较模型解释变动和不能解释变动二者方差的比，若这个值接近 1，说明模型解释不好

如果我们的模型是正确的，则：

$$E\{RSS/(n-p-1)\} = \sigma^2$$

否则若零假设成立有：

$$E\{(TSS - RSS)/p\} = \sigma^2$$

据上可知，当响应变量和预测变量不存在关系时，我们预期 F 统计量的取值接近于 1；若 H_a 为真，预期 $F > 1$

有时候，我们希望检验 p 个变量的子集是否与响应变量有关，此时零假设可设为：

$$H_0 : \beta_{p-q+1} = \beta_{p-q+2} = \cdots = \beta_p = 0$$

此时的 F 统计量为：

$$F = \frac{(RSS_0 - RSS)/q}{RSS/(n-p-1)}$$

3.2.2.2 确定重要变量

模型中的所有的预测变量可能都与响应有关，然而更多的情况下只是其中的一个子集，常用的指标包含：

1. **马洛斯 C_p 值 (Mallow's C_p)**
2. **赤池信息准则 (Akaike information)**
3. **贝叶斯信息准则 (Bayesian information criterion)**
4. **调整后的 R^2 (adjusted R^2)**

考虑每种可能，对于 p 个参数，我们需要测试 2^p 个模型，全部测试会占用大量的计算资源。常用的不全部测试模型的方法有：

1. **前项选择 (Forward selection)**：从一个只含有截距项目的**零模型 (null model)** 开始，逐步地添加变量 (可以依据 t 统计量等)，每次添加后重新评估剩余变量，直到达到显著性水平的限定
2. **后项选择 (Backward selection)**：从含有全部参数的模型开始，逐步减少变量，直到达到显著性水平的限定
3. **混合选择 (Mixed selection)**：混合使用前面两种方法，直到达到设定的显著性水平

当 $p > n$ 时，不能使用后项选择；任何时候都可以使用前项选择，然而最初选择的变量后来可能变得冗余 (贪心策略)，混合选择弥补了这一点

3.2.2.3 模型适配

R^2 表示模型解释的变异性占总变异性的比例。一般来说，R^2 越接近 1，模型的适配程度越好，预测的能力越强。然而，加入一个预测变量都有可能使 R^2 增加，这个增加值可能很微弱，此时也可以证明此预测变量和响应变量弱相关

> R^2 在添加变量后总增加

绘出响应变量与两个预测变量之间的二维图，可以发现模型容易高估单独的预测变量的作用，而低估两个变量相互分割的作用。这表明，预测变量之间存在的相互作用，要比单独一个变量的作用影响更大

3.2.2.4 预测

获取估计的参数后，我们得到二维预测平面：

$$\hat{Y} = \hat{\beta}_0 + \hat{\beta}_1 x_1 + \cdots + \hat{\beta}_p x_p$$

这是对真实回归平面：

$$f(X) = \beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p$$

的一个估计。系数的不准确性与可约误差有关，可通过计算置信区间来计算两个平面的接近程度

我们假设的线性模型是对现实的一种近似，因此存在一个额外可以减少的误差来源，称为**模型偏差 (model bias)**。我们使用的线性模型估计的是真实表面的最佳线性近似，将会忽略这种差异，认为模型正确

即使知道参数的真实值，由于随机误差 ϵ，我们也不能完美地预测响应值 (不可约误差)。使用**预测区间 (prediction intervals)** 来考虑误差。这个区间总是比置信区间宽，因为包含了可减少误差也包含了不确定性

> 使用置信区间，估计的是参数平均的变动范围 (真实值在此区间取到)；而使用预测区间，衡量的则是预测某一特定对象参数的变动范围，因此范围波动更大

3.3 线性回归模型其他需要考虑的量

3.3.1 定性预测因子

因子水平 (factor level)：指的是一个因子（或称为变量）可以取的不同值或状态，是离散且有限的

哑变量 (dummy variable)：定性分析时，可以考虑将有限个变量取值用 0 和 1 表示，这个变量就是哑变量

> 例如，男性为 1，女性为 0；红色为 red 属性 1，绿色为 green 属性 1，两个属性都为 0 表示蓝色，这时候蓝色就是基准线 (base line)
> 在机器学习社区中，创建哑变量来定性预测变量被称为"独热编码 (one-hot encoding)"

3.3.2 线性回归的扩展

经典线性回归模型的两个基本假设是：

1. 变量之间可加，且每个单位的变化导致的变化是相同的
2. 变量的变化是线性的

3.3.2.1 移除可加假设

统计学中的交互作用 (interaction effect)，在市场营销学中称为协同效应 (synergy effect)，提供了一种扩展线性回归模型的思路，即添加一个交互项作为预测因子：

$$Y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_1 x_2 + \epsilon$$

使用交互项可以减轻加法假设，注意到：

$$
\begin{aligned}
Y &= \beta_0 + (\beta_1 + \beta_3 x_2)x_1 + \beta_2 x_2 + \epsilon \\
&= \beta_0 + \tilde{\beta}_1 x_1 + \beta_2 x_2 + \epsilon
\end{aligned}
$$

其中 $\tilde{\beta}_1 = \beta_1 + \beta_2 x_2$。此时 $\tilde{\beta}_1$ 是 x_2 的一个参数，因此 x_1 和 Y 的联系不再呈常数增加

主要效应 (main effect)：是指非交互项的效应。交互项的效应可能强于主要效应

层次性原则 (hierarchical principle)：指的是若采用了交互项 (交互项与 Y 有一定相关性)，即使形成交互项的两个预测变量的 p 值较大，也应当把他们考虑到模型中

> 事实上，定性变量和定量变量之间也能产生交互项，且这对模型的优化效应是显著的

3.3.2.2 非线性关系

多项式回归 (polynomial regression) 是一种简单的拟合非线性关系的方法。例如，对于**二次的 (quadratic)** 散点图，可以考虑：

$$
Y = \beta_0 + \beta_1 x + \beta_2 x^2 + \epsilon
$$

这仍然是一个线性模型——我们将 x 看作 X_1，x^2 看作 X_2

> 增加多项式的次数，可能产生更好的拟合效果，但我们不能确定是否必要，即可能产生过拟合

3.3.3 潜在问题

使用线性回归可能的问题包括：

1. 非线性关系
2. 误差项的相关性
3. 误差项的非常数方差
4. 离群值
5. 高杠杆点
6. 共线性

3.3.3.1 数据的非线性

残差图 (residual plots) 可用来判别数据的线性关系。较好的模型，残差值应当落在 0 附近

> 使用不同的非线性参数如 $\sqrt{x}$ 和 x^2 等，用残差图比较模型拟合程度

3.3.3.2 误差项的相关性

线性模型的一个假设是误差项是不相关的。若相关，我们可能高估置信区间，低估 p 值，错误地信任模型

误差项的相关常常发生在**时间序列数据 (time series data)** 中。在这种情况下，相邻时间上的数据的残差倾向于取相似的值

> 在残差图上，可能出现喇叭形，也就是异方差

3.3.3.3 误差项的非恒定方差

残差的大小随着拟合值的增大而增大，在残差图中就是出现了**异方差 (heteroscedasticity)** 。面对此问题，可以考虑用凹函数如 $\log Y$ 或 $\sqrt{Y}$ 进行变换，使得较大的预测量对应的响应量降低，从而降低异方差

用**加权最小二乘法 (weighted lease squares)** 给方差最小的观测值以更高的权重，使模型的拟合度更高

3.3.3.4 离群值

离群值 (outliers)：离群值是离模型拟核区域较远的值。它们可能发生于数据搜集过程中错误地记录

用残差图可以识别异常值——它的残差非常大。然而，如何确定足够大的残差以识别离群值是一个新的问题。我们使用**学生化残差 (studentized residual)** 来解决

> 离群点可能是由错误地数据搜集导致的，此时我们删除之；离群点也可能表明模型的缺陷，如一个缺失的预测变量

3.3.3.5 高杠杆点

高杠杆点 (High Leverage Points)：是指距离大部分观测点 x_j 较远的观测值。它们对回归直线的影响比离群值更加显著，因此很有必要识别

在低维空间中容易识别高杠杆点，通过画图可以很容易发现哪些点是高杠杆点。然而在高维空间中，我们不容易作图识别。为了量化高杠杆点，我们计算**杠杆统计量 (leverage statistic)**：

$$h_i = \frac{1}{n} + \frac{(x_i - \overline{x})^2}{\sum_{i'=1}^{n}(x_{i'} - \overline{x})^2}$$

杠杆值的取值在 1 到 $\frac{1}{n}$ 之间，杠杆的平均值落在 $\frac{(p+1)}{n}$。若有一个观测值的杠杆远超于 $\frac{(p+1)}{n}$，我们认为该点具有高杠杆性

3.3.3.6 共线性

共线性 (Collinearity)：是指多元线性模型中两个预测变量存在的较大的相关性 (同时增加或同时减少)。这种相关性可以从相关性矩阵中得到

当两个变量存在较高共线性时，由于它们同时变化的特性，很难找到一对值，使得模型 RSS 最小。计算它们组成的模型 t 统计量较小，p 值较大，可能会使我们放弃模型，这就使检查相关性的假设失效

多个变量之间存在的共线性叫做**多重共线性 (multicollinearity)**，此时通过计算**方差膨胀因子 (variance inflation factor, VIF)**：

$$VIF(\hat{\beta}_j) = \frac{1}{1 - R^2_{X_j|X_{-j}}}$$

其中 $1 - R^2_{X_j|X_{-j}}$ 是要计算的变量作为因变量，其他变量作为自变量的线性回归的 R^2。当 VIF 接近 1 时，说明共线性小；若 VIF 很大，考虑高度共线性

解决共线性的方法包括：

1. 直接在模型中删掉其中一个共线变量
2. 考虑一个新的变量，例如两个共线变量的平均数

3.4 KNN 与线性回归的比较

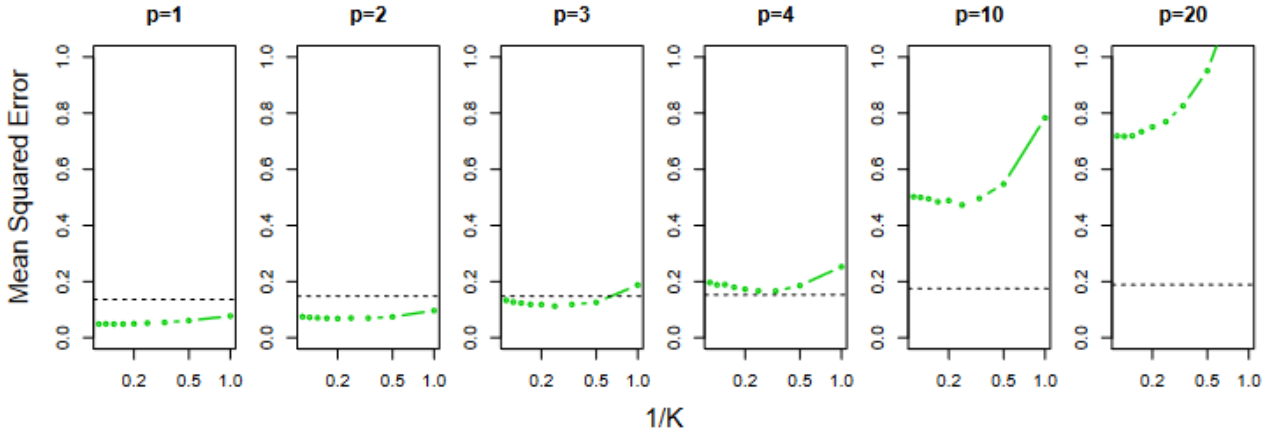
K 近邻回归 (K-nearest neighbors regression)：是一种类似于 K 近邻分类器的方法。设定一个 K 值，对于一次预测变量 X_0，将会找到与其最近的 K 的训练集中的 x，并根据这些 x 的平均数回答 X_0 对应的值

K 近邻回归是非参数化方法。当 K 值很小时，拟合的数据较少，拟合平面凹凸程度大；当 K 值较大时，拟合平面更平滑

对于高度线性的数据，K 近邻回归在 K 值较大时拟合程度较好，K 值较小时拟合程度差。两种情况下的拟合程度劣于线性回归

对于非线性的数据，当非线性程度大时，K 近邻回归预测效果较好，且较大的 K 值预测效果更好

在高维度数据上，KNN 模型的 RSE 劣化速度比线性回归快得多。这是因为高维度数据的相邻数据比低纬度稀疏得多，会遇到所谓的“高维度诅咒”。因此参数模型在数据较少时性能比非参数模型高



尽管维度很低，我们也优先考虑线性回归模型，虽然可能会损失一些预测精度。因为线性回归模型的可解释性较好，且模型更简单

第四章分类

本章介绍的分类器：

- 逻辑斯蒂回归 (Logistic regression)
- 线性判别分析 (linear discriminant analysis)
- 二次判别分析 (quadratic discriminant analysis)
- 朴素贝叶斯 (naive Bayes)
- K-近邻 (K-nearest neighbors)
- 广义线性模型 (generalized linear models)
- 泊松回归 (Poisson regression)

4.1 分类概述

分类问题的例子：

- 急诊室病人有几个病症特征，确定其疾病
- 网络银行根据交易的 IP、历史记录等判断交易是否欺诈
- 在患病和不患病的动物中提取 DNA，确定哪些序列会造成疾病

4.2 为什么不线性回归

在上面的急诊室例子中，若使用线性回归，则需要将可能得情况映射到一些数字中。实际上，不能确定不同情况之间的差距有多大，不同的数字编码会造成完全不一样的判断，因此不能使用线性回归 (尤其是多个分类的情况下。二元回归翻转数字结果相同，但在概率上可能超过[0-1]值域)

> 例如将 3 种疾病作为分类，无法用数字衡量它们之间判断的差距

不使用线性回归的原因：

1. 线性回归不能处理多个取值定性变量的情况
2. 即使是两个取值，线性回归也可能无法解释超过 0-1 区间的预测值

> 此处的预测值指 Y

4.3 逻辑斯蒂回归

用逻辑斯蒂回归，可以将概率控制在[0-1]之间；使用线性回归，概率可能超过这个值域 (在二元回归中)

4.3.1 逻辑斯谛模型

用**逻辑斯蒂方程**，可以将概率的值落到[0-1]之间，方程形式如下：

$$p(X) = \frac{e^{\beta+0+\beta_1 X}}{1 + e^{\beta_0+\beta_1 X}}$$

逻辑斯蒂方程总能将概率转化为一个 S 型曲线。在数学上变换我们有：

$$\frac{p(x)}{1-p(x)}=e^{\beta_0+\beta_1X}$$

其中的 $\frac{p(x)}{1-p(x)}$ 被称为**赔率 (*odds*)**，表示的是发生概率与不发生概率的比，取值总在 $0-1$ 之间。

> 一些与赌注有关的活动常常使用赔率而不是概率

取对数后有：

$$\log(\frac{p(x)}{1-p(x)})=\beta_0+\beta_1X$$

其中 $\log(\frac{p(x)}{1-p(x)})$ 被称为**对数赔率 (*logit*)**。我们注意到在逻辑斯蒂方程中，对数赔率是 X 的线性函数

> 虽然概率 $p(X)$ 并不是 X 的线性函数，我们仍可以发现 X 在正方向上的变化会引起 $p(X)$ 在正方向上的变化

4.3.2 估计回归参数

我们试图找到估计参数 β_0 和 β_1，使得对于所有状态为*非*的个体，获得一个概率接近 0 的数；对所有状态为*是*的个体，获得概率接近 1 的数，用**极大似然法**可以获得一个**近似函数**：

$$l(\beta_0,\beta_1)=\prod_{i:y_i=1}p(x_i)\prod_{i':y_i=0}(1-p(x_{i'})$$

> 最小二乘法是极大似然估计的一种特殊情况

4.3.3 做出预测

对于定性变量，可以考虑哑变量来设置逻辑斯蒂方程。例如将状态是设置为 1，状态否设置为 0

4.3.4 多元逻辑斯蒂回归

类比线性回归，多元逻辑斯蒂方程可以写为：

$$\log(\frac{p(x)}{1-p(x)})=\beta_0+\beta_1X+\beta_2X_2+\cdots+\beta_pX_p$$

或者用向量表示为：

$$z=\mathbf{w}^T\mathbf{x}+b$$

同样地：

$$p(X)=\frac{e^{\beta_0+\beta_1X+\beta_2X_2+\cdots+\beta_pX_p}}{1+e^{\beta_0+\beta_1X+\beta_2X_2+\cdots+\beta_pX_p}}$$

用极大似然法估计系数参数

用一个预测变量进行逻辑斯蒂回归发生预测错误的风险比多个预测变量时大得多，这种现象被称为**混杂 (*confounding*)**

> 混杂变量的出现需要满足混杂变量与 X 和 Y 都相关，且会影响他们之间的关系

4.3.5 多项逻辑斯蒂回归

当分类情况超过两类时，逻辑斯蒂回归需要扩展为**多项逻辑斯蒂回归 (*multinomia logistic regression*)**

> 此处的分类情况指的是 Y 的取值；应当与多元逻辑斯蒂回归相区分

此时，我们的概率模型将转换为：

$$Pr(Y=k|X=x)=\frac{e^{\beta_{k0}+\beta_{k1}x_1+\cdots+\beta_{kp}x_p}}{1+\sum_{l=1}^{K-1}e^{\beta_{l0}+\beta_{l1}x_1+\cdots+\beta_{kp}x_p}}$$

其中的 $k=1,2,\cdots,K-1$，K 表示分类种类数，并有：

$$Pr(Y=K|X=x)=\frac{1}{1+\sum_{l=1}^{K-1}e^{\beta_{l0}+\beta_{l1}x_1+\cdots+\beta_{lp}x_p}}$$

不难发现：

$$\log(\frac{Pr(Y=k|X=x)}{Pr(Y=K|X=x)})=\beta_{k0}+\beta_{k1}x_1+\cdots+\beta_{lp}x_p$$

> 当 $X=x$ 时，$Y=K$ 的概率被记为基线 (*baseline*)，基线的选择对分类结果没有影响；然而，不同基线选择会导致模型的解释情况不同，因为逻辑斯蒂回归产生的结果是相对于基准线的赔率，不同的基准线的解释情况不同

softmax 编码是多分类问题中常用的一个函数，它将所有的类别视为对称的而非选择一个基线，每种类别都将输出一个对应的概率值：

$$Pr(Y=k|X=x)=\frac{e^{\beta_{k0}+\beta_{k1}x_1+\cdots+\beta_{kp}x_p}}{\sum_{l=1}^{K}e^{\beta_{l0}+\beta_{l1}x_1+\cdots+\beta_{lp}x_p}}$$

在此情况下，第 k 类和第 k' 类的比值赔率相等：

$$\log(\frac{Pr(Y=k|X=x)}{Pr(Y=k'|X=x)})=(\beta_{k0}-\beta_{k'0})+(\beta_{k1}-\beta_{k'1})x_1+\cdots+(\beta_{kp}-\beta_{k'p})x_p$$

4.4 用于分类的生成模型

逻辑斯蒂回归是一种相对直接的分类模型，在某些情况下，它的效果可能不好，例如：

- 两类之间存在明显的分离情况，此时逻辑斯蒂回归模型的参数会异常不稳定
- 预测变量 X 在每个类别中的分布近似正态分布，且样本量较小。利用非逻辑斯蒂回归更准确
- 部分多分类问题

考虑用**贝叶斯公式**求当 $X=x$ 时，Y 属于类别 k 的概率：

$$Pr(Y=k|X=x)=p_k(x)=\frac{\pi_k f_k(x)}{\sum_{l=1}^{K}\pi_l f_l(k)}$$

其中 π_k 表示**先验概率**，即 $Y=k$ 的概率 (第 k 类占比 $P(Y=k)$；$f_k(x)$ 表示在第 k 类中，$X=x$ 的概率，也即 X 在 $Y=k$ 上的**密度函数**，满足 $f_k(x)=Pr(X|Y=k)$；称 $p_k(x)$ 为**后验概率**

在一个数据集中，估计 π_k 是很容易的；然而，确定 $f_k(x)$ 却困难得多。如果我们能找到一个近似的方法估计 $f_k(x)$，我们就能快速获取后验概率

4.4.1 线性判别分析

4.4.1.1 单特征线性判别分析

首先假设 $f_k(x)$ 是 **正态分布 (*normal*)** 的，并在单维度设定下，正态密度：

$$f_k(x)=\frac{1}{\sqrt{2\pi}\sigma_k}\exp(-\frac{1}{2\sigma_k^2}(x-\mu_k)^2)$$

> 特征 x 按高斯分布

其中，μ_k 是第 k 类的均值，σ_k^2 是第 k 类的方差

> 符号 exp 表示指数函数 e 的幂

进一步假设 $\sigma_1^2=\sigma_2^2=\cdots=\sigma_K^2$，也就是说 K 类都是共方差的。用 σ^2 表示并代入之前的 $p_k(x)$ 的贝叶斯公式中计算，有：

$$p_k(x)=\frac{\pi_k\frac{1}{\sqrt{2\pi}\sigma}exp(-\frac{1}{2\sigma^2}(x-\mu_k)^2)}{\sum_{l=1}^{K}\pi_l\frac{1}{\sqrt{2\pi}\sigma}exp(-\frac{1}{2\sigma^2}(x-\mu_l)^2)}$$

贝叶斯分类器将 $X=x$ 分类到可能概率最大的一个类别中去。对上式取对数，我们得到一个简化的**判别函数**：

$$\delta_k(x)=x\frac{\mu_k}{\sigma^2}-\frac{\mu_k^2}{2\sigma^2}+log(\pi_k)$$

也就是找到最大的 $\delta_k(x)$，使得 $X=x$ 获得分类 k。例如在两个分类等可能出现时，贝叶斯边界即为 $\sigma_1(x)=\sigma_2(x)$ 处的预测变量 x 取值，此时：

$$x=\frac{\mu_1^2-\mu_2^2}{2(\mu_1-\mu_2)}=\frac{\mu_1+\mu_2}{2}$$

线性判别分析 (*linear discriminant analusis, LDA*) 通过将 π_k、μ_k 和 σ^2 的估计值代入简化后的判别函数来进行概率的计算。参数估计的公式为：

$$\hat{\mu}_k = \frac{1}{n_k} \sum_{i:y_i=k} x_i$$

$$\hat{\sigma}^2 = \frac{1}{n-K} \sum_{k=1}^{K} \sum_{i:y_i=k} (x_i - \hat{\mu}_k)^2$$

其中 $\sum_{i:y_i=k} x_i$ 表示对所有属于第 k 类的样本点求和，n 为训练观测数据的总数，n_k 表示在训练数据中的 k 类个数，μ_k 表示训练数据中第 k 类的均值，σ^2 表示 K 个样本变量的加权方差

有时候 π_k 可以直接获取，否则需要估计：

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

> 也就是第 k 特征的出现频率

将所有估计好的参数代入判断函数：

$$\hat{\delta}_k(x) = x\frac{\hat{\mu}_k}{\hat{\sigma}^2} - \frac{\hat{\mu}_k}{2\hat{\sigma}^2} + \log(\hat{\pi}_k)$$

则我们将 $X = x$ 判断为第 k 类的依据是 $\hat{\delta}_k(x)$ 最大

> 线性判别分析的"线性"即是来源于判别函数是 x 的线性函数

注意：贝叶斯决策边界就是两种分类在后验概率下相等的区域；训练数据集的决策边界是找到这样的一条线，使得不同的分类分隔开。这二者是不同的

4.4.1.2 多特征线性判别分析

多元高斯分布 (*multivariate Gaussian*)：若一个随机向量 $\mathbf{X} = (X_1, X_2, \ldots, X_p)$ 中的每个元素都满足正态分布，且协方差矩阵相同，则这个随机向量满足多元高斯分布。高斯分布的形状称为**钟型**；当预测变量是相关的，或者方差不相等，钟型将被扭曲

> 满足二元高斯分布的图像从 X_1 或 X_2 轴切割得到的截面形状具有一维正态分布的形状

为了说明一个 p 维随机向量是正态分布的，我们记作：

$$\mathbf{X} \backsim N(\mu, \Sigma)$$

其中 $\mu = E(X)$ 是向量 X 的期望，$\Sigma = Cov(X)$ 是 p 个元素的 $p \times p$ 协方差矩阵

多元高斯分布的密度函数：

$$f(x) = \frac{1}{(2\pi)^{p/2}|\Sigma|^{1/2}} \exp(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(X-\mu))$$

LDA 分类器假设第 k 类的观测值都来自于多元高斯分布 $N(\mu_k, \Sigma)$。其中 μ_k 是这类的均值向量，Σ 是所有 K 类的共有的协方差矩阵

用类似的数学运算，我们可以由贝叶斯公式推到得到多元线性判别的简化判断函数：

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

> 这可以看做是一个向量/矩阵版的判别公式

当它最大时进行分类，这实际上是单特征判断函数的向量形式。计算贝叶斯决策边界满足 $\delta_k(x) = \delta_l(x)$，即：

$$x^T\Sigma^{-1}\mu_k - \frac{1}{2}\mu_k^T\Sigma^{-1}\mu_k = x^T\Sigma^{-1}\mu_l - \frac{1}{2}\mu_l^T\Sigma^{-1}\mu_l$$

通过判断函数我们发现，$\delta_k(x)$ 是 x 中元素的线性组合，这也是 LDA 中线性的意思。

用模型参数 p 和样本数 n 的比值 $\frac{p}{n}$ 来表示过拟合程度，容易发现这个值越小说明越不可能出现过拟合

检验一个分类器是否分类得好，不应该只关注它的错误率。事实上，有些分类器的错误率不比一个零分类器好，这是因为数据集本身的“错误率”就很低

> 对于一个**零分类器 (*null*)**，即不论什么预测向量 X 都将其归为一类，这样的分类器导致的错误率跟我们模型的错误率进行比对，若相差不大，我们需要考虑模型的优化或数据的选取问题

通常，一个二值分类器 (*Binary Classifier*) 会导致两类错误，即**取伪**或**去真**。使用**混淆矩阵 (*confusion matrix*)** 直观地显示有关错误的信息，我们通常期望其中一类错误尽可能小，因此可以降低其中一个概率取值的判断阈值

> 在医学或生物学中，**敏感性 (*sensitivity*)** 指的是很少错过真正有病的病例，即去真少；**特异性 (*specificity*)** 指的是很少将健康人误诊为病例，即取伪少

在实际问题的应用上，由于 LDA 会将所有的错误率按尽可能低的方式逼近贝叶斯分类器，因此可能对两类错误的取舍不符合实际问题的预期。所以，我们有必要开发更符合实际问题的分类器，例如改变取舍**阈值**。一般来说，增大阈值会减小敏感性，减小阈值会减小特异性

我们用**ROC 曲线 (*ROC curve*)** 来表示分类器性能的权衡。ROC 曲线的横轴表示取伪率，纵轴表示取真率。理想分类器的 ROC 曲线特别靠近左上角 $(0, 1)$ 点，面积 AUC 越大表示分类性能越好，当 $AUC = 0.5$ 时相当于随机分类，小于 0.5 时分类效果不如随机分类

> 比较模型性能，可以通过绘制模型的 ROC 曲线，然后计算 AUC 值

4.4.3 二次判别分析

与 LDA 不同，**二次判别分析 (*Quadratic Discriminant Analysis*)** 考虑每一个类都有一个协方差矩阵，每个协方差矩阵不一定相同，也就是类内数据散布不同。来自第 k 类的观测满足 $X \backsim N(\mu_k, \Sigma_k)$，其中 Σ_k 表示第 k 类的协方差矩阵。此时判别函数：

$$\begin{aligned}\delta_k(x) &= -\frac{1}{2}(x-\mu_k)^T\Sigma_k^{-1}(x-\mu_k) - \frac{1}{2}\log|\Sigma_k| + \log\pi_k \\ &= -\frac{1}{2}x^T\Sigma_k^{-1}x + x^T\Sigma_k^{-1}\mu_k - \frac{1}{2}\mu_k^T\Sigma_k^{-1}\mu_k - \frac{1}{2}|\Sigma_k| + \log\pi_k\end{aligned}$$

> QDA 比 LDA 更加灵活，适合类别之间数据散布不同的情况

我们需要估计 Σ_k、μ_k、π_k

对于不同的数据集，LDA 和 QDA 的适用不同。如果贝叶斯边界为线性的，LDA 更能反映实际情况；否则，QDA 更加反应实际情况

> 注意 δ_k 是一个 x 的二次函数，因此得名QDA

LDA 与 QDA 的选择取决于方差-偏差权衡。LDA 拥有更低的方差，因此不如 QDA 灵活。因此，建议在训练集较小的时候使用 LDA，训练集较大时使用 QDA，防止过拟合问题

4.4.4 朴素贝叶斯

对比 LDA 和 QDA 作的假设：

- LDA 假设 f_k 是一个多元正态随机变量的密度函数，每个类共享 μ 和协方差矩阵 Σ
- QDA 假设对于多元正态随机变量的密度函数 f_k，每一类有它的 μ_k 和 Σ_k

朴素贝叶斯 (*naive Bayes*) 分类器作出了另一个假设：对于每个 k 类，p 个预测变量都是独立的。从数学上说，这表示：

$$f_k(x) = f_{k1}(x_1) \times f_{k2}(x_2) \times \cdots \times f_{kp}(x_p)$$

这里 f_{kj} 是第 k 类观测值中第 j 个预测因子的密度函数

> 这相当于，在样本类别为 k 时，观察到一个样本 $\mathbf{x}$，包含 m 个属性的条件概率 $P(x_1, x_2, \ldots, x_m|k)$ 被假设为条件独立情况下的 $\prod_{i=1}^{m} P(x_i|k)$

一个观测向量 X 内部的协变量 x_i 之间的相互关系通常很难判断，通过朴素贝叶斯的假设我们很快地消除了它们之间的相互关系。尽管在事实上常常并非如此，但是在观测值数 n 相对于观测变量 p 很小时，这种假设也很有用

边缘分布 (*Marginal Distribution*) 是指在处理多维随机变量时，只关注部分变量的概率分布；**联合分布 (*Joint distribution*)** 用于计算随机变量组合取特定值的概率，在两个随机变量独立时有 $f(x,y) = f_X(x)f_Y(y)$，朴素贝叶斯使这两类计算变得很容易。特别是在 n 相对于 p 不够大时，这个假设可以使我们有效地估计联合分布

由朴素贝叶斯假设：

$$Pr(Y = k|X = x) = \frac{\pi_k \times f_{k1}(x_1) \times f_{k2}(x_2) \times \cdots \times f_{kp}(x_p)}{\sum_{l=1}^{K} \pi_l \times f_{l1}(x_1) \times f_{k2}(x_2) \times \cdots \times f_{kp}(x_p)}$$

现在，我们的目标是估计每个一维密度函数。下面有几种选择：

- 对于每个类中定量的随机变量 X_j，直接认为每个一维密度函数服从各自的正态分布，即 $X_j|Y = k \sim N(\mu_{jk}, \sigma_{jk}^2)$

> $f_{kj}(x_j)$ 表示在 k 类别中，特征 X_j (特征向量中的一个) 取值为 x_j 的概率密度

- 对于每个类中定量的随机变量 X_j，考虑用非参数方法估计这些密度函数 (分布律)。例如，画出直方图来估计特征取值 x_j，或者考虑用**核密度估计 (kernel destity esimator)**

> 直方图估计法类似于用频率估计概率，只不过变量的取值是连续的

- 对于每个类中定性的随机变量 X_j，我们直接统计每个类中 x_j 的出现频率，用频率估计概率

朴素贝叶斯取得的分类效果不一定胜过 LDA，这取决于朴素贝叶斯减少的方差相对于偏差来说不一定是值得的。选择朴素贝叶斯的情况是在 p 相对大 n 相对小的情况下，这时候，减少方差才是比较重要的

4.5 关于分类器的比较

4.5.1 分析比较

通过研究对数赔率：

$$\log(\frac{P(Y=k|X=x)}{P(Y=K|X=x)})$$

来研究不同分类器的特性

对于 LDA 模型，通过计算可以得到：

$$\log(\frac{P(Y=k|X=x)}{P(Y=K|X=x)})=a_k+\sum_{j=1}^{p}b_{kj}x_j$$

其中，a_k 满足：

$$a_k=\log(\frac{\pi_k}{\pi_K})-\frac{1}{2}(\mu_k+\mu_K)^T\Sigma^{-1}(\mu_k-\mu_K)$$

b_{kj} 满足：

$$b_{kj}=\Sigma^{-1}(\mu_k-\mu_K)$$

> a_k 是一个常数项，b_{kj} 可以认为是第 j 个特征 x_j 对该对数几率的权重

这说明 LDA 模型就像逻辑斯蒂回归模型一样，得到的关于后验概率的对数赔率是 x 的一个线性组合

同样的方法可以推出关于 QDA 的对数赔率：

$$\log(\frac{P(Y=k|X=x)}{P(Y=K|X=x)})=a_k+\sum_{j=1}^{p}b_{kj}x_j+\sum_{j=1}^{p}\sum_{l=1}^{p}c_{kjl}x_jx_l$$

其中的 a_k,b_{kj},c_{kjl} 都是关于 $\pi_k,\pi_K,\mu_k,\mu_K,\Sigma_k$ 和 Σ_K 的函数。可以看出，QDA 的对数赔率是关于 x 的二次函数

推导朴素贝叶斯的对数赔率：

$$\log(\frac{P(Y=k|X=x)}{P(Y=K|X=x)})=a_k+\sum_{j=1}^{p}g_{kj}(x_j)$$

其中的 $a_k=\log(\frac{\pi_k}{\pi_K})$，$g_{kj}(x_j)=\log(\frac{f_{kj}(x_j)}{f_{Kj}(x_j)})$。这个形式也被称为**广义相加模型**

下面是一些结论：

- LDA 是 QDA 的特殊版本，即限制了每个分类都有相同的特征分布
- 任何具有线性决策边界的分类器都是朴素贝叶斯的一个特例，这意味着 LDA 是朴素贝叶斯的一个特例
- 如果在朴素贝叶斯中使用一维高斯分布对每个特征分布进行建模，可以发现朴素贝叶斯是 LDA 的一个特例，即特征之间的协方差阵时对角型矩阵
- 朴素贝叶斯是一个**加性模型**，在特征独立时优势明显；QDA 适用于捕捉特征之间的交互性

> 上面的分析指出，没有一种分类器占据绝对优势，而应该根据实际选取

逻辑斯蒂回归的对数赔率是：

$$\log(\frac{P(Y=k|X=x)}{P(Y=K|X=x)})=\beta_{k0}+\sum_{j=1}^{p}\beta_{kj}x_j$$

可以发现形式与 LDA 类似。二者之间的差别在于逻辑斯蒂回归的参数选择是为了满足极大似然函数，而 LDA 的选择是基于贝叶斯公式。我们期望在高斯分布假设成立时，LDA 优于逻辑斯蒂回归，否则逻辑斯蒂回归更优

KNN 模型相对于上面的模型来说有这样的特征：

- 完全没有参数，拥有非线性决策边界。因此在贝叶斯边界是非线性时表现得比 LDA 和逻辑斯蒂回归好
- KNN 需要大量的训练数据，即 $n>>p$。这是因为它倾向于减少偏差，同时产生大量方差
- 样本量小时，或者说 n 不大 p 不小时，QDA 比 KNN更优。
- 不同于逻辑斯蒂回归，KNN 是不会说明哪些预测因子是重要的 (无参数)

4.5.2 经验比较

为了比较这几种模型的适用情况，书中生成了 6 组不同的数据集用于检测。其中，前 3 种拥有线性决策边界，后三种拥有非线性决策边界，下面是比较的结论：

- 没有一种方法在任何情况下都占据绝对优势
- 当真实决策边界是线性的，LDA 和逻辑斯蒂回归表现得较好
- 当真实决策边界中度非线性，QDA 或朴素贝叶斯表现得较好
- 当真实决策边界非常复杂，KNN 等非参数方法更具有优势。尽管如此，也应当正确地选择平滑程度

在线性回归中使用的 X^2、X^3 等高次交互项也可以在分类器中使用。不过，要注意偏差-方差均衡

4.6 广义线性回归

在实际生活中，有一些预测变量 Y 的取值既不是定性的也不是定量的，例如比率、集合、高维度数据等。在这种情况下，使用线性回归方法或分类器方法都不能达到良好的效果，我们因此引入**广义线性回归 (Generalized Linear Models)**

本部分使用了“BikeShare”数据集

4.6.1 用线性回归拟合 Bikeshare

在用线性回归拟合 Bikeshare 数据集时出现了下面的问题：

- 负数预测值，不符合实际
- 过强的异方差性，即随机误差是关于随机变量的函数，不符合线性回归假设
- 连续的预测区间，不符合实际 (离散型)

使用对数变换，即令 $\log Y = \sum_{j=1}^{p} X_j\beta_j + \epsilon$ 可以减小异方差，但是这导致了解释上的困难 (每单位的变量改变对应的响应变量的变化不好解释)

4.6.2 泊松回归

泊松分布 (Poisson) 是一类关于离散型随机变量的分布，事件 X 发生 k 次的概率取值为：

$$P(X=k)=\frac{e^{-\lambda}\lambda^k}{k!}\quad k=0,1,2,\cdots$$

泊松分布的特点是均值和方差相等，即：

$$E(X)=D(X)=\lambda$$

这说明随着 $E(X)$ 的变大，$D(X)$ 会跟着变大

泊松分布常常用于对**可数量 (counts)** 的建模，因为泊松分布取非负值。

我们将一天中的 Biker 建模为满足某个参数为 λ 的泊松分布，这就是**泊松回归 (Poisson Regression)**。此处，λ 是变量 $\mathbf{X}=(X_1,X_2,\cdots,X_p)$ 的函数，记为：

$$\log(\lambda(X_1,X_2,\cdots,X_p))=\beta_0+\beta_1X_1+\beta_2X_2+\cdots+\beta_pX_p$$

等价于：

$$\lambda(X_1,X_2,\cdots,X_p)=e^{\beta_0+\beta_1X_1+\beta_2X_2+\cdots+\beta_pX_p}$$

其中的 β_i 是需要估计的参数

考虑用极大似然估计法估计这些参数，有：

$$l(\beta_0,\beta_1,\cdots,\beta_p)=\prod_{i=1}^{n}\frac{e^{-\lambda(x_i)}\lambda(x_i)^{y_i}}{y_i!}$$

其中的 $\lambda(x_i)=e^{\beta_0+\beta_1x_{i1}+\beta_2x_{i2}+\cdots+\beta_px_{ip}}$。我们找到一组 β_i 的取值使得上式最大

下面是关于泊松回归的特点：

- 参数为了解释泊松回归模型的参数，我们需要关注 λ 关于特征向量的函数。不同的特征向量变动导致不同的 λ 取值，使我们可以对概率作出预测
- 均值-方差关系泊松回归假定均值和方差会同时发生变动，而回归分析则假设均值和方差保持不变。因此，在某些情况下，泊松回归能正确地捕捉均值和方差的变动关系
- 非负值拟合泊松回归得到的结果都是非负值，而线性回归可能导致负值出现

4.6.3 更具普遍性的广义回归模型

我们讨论过的三种回归模型，即线性回归、逻辑斯蒂回归和泊松回归有一些相同的特性：

1. 响应变量 Y 都被假设服从某个分布。线性回归假设 Y 服从正态分布，逻辑斯蒂回归假设 Y 服从伯努利分布 (二项分布)，泊松回归假设 Y 服从泊松分布
2. 响应变量 Y 的均值都被建模为了特征向量的一个函数。线性回归中 $E(Y|X_1,X_2,\cdots X_p)=\beta_0+\beta_1X_1+\cdots+\beta_pX_p$，逻辑斯蒂回归中 $E(Y=1|X_1,X_2,\cdots X_p)=\frac{e^{\beta_0+\beta_1X_1+\cdots+\beta_pX_p}}{1+e^{\beta_0+\beta_1X_1+\cdots+\beta_pX_p}}$，泊松回归中 $E(Y|X_1,\cdots,X_p)=e^{\beta_0+\beta_1X_1+\cdots+\beta_pX_p}$

对于性质 2，我们抽象出这些表达式都使用了所谓的**连接函数 (link function)** η 来建模。通过使用链接函数，能够将 $E(Y|X_1,\cdots,X_p)$ 转换为特征向量的一个线性函数，也就是：

$$\eta(E(Y|X_1,\cdots,X_p))=\beta_0+\beta_1X_1+\cdots+\beta_pX_p$$

上面三种模型的链接函数分别是：

- 线性回归 $\eta(\mu)=\mu$
- 逻辑斯蒂回归 $\eta(\mu)=\log(\mu/(1-\mu))$
- 泊松回归 $\eta(\mu)=\log(\mu)$

高斯分布、伯努利分布和泊松分布都属于**指数分布族 (exponential family)**。此外，属于指数分布族的分布还有**指数分布 (expoential distribution)**、Γ **分布 (Gamma distribution)** 和 **负二项分布（negative binomial distribution)**。总的来说，属于指数分布族的分布都可以用广义回归模型来建模，即找到一个估计函数 η 来估计 $E(Y|X_1,\cdots,X_p)$。我们上面提到的三种模型都是广义回归模型的例子

第五章 重采样方法

重采样方法 (Resampling methods) 是现代统计学中不可缺少的方法。为了获取更多的样本信息，我们对样本进行重复采样，得到更多的数据集来训练或评估模型。这一章介绍的重采样方法为交叉检验和自助检验

5.1 交叉检验

在第二章中介绍的测试误差和训练误差分别是使用测试数据集和训练数据集得出的。当我们没有足够大的数据集以划分训练集和测试集时，此时就无法直接用测试集来计算测试误差，只能通过训练集的训练误差来估计。然而，训练误差常常低估测试误差。因此我们必须考虑用别的方法来估计测试误差

5.1.1 验证集方法

验证集方法 (the validation set approach) 是一种用来估计测试误差的方法。通过随机划分测试集和**验证集 (Validation Set)** 来分别训练和评估模型

验证集方法的优点在于简单易用，缺点是随机划分的测试集和验证集会导致每次得到的测试误差估计都不一样，且有许多数据未被用于训练模型 (训练数据越多，模型越准确) 造成浪费，容易高估测试误差，在数据集小的时候不实用

> 验证集方法可以用于大数据的模型。此时，需要将数据集划分为训练集、验证集和测试集三个集合，用训练集拟合模型，再用验证集评估参数选择方案，最后用测试集评估选好参数的模型效果

5.1.2 留一交叉检验

留一交叉检验 (Leave-one-out Cross Validation, LOOCV) 的思想近似于验证集方法，但能克服其缺点

留一交叉检验的步骤如下：

- 划分检验集与训练集。特别地，检验集仅包含一个观测值，训练集包含 $n-1$ 个观测值
- 用训练集拟合模型，并用来估计 $MSE_1=(y_1-\hat{y}_1)^2$，这个估计近似于 MSE_1 的无偏估计

  > 注意不同 MSE 值不同

- 重复步骤 1 和 2，不同的是更新检验集，直到所有的观测数据都被使用过一次
- 计算估计的总体 MSE

$$CV_{(n)}=\frac{1}{n}\sum_{i=1}^{n}MSE_i$$

留一交叉检验相比于检验集方法的优点在于：

- 训练模型的偏差小，因为训练集大
- 不会因为数据集的随机分割而导致不同的结果，重复试验结果相同

留一交叉检验在 n 大的时候计算量很大，因为留一交叉检验需要计算 n 次，然而用下面的等式可以用一次计算得出 $CV_(n)$

在最小二乘回归或多项式回归中，下面的等式成立：

$$CV_{(n)} = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - \hat{y}_i}{1-h_i}\right)^2$$

其中 h_i 是之前定义的杠杆值，计算公式为：

$$h_i = \frac{1}{n} + \frac{(x_i - \overline{x})^2}{\sum_{i'=1}^{n}(x_{i'} - \overline{x})^2}$$

> 这个公式用 h_i 杠杆来影响残差，使得等式成立

LOOCV 是一种非常通用的方法，可以与任何一种预测模型结合使用，例如逻辑回归和 LDA。需要注意的是，在这些模型中，上面的杠杆值估计公式不成立，必须重新拟合 n 次模型

5.1.3 K-fold 交叉检验

一种替代 LOOCV 的检验方式是**K-fold 交叉检验 (K-fold Cross Validation)** 。这种方法非常类似 LOOCV，不同的是他将数据集划分为 k 个**折叠 (fold)**，将其中的一个折叠用于检验集，另外 $k-1$ 个用于训练模型，重复 k 次后有：

$$CV_{(k)} = \frac{1}{k}\sum_{i=1}^{k} MSE_i$$

不难发现，LOOCV 是 K-fold 交叉检验的一种特例。K-fold 交叉检验的优势不仅体现在计算次数减少上，还有可能带来非计算的方差-偏差权衡

K-fold 交叉检验得到的测试误差估计仍然是变动的，不过相较于 LOOCV，这种变动在合适的 k 上变化不大

用已知分布的模拟数据检验 LOOCV 和 $K = 10$ 的 K-fold 交叉检验可发现，二者的区别并不太大

有时我们的目的并不是观察不同参数模型拟合过程中测试 MSE 的变动，而是寻找拟合模型的最小 MSE 点，这是也可以借助交叉检验，因为上面的验证中，交叉检验预测的最低 MSE 点与实际差距不大

5.1.4 K-fold 交叉检验的偏差-方差权衡

抛开计算问题，一个 K-fold 交叉检验的潜在优势是偏差-方差权衡

用 LOOCV 训练的模型虽然偏差低于 K-fold 交叉检验，但是方差往往是前者高于后者。这是因为 LOOCV 虽然拟合了多个模型，但是这些模型的相关性非常高。数学上的证明指出，这样得到的测试 MSE 往往具有更大的方差

为了偏差-方差权衡，我们一般使用 $k = 5$ 或 $k = 10$ 的 K-fold 交叉检验

5.1.5 分类器中的交叉检验

在分类器中，我们关心的一个类似的指标是误分率。例如，此时 LOOCV 计算的是：

$$CV_{(n)} = \frac{1}{n}\sum_{i=1}^{n}\mathrm{ERR}_i$$

其中，$\mathrm{Err}_i = I(y_i \neq \hat{y}_i)$。K-fold 交叉检验的定义与之类似

很容易发现随着模型灵活度的增加，训练误差的趋势是一直减小的，这是因为模型会尽可能地拟合每一个使用的数据。然而，用交叉检验估计的测试误差则呈一个 U 型曲线，表示出过拟合的现象。进一步的模拟表明，这个 U 型曲线是对真实测试误差的很好的估计

> 值得注意的是，交叉检验对测试误差的最小值估计也非常接近

5.2 自助法

自助法 (bootstrap) 是一种广泛而常用的重采样方法，常用于统计量分布的估计和参数置信区间的计算，也可以用于评估模型性能

下面举例说明自助法的应用。假设需要投资 X 和 Y 两类商品 (X, Y 表示收益率)，我们将部分资金 α 投资于 X，另一部分 $1-\alpha$ 投资于 Y。现在，我们希望确定一个组合来最小化风险 $Var(\alpha X + (1-\alpha)Y$，也即是最小化：

$$\alpha = \frac{\sigma_Y^2 - \sigma_{XY}}{\sigma_X^2 + \sigma_Y^2 - 2\sigma_{XY}}$$

实际中，σ 是未知的，我们通过过去的收益率来估计 $\hat{\sigma}$，得到：

$$\hat{\alpha} = \frac{\hat{\sigma}_Y^2 - \hat{\sigma}_{XY}}{\hat{\sigma}_X^2 + \hat{\sigma}_Y^2 - 2\hat{\sigma}_{XY}}$$

为了估计 $\hat{\sigma}$，我们考虑生成用数据集生成 100 个 X, Y 数据对，估计一次 $\hat{\alpha}$，将这个过程重复 1000 次，从而得到 1000 个 $\hat{\alpha}_i$ 的估计，用他们的均值来估计总体 α：

$$\overline{\alpha} = \frac{1}{n}\sum_{i=1}^{n}\hat{\alpha}_i$$

我们可以发现这个模拟值与真实的 α 非常接近，标准差也非常小

当我们只有一个数据集时，使用自助法可以生成许多用于估计总体参数的样本，而避免了继续向总体抽取样本。自助法通过有放回地向样本抽样得到多个新样本，估计参数的均值完成对总体估计。实践中，这个估计非常的好

> 这里作的假设是样本分布可以代表总体分布
> 一般来说，抽取新样本的次数为 1000 或 2000 次

第六章 线性模型选择和正则化

我们在线性回归中用到的模型：

$$Y = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p + \epsilon$$

包含 p 个特征值和一个随机误差项 ϵ。当 $n >> p$ 时，最小二乘法对这个模型系数的估计拥有较低的偏差和方差，表现良好

然而当 n 仅仅大于 p，甚至 $n < p$ 时，最小二乘法的方差会变得非常大，导致模型效果变差。此时，我们必须考虑用其他的方法，以提高模型的**预测精度 (prediction accuracy)** 和 **可解释性 (interpretability)**

> 其他的一些线性模型，可以实现自动的**特征选择 (feature selection)** 或 **变量选择 (variable selection)**

本章主要介绍三类重要的模型：

- **子集选择 (Subset Selection)** 找到系数的较优子集作为模型的系数估计
- **收缩 (Shrinkage)** 从拟合所有系数收缩到部分系数进行模型复杂度的降低，也可以用于变量选择
- **降维 (Dimension Reduction)** 将 p 个特征值投影到低维空间，使得 $M < p$ 个投影特征用于估计系数

6.1 子集选择

6.1.1 最优子集选择

在**最优子集选择 (best subset selection)** 策略中，对于 p 个特征值，我们拟合 $\sum_{i=1}^{p} C_p^i = 2^p$ 个模型，并找到最优模型作为我们的参数选择

在子集选择过程中，我们考虑 $n = 1, 2, \cdots, p$ 个特征的模型，并在每一类的模型中，从 p 个参数中选取 k 个来拟合此类模型；对于每一类模型，我们得到该类最优模型；选择完毕后，我们得到 p 个模型，最后从这 p +1 个模型 (含 null 模型)中得到最优模型

> 每类模型的判断标准为 RSS 或 R^2，$p+1$ 个模型的判断标准为交叉检验、$C_p, BIC,$ adjusted R^2 等

最优子集选择容易导致选择次数过多的问题，从而无法在限定时间内得到结果；而且，过大的子集选择范围，可能导致模型较大的方差

6.1.2 逐项选择

前向选择 (Forward Stepwise Selection) 是一种替代最优子集选择的策略。从 null 模型出发，每次添加一个最优特征值，直至选完所有特征值，再对 $p+1$ 个模型选择得到最优模型。不难得出前向选择的选择次数为 $1 + \frac{p(p+1)}{2}$

虽然实践中前向选择的表现不差，但由于缩小选择子集的缘故，前向选择不能保证获得最优解，因为前向选择可能导致某类模型变量的遗漏

在高维数据中，前项选择也是可以利用的，但只能构筑 M_{n-1} 即特征值为 $n-1$ 个的模型，原因是最小二乘法的唯一解性

后向选择 (Backward stepwise selection) 也是一种替代最优子集选择的方法。与前向选择相反，该算法从包含 p 个特征值的模型**全模型 (full model)** 出发，每次减少一个特征值，并在此类中留下最优模型，直到 null 模型为止，并选择 $p+1$ 个模型作为最优模型

后向选择不能在 $p > n$ 是使用，只能在 $n \geq p$ 时使用，因为需要拟合全模型

混合选择 (Hybrid Selection) 在前向选择的同时，可能删除某些变量回退到后向选择，直至获取最优模型。这个算法保留了计算优势，同时紧密地模仿最优子集选择

6.1.3 最优模型选择

通过上述方法得到的模型都会包含全模型。全模型的偏差最小，但方差不一定最小，因此不适合用 R^2 来比较不同特征值数量的模型

比较不同特征值数量的模型，常用的方法有：

1. 直接估计测试误差，利用交叉检验等方法
2. 对训练误差进行调整，从而间接估计训练误差

6.1.3.1 调整训练误差

对于一个最小二乘法拟合的包括 d 个特征值的模型，$\mathbf{C}_p$ 估计训练误差：

$$C_p = \frac{1}{n}(RSS + 2d\hat{\sigma}^2)$$

其中的 $\hat{\sigma}^2$ 是对回归方程中随机误差 ϵ 的估计。C_p 通过添加 $wd\hat{\sigma}^2$ 来放大训练误差，注意到随着特征值的增多，C_p 也跟着增大

> C_p 有两个公式，这两者计算的 C_p 正相关

可以证明 C_p 是 MSE 的一个无偏估计，因此用最小的 C_p 可以判断一个最优的拟合模型

赤池信息准则 (AIC) 是针对极大似然法估计的一大类模型定义的。在假设 ϵ 满足高斯分布后，极大似然估计与最小二乘估计结构相同，此时：

$$AIC = \frac{1}{n}(RSS + 2d\hat{\sigma}^2)$$

这里省略了一些常数，可知 AIC 与 C_p 成正比

> AIC 有两个公式

贝叶斯信息准则 (BIC) 是根据贝叶斯公式推导的信息准则，形式接近于 C_p 和 AIC。对于 p 个特征值的最小二乘模型：

$$BIC = \frac{1}{n}(RSS + \log(n)d\hat{\sigma}^2)$$

根据公式可知 BIC 对复杂模型惩罚力度大，因此倾向于选择参数较少的模型 (相对于 AIC 和 C_p)

调整后 R^2 (adjusted R^2) 是一种留下的选择模型的准则。根据 R^2 的公式，当模型变量越多时，R^2 越大。对于 d 个特征值拟合的最小二乘模型：

$$Adjusted\ R^2 = 1 - \frac{RSS/(n-d-1)}{TSS(n-1)}$$

根据公式，当模型中变量增多时，R^2_{adj} 可能增大也可能减小，取决与变量增多是否能增大模型的拟合效果

> 这一推断取决于 $RSS/(n-d-1)$ 的变化

调整后 R^2 背后的直觉是当所有正确的变量都被包含在模型中，额外添加噪声将导致 R^2_{adj} 下降 (少量 RSS 下降伴随着 $n-d-1$ 增加)。因此 R^2_{adj} 最大的模型理论上将只包含正确变量

需要注意的是：

1. 理论上 R^2_{adj} 不如其他三个准则那么可解释
2. 其他三个准则有复杂的公式推导，基于渐进估计 $n \to \infty$
3. AIC 和 BIC 可以应用于更加复杂的模型，有不同的公式形式

6.1.3.2 交叉检验方法

交叉检验方法可以替代上面的间接估计对训练误差作出直接估计，且对模型作出的假设更少，能适应更广泛的模型

值得注意的是，当使用交叉验证时，选择算法中的模型 M_k 的序列对于每一个训练折叠都是单独确定的，并且对于每一个模型大小 k，验证误差在所有折叠中都是平均的。这意味着，例如在最佳子集回归中，M_k，即大小为 k 的最佳子集，可以在不同的折叠中有所不同。一旦选择了最佳尺寸 k，我们在全数据集上找到了该尺寸的最佳模型。

> k 表示选取的特征数量，当 $k < m$，m 表示特征总数时，一定能选出多种不同的特征集合作为全特征的子集

对于不同的 fold，我们得到每个模型的估计训练误差一定是不同的，但可能相差不多。在这种情况下，我们计算交叉检验的最优模型测试误差，并在它的一个标准差内，选择最简单的模型，这个方法叫做 $1SE$ **规则 (one standard error rule)**，鼓励选择变量少的模型

> 简单理解为选择在最优估计测试误差一个标准差内的最简模型

6.2 收缩方法

在 6.1 节讨论的方法通过选择最小二乘回归你和的子集来选择最优的特征值，下面介绍的**收缩方法 (shrinkage methods)** 将通过拟合所有的特征值，再通过收缩去除某些特征值来选择模型

6.2.1 岭回归

最小二乘回归试图估计一组 $\beta_0, \beta_1, \cdots, \beta_p$，使得：

$$RSS = \sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2$$

取得最小值。**岭回归 (Ridge Regression)** 的策略与此类似，它最小化：

$$RSS = \sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 + \lambda\sum_{j=1}^{p}\beta_j^2 = RSS + \lambda\sum_{j=1}^{p}\beta_j^2$$

来达到选择变量目的，其中 $\lambda \geq 0$ 是一个**调谐变量 (tuning parameter)**

注意到岭回归通过引入 $\lambda\sum_{j=1}^{p}\beta_j^2$ 来寻找回归系数。事实上 $\lambda\sum_{j=1}^{p}\beta_j^2$ 是一个**收缩惩罚 (shrinkage penalty)**，他在 $\beta_0, \beta_1, \cdots, \beta_p$ 趋近于 0 时很小，因此岭回归倾向于将回归系数估小

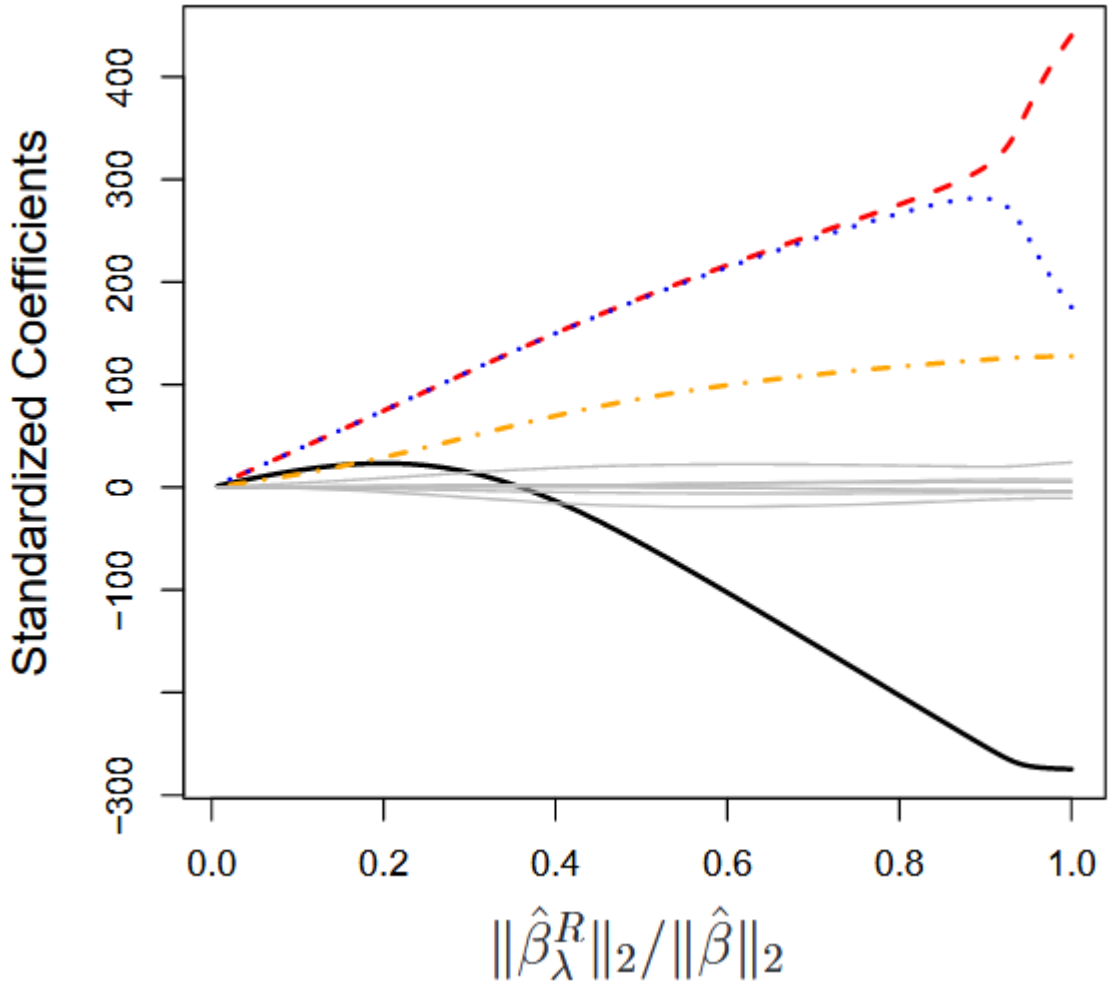
当 $\lambda = 0$ 时，岭回归退化到最小二乘回归，此时惩罚项不起作用；当 $\lambda \to \infty$ 时，惩罚项将回归系数控制到趋近于 0。不同于最小二乘回归，岭回归在不同的 λ 下都会产生不同的回归系数估计，因此选择一个适当的 λ 非常重要

我们不希望岭回归对截距项进行惩罚，因为它是 $\mathbf{X} = 0$ 时均值的估计。当 X 已经中心化为 0 时，估计的截距为 $\hat{\beta}_0 = \overline{y} = \sum_{i=1}^{n} y_i/n$

> $\lambda \to \infty$ 时对应的实际上是零模型

---

l_2 **范数 (l_2 norm)** 是用于衡量一个向量与原点距离的值，计算为 $||\beta||_2 = \sqrt{\sum_j^p = \beta_j^2}$。通过计算 $||\hat{\beta}_\lambda^R||_2/||\hat{\beta}||_2$ 值的变化，我们可以很直观的观察 λ 增大对系数估计的影响



与 OLS 模型不同，岭回归不是**等尺度变化 (scale equivariant)** 的。若某个特征值的单位发生变化导致特征值本身数值发生缩放，则岭回归估计的系数将会发生复杂的变化，原因在于惩罚项中 $\hat{\beta}$ 是二次的形式，因此有必要对特征值的观测值进行标准化：

$$x_{ij} = \frac{x_{ij}}{\sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_{ij} - \overline{x}_j)^2}}$$

以保证数据在同一尺度

岭回归的计算量相比于最小二乘回归也非常有优势。特别地，对于一个确定的 λ，岭回归只得到一个模型 (相比最小二乘回归每 p 个特征值需要你和多个模型)

6.2.2 拉索回归

岭回归有一个明显的缺点是模型总是包括所有的特征值，这虽然不会显著降低预测精度，但带来了解释上的困难

为了克服这个缺点，我们引入**拉索回归 (lasso regression)**，拉索回归试图通过：

$$\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 + \lambda\sum_{j=1}^{p}|\beta_j| = RSS + \lambda\sum_{j=1}^{p}|\beta_j|$$

获取回归系数的预测值 $\hat{\beta}_{\lambda}^{L}$。与岭回归不同之处在于拉索回归引入的乘法项采用的是 l_1 惩罚,，计算方式为 $||\beta||_1 = \sum|\beta_j|$

拉索回归的惩罚项在 $n \to \infty$ 时会将某些特征值控制为 0，实际上执行了参数选择的效果

拉索回归和岭回归可以用下面的两个式子联系起来：

$$min\left\{\sum_{i=1}^{n}\left(y_i-\beta_0-\sum_{j=1}^{p}\beta_j x_{ij}\right)^2\right\}\text{subject to}\sum_{j=1}^{p}|\beta_j|\leq s min\left\{\sum_{i=1}^{n}\left(y_i-\beta_0-\sum_{j=1}^{p}\beta_j x_{ij}\right)^2\right\}\text{subject}$$

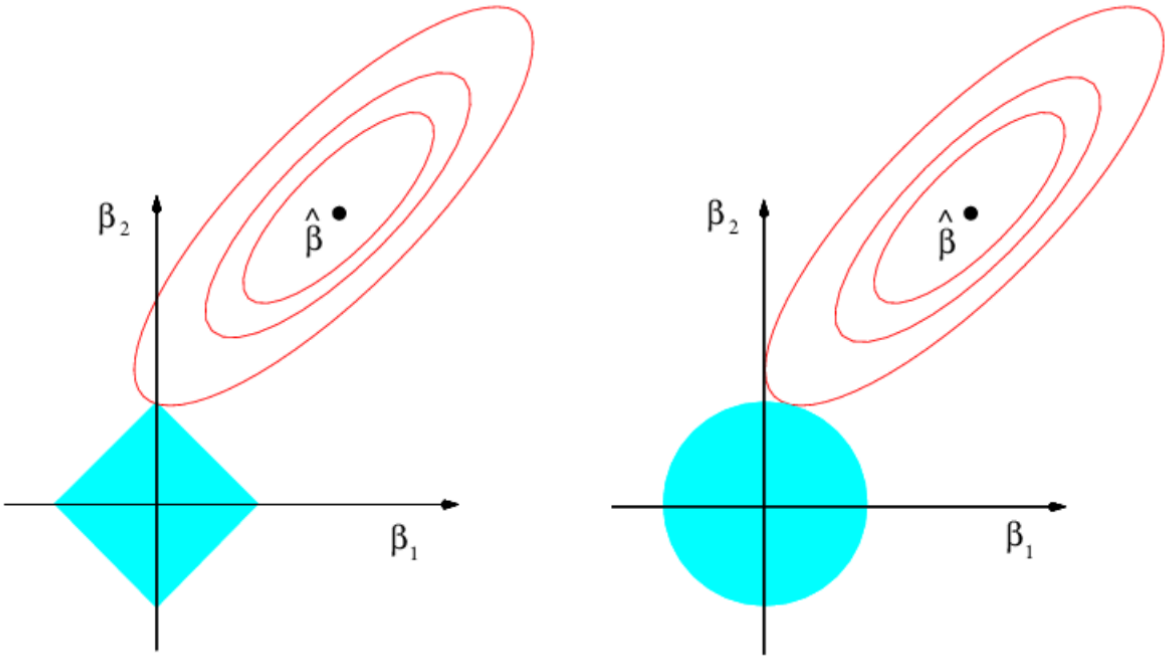
换句话说，给定一个值 s 用以约束最小二乘回归，使得系数的某种组合必须满足约束值 s

上面两个式子将拉索回归和岭回归联系起来了。同样地，我们考虑下面的式子：

$$min\left\{\sum_{i=1}^{n}\left(y_i-\beta_0-\sum_{j=1}^{p}\beta_j x_{ij}\right)^2\right\}\text{subject to}\sum_{j=1}^{p}I(\beta_j\neq 0)\leq s$$

其中 I 是一个指示变量，表示系数 $\beta_j \neq 0$ (此时值为 1)。这实际上是子集选择的数学表示

当 p 很大时，子集选择很难计算。通过上面的三个式子，我们可以发现岭回归和拉索回归就是子集选择回归的一种代替



对比上面的两个式子，可以发现拉索回归的系数约束区域是一个菱形，而岭回归的系数约束区域是一个圆形，系数的估计区域落在约束区域和估计曲线的交点上。这可以从几何上解释为什么拉索回归可以将系数压缩到 0

> 上述的菱形和圆形在 $p > 2$ 时扩展为对应的几何图形

不同的情况下，我们选择不同的模型。一般在预测变量很多且少量的特征拥有大系数时，考虑用拉索回归；在大部分系数都和响应变量有关，且系数相差不大时，我们考虑岭回归。实际应用中我们无法预知这些情况，所以可以考虑交叉检验来实际计算检验误差

> 值得注意的是，拉索回归的可解释性比岭回归好

拉索回归通过**软阈值 (soft thresholding)** 算法来执行变量选择的工作。简单的举例，若 $|\beta > \lambda$ 则 β 减小到 $\beta - \lambda$；若 $|\beta| \leq \lambda$ 则 β 被置为0

利用**贝叶斯观点 (Bayes Lens)** 研究拉索回归和岭回归得到的结论是：

- 岭回归相当于假设回归系数服从正态分布的先验，最大化后验分布得到的解趋向于平滑且不会将系数压缩为零
- 拉索回归则假设回归系数服从**拉普拉斯分布 (Laplace distribution)** 的先验，最大化后验分布得到的解倾向于稀疏化，即一些系数会被压缩为零，从而自动进行特征选择

> 理论分析难度很大，忽略

6.2.3 选择调谐参数

使用交叉检验我们可以确定一个合适的 λ 参数值以进行岭回归和拉索回归

在回归领域，相关和不相关的特征被分别称为**信号 (signal)** 和噪声变量

6.3 降维方法

前面使用的两类方法使用的特征向量都是由原始特征得来的。下面介绍的方法将预测变量进行转换，对转换后的预测变量进行最小二乘回归

令 $Z_1, Z_2, \cdots, Z_m$ 表示 $M < p$ 个原始预测变量的**线性组合 (linear combination)**：

$$Z_m = \sum_{j=1}^{p} \phi_{jm} X_j$$

其中 $\phi_{1m}, \phi_{2m}, \cdots$ 是常数。我们用这些预测变量进行最小二乘回归：

$$y_i = \theta_0 + \sum_{m=1}^{M} \theta_m z_{im} + \epsilon_i, \quad i = 1, \ldots, n$$

当我们的参数 φ_{pm} 选择得当时，拟合上面降维后的线性回归要比直接拟合线性回归得到的结果更好。这时候，问题的维度从 $p+1$ 下降到 $M+1$

注意到：

$$\sum_{m=1}^{M} \theta_m z_{im} = \sum_{m=1}^{M} \theta_m \sum_{j=1}^{p} \phi_{jm} x_{ij} = \sum_{j=1}^{p} \sum_{m=1}^{M} \theta_m \phi_{jm} x_{ij} = \sum_{j=1}^{p} \beta_j x_{ij}$$

其中：

$$\beta_j = \sum_{m=1}^{M} \theta_m \phi_{jm}$$

因此降维的线性回归可以看做原线性回归的一个特例，是为了限制原系数 β_j 必须满足某个特殊形式 (降维后的线性回归)

在 p 相对于 n 很大的情况下，选择 $M << p$ 的一个值可以显著降低预测的方差。当 $M = p$ 且所有 Z_m 线性无关时，执行的实际上是原线性回归

降维方法都分为下面两步：

1. 对预测因子进行转换
2. 对转换后的预测因子进行拟合

然而，如何选组转换系数有多种方法。下面介绍两种方法：主成分分析和偏最小二乘

6.3.1 主成分分析

主成分分析 (Principal components analysis) 是一种常用的降维工具

主成分分析的操作方式是将原数据**投影（project)** 到一条直线上，使原数据在这条新直线上的方差最大

> 方差最大的目的是保留最多的原始信息

我们可以考虑构造多个这样的直线，用以投影原始数据的信息。其中，用于投影数据的直线叫做**主成分轴**，捕获最多变量的主成分是**第一主成分 (first principal component)**，依次类推可以得到第二主成分等，原数据投影后得到的数据叫做**主成分得分 (principal component scores)**。

> 另一种描述主成分轴的方法是主成分轴刻画了到所有原始数据距离最近的一条直线

主成分之间应当是线性无关的。可以证明，若两个主成分之间线性无关，则主成分轴**垂直 (perpendicular)**

对于第 i 主成分，捕获的信息量依次降低。通过选择前 k 个主成分忽视后面的主成分，可以达到降维的效果，同时去除数据的共线性

主成分回归 (principal components regression, PCR) 通过先对原始数据矩阵进行主成分分析 PCA，再用主成分得分进行线性回归

> 这里的假设是，$X_1, X_2, \cdots, X_p$ 变化最大的方向是跟 Y 有关的方向

当主成分回归的假设成立时，回归得到的结果要比用全部特征值进行拟合更好，原因是降低了过拟合的风险

使用模拟的数据表明，主成分回归能在一些数据集中表现出明显优于 OLS 的效果。主成分回归实际上不能执行变量选择的效果，因为它拟合的是所有变量的线性组合。在这个意义上，主成分回归跟岭回归很类似 (事实上岭回归可以看做主成分回归的连续版)

在进行主成分分析是，通常先对数据进行标准化，防止高方差对主成分的干扰；除非数据都是同一尺度下的

6.3.2 偏最小二乘回归

上面描述的 PCR 方法涉及识别最能代表特征向量变化的线性组合，这是以无监督的形式实现的，因为 Y 不用于确定主成分方向。因此 PCR 的缺点是虽然获得了最能代表特征变化的方向，但无法保证是最能预测 Y 的变化方向

现在介绍**偏最小二乘回归 (Partial Least Squares, PLS)** 是一种**监督 (supervised)** 形式的 PCR 代替。它不仅试图找到最能代表特征向量的变化方向，还试图找到最能贴近 Y 变化的方向

PLS 的执行方式是，进行数据标准化后，首先将 Y 和 X 投影到低维空间得到一组**潜变量 (Latent Variables)**，再将解释不了的部分放回原矩阵，重新迭代，直至选择前 k 个潜变量进行线性回归分析

PLS 在实际应用中并不总是比 PCR 或岭回归等更好，因此需要结合实际应用

6.4 高维问题的思考

6.4.1 高维数据

现代科技使得数据的维度越来越多，例如：

- 用 SNPs (DNA 上的核苷酸) 来预测患者的血压、脉搏等。此时的 p 可能达到 $500,000$ 的水平
- 将每个用户输入的关键词都当做一个特征，这常常被称为词袋分析 (bag-of-words)，这里的 p 将会达到一个特别大的水平

特征值多的问题被称为**高维的 (high-dimensional)**。这些高维度的问题通常被认为是 $p > n$ 的问题，我们讨论的问题也实际 n 略大于 p 的问题

6.4.2 高维度下的问题

下面以最小二乘回归为例，介绍高维度问题下需要注意的技术

当 $p > n$ 时，不论特征与响应之间是否有关，都会产生一组完美系数使得残差为 0。这使得我们可能误认为特征越多的模型预测效果是最好的，因为它的训练误差或 R^2 是最好的

值得注意的是，在高维度状态下，C_p、AIC 和 BIC 是不合适的，因为在高维度情况下根据已知公式得出的 $\hat{\sigma}^2 = 0$。同样地，调整后 R^2 也不适用，因为可以很快地得到一个 $R^2 = 1$ 的模型

6.4.3 高维度回归

拉索回归可以在高维度模型下显示出一定的优越性能，这是由于它 λ 取值对模型特征值的压缩实现的。一个良好的 λ 值可以避免过拟合问题。通过对 λ 选取显示的拟合情况的观察，我们发现：

- 正则化或收缩在高维问题中非常关键
- 选择合适的调谐参数对于发挥模型性能非常重要
- 测试误差往往随着特征维数的增加而增加，除非额外的特征与响应真正相关

上述第三点被称为**维度灾难 (curese of dimensionality)**。这是因为噪声增加的问题维度加剧了过拟合问题

6.4.4 高维问题下的结果解释

在高维度问题下执行拉索回归等程序，必须严谨地判断结果。在第三章学习的多重共线性告诉我们，回归中的变量可能存在相关性，这在高维问题中表现为某个特征是其他特征的线性组合，导致我们无法得知到底哪些变量才是对回归结果的真正预测 (最多将大系数分给这个特征)

在模型特征选择上得到的模型可能有很好的预测效果，但我们必须谨慎地确定是否它们确实是影响模型的特征 (而非它们的线性组合)。

需要注意 $p > n$ 时，不能使用 R^2 和 MSE 等来度量模型有效性 (除了测试集上的)

第七章 超越线性

本章主要讨论线性模型的非线性扩展，最后证明这些扩展可以集成到一起

7.1 多项式回归

多项式回归 (polynomial regression) 是一种线性回归的非线性扩展，形式如下：

$$y_i = \beta_0 + \sum_{k=1}^{d} \beta_k x_i^k + \epsilon_i$$

其中 ϵ_i 是误差项。对于一个足够大的 d，我们能得到一个相当非线性的曲线。系数可用最小二乘回归拟合，即把 x_i^k 当做一个变量

> 一般来说，$d \leq 4$，过于大的 d 会导致曲线变得很奇怪 (过拟合)

使用预测点 $\hat{f}(x_0)$ 的标准差，我们可以画出预测曲线的 95% 置信区间，他大约在 $2 \pm \sigma$ 上。$Var[\hat{f}(x_0)]$ 满足：

$$Var[\hat{f}(x_0)] = l_0^T \hat{C} l_0$$

其中 $l_0^T = (1, x_0, x_0^2, \cdots)$，$\hat{C}$ 是回归系数 $\hat{\beta}$ 的协方差矩阵，满足：

$$\hat{C} = Var(\hat{\beta}) = \sigma^2 (X^T X)^{-1}$$

其中 σ^2 是误差项的方程，用残差方差估计为 $\hat{\sigma}^2 = \frac{RSS}{n-p-1}$，$X$ 表示设计矩阵

根据某一预测值的方差计算公式，我们发现该处的方差不仅与该点的特征向量有关，还和该点的系数协方差矩阵有关。这可以解释距离数据区域很远的点的极大方差

7.2 阶跃函数

现在除去线性回归的全局线性假设，我们尝试通过用常数和离散化的矩阵 X 来对模型进行预测

定义**阶跃函数 (Step Functions)** 的指示函数为：

$$
\begin{aligned}
C_0(X) &= I(X < c_1), \\
C_1(X) &= I(c_1 \leq X < c_2), \\
C_2(X) &= I(c_2 \leq X < c_3), \\
&\vdots \\
C_{K-1}(X) &= I(c_{K-1} \leq X < c_K), \\
C_K(X) &= I(c_K \leq K)
\end{aligned}
$$

$I(\cdot)$ 表示当条件满足时值为 1，否则为 0。因此这些函数满足 $\sum_{i=0}^{K} C_i(X) = 1$，将函数划分为了 $K+1$ 段，并在每段中用一个常数 $\beta_0 + \beta_i$ 来预测对应的值。用阶跃函数进行回归的表达式：

$$
y_i = \beta_0 + \sum_{i=1}^{K} C_i(x_i) + \epsilon_i
$$

在这里，我们可将 β_0 解释为预测值的均值，β_i 表示在 X 满足某条件下相对于均值的平均涨幅

像类似年龄这样存在自然断点的函数可以用阶跃函数回归来拟合。然而，其他的函数则不适用，因为它不够平滑，会忽略断点之间的过渡关系

7.3 基函数

基函数 (Basis Funcions) 是对上面两种方法的统称。我们注意到这两种方法都是需找到一类函数 (多项式函数或分段常数函数) 来拟合数据，可以写成如下的同一函数：

$$
y_i = \beta_0 + \sum_{i=1}^{K} \beta_i b_i(x_i) + \epsilon_i
$$

除了多项式函数和分段常数函数，我们还可以构造其他函数来进行拟合，如微波和傅里叶展开

7.4 回归样条

7.4.1 分段多项式回归

不同于考虑拟合一个全局的多项式回归函数，我们将特征矩阵 X 划分为 K 个不同的部分，并在每个部分用一个次数较低的多项式函数进行拟合，其中每段的多项式函数同

$y_i = \beta_0 + \sum_{k=1}^{d} \beta_k x_i^k + \epsilon_i$，在不同段中，$\beta_{ij}$ 的估计值不同。这种估计方法叫做**分段多项式回归 (piecewise polynomial regression)**，分段处叫做**结点 (knot)**

更多的结点会导致更灵活的分段多项式，对于 K 个结点，我们有 $K+1$ 个多项式。分段常数函数是 0 次分段多项式。拟合分段多项式模型需要消耗对应的**自由度 (degrees of freedom)**

7.4.2 约束和样条

直接进行分段多项式回归得到的样条会出现间断点，导致解释上的缺失，我们通过添加约束条件来改进：

- 连续的结点：要求结点两端的函数值相等
- 平滑过渡：要求结点两端的函数的**导数 (derivative)**连续，可延伸到多阶导数。设定连续型的阶数为 n 的曲线称为 n **次样条曲线** 。带有 K 个结点的三次样条曲线消耗 $4+K$ 个自由度

7.4.3 样条的基表示

一个三次样条曲线可以表示为：

$$y_i = \beta_0 + \beta_1 b_1(x_i) + \beta_2 b_2(x_i) + \cdots + \beta_{K+3} b_{K+3}(x_i) + \epsilon_i$$

共有 $K+4$ 个自由度。为了表示这个函数，我们将它分为两个部分：

- 描述整体立方关系 x, x^2, x^3
- 对应段的描述 **截断幂基函数 (truncated power basis)**

  $$h(x, \xi) = (x - xi)_+^3 = \begin{cases} (x - xi)^3, & x > \xi \\ 0, & otherwise \end{cases}$$

  其中 ξ 是结点。对于三次多项式，加入 $h(x, \xi)$ 将会保证二阶之前的倒数连续，但是三阶导数不一定连续

  > 随着 K 的增加，将会添加更多的 $h(x, \xi)$ 项；每经过一个结点就会加上一次 $h(x, \xi)$

普通的三阶样条曲线在边界上的置信区间非常宽泛，我们用**自然样条 (natural spline)** 来规避这个问题。自然样条要求边界处的预测曲线是直线，以降低预测的方差

7.4.4 选取结点数量和位置

我们可以通过观察数据分布来选择结点数量和位置，例如，在数据变动大的地方放置结点，以提高灵活性；数据变动少的地方则不放置结点

实践中，我们通常指定模型需要的自由度，然后利用软件在数据的某些分位点放置结点

选择最优的结点划分数量，可以通过交叉检验计算模型的均方误差来确定

提到的方案：

- 观察散点图
- 指定自由度，自动选择分位点
- 交叉检验

7.4.5 与多项式回归的比较

结论：样条在边缘区域的拟合效果比过于灵活的多项式回归更好

7.5 平滑样条

7.5.1 平滑样条简介

最小二乘回归的目的是找到一条合适的曲线使得：

$$RSS = \sum_{i=1}^{n}(y_i - g(x_i))^2$$

潜在的问题是可能找到一条曲线过于灵活，导致 $RSS = 0$

不同于单纯考虑 RSS，我们希望 $g(x_i)$ 能最小化：

$$\sum_{i=1}^{n}(y_i - g(x_i))^2 + \lambda \int g''(t)^2 dt$$

其中的 λ 为非负条件参数，满足上式的 $g(x_i)$ 称为**光滑样条 (smoothing spline)**，
$RSS = \sum_{i=1}^{n}(y_i - g(x_i))^2$ 称为**损失函数 (loss function)**

$\lambda \int g''(t)^2 dt$ 是一个惩罚项。它衡量了曲线变动的幅度。若曲线变动幅度大，则此项的值会变大；相对的，若曲线变动幅度很小，则此项的值也会很小。因此，$\lambda \int g''(t)^2 dt$ 鼓励更加平滑的曲线。$\lambda = 0$ 时，曲线波动异常，容易过拟合；$\lambda \to \infty$ 时，曲线变为直线

平滑样条的最优解是一个自然三次样条，它的特点是：

- 由三次多项式拼接二次
- 结点自动放在 $\forall x_i$ 处
- 边界区域的曲线是线性函数

但是它又是一个“收缩版本”，λ 控制了曲线的收缩程度

7.5.2 选择平滑参数 λ

合适的平滑参数 λ 能够压缩**有效自由度 (effective degrees of freedom)**。我们讨论有效自由度而不是自由度的原因是原有的自由度受到了严重压缩，有效自由度的定义为：

$$\hat{\mathbf{g}}_\lambda = \mathbf{S}_\lambda \mathbf{y}$$

其中 $\hat{\mathbf{g}}_\lambda$ 是所有训练数据点的预测值向量，大小 $n \times 1$；y 是真实响应值

上式表明，对数据施加平滑样条时拟合值的向量可写错 $n\times n$ 矩阵 $\mathbf{S}_\lambda$ 乘以响应向量 y。定义有效自由度为：

$$df_\lambda=\sum_{i=1}^n\{\mathbf{S}_\lambda\}_{ii}$$

也即是 $\mathbf{S}_\lambda$ 的对角线元素之和

拟合光滑样条的时候，我们不需要选择节点数或位置，因为每一个训练值都是一个结点。经验表明，使用 LOOCV 可以非常有效地计算 RSS 较小的 λ 值，下面是简化公式：

$$\begin{aligned}RSS_{cv}(\lambda)&=\sum_{i=1}^n(y_i-\hat{g}_\lambda^{(-i)}(x_i))^2\\&=\sum_{i=1}^n\left[\frac{y_i-\hat{g}_\lambda(x_i)}{1-\{\mathbf{S}_\lambda\}_{ii}}\right]^2\end{aligned}$$

其中 $\hat{g}_\lambda^{(-i)}(x_i)$ 是不包含 (x_i,y_i) 时的模型拟合值；$\hat{g}_\lambda(x_i)$ 表示所有的平滑样条拟合值

> 在有效自由度定义下，自由度可以是非整数

7.6 局部回归

局部回归 (Local regression) 是一种不同的拟合非线性模型的方法，其算法可以阐述如下：

1. 收集距离点 x_0 最近的 k 个训练值 x_i
2. 对每个点赋予权重 $K_{i0}=K(x_i,x_0)$，使得距离 x_0 最近的点权重最大，距离 x_0 最远的点权重最小；且非最近 K 个点的权重为0
3. 拟合**加权最小二乘模型 (weighted least squares regression)**，例如，一种最小化目标是最小化：

$$\sum_{i=1}^nK_{i0}(y_i-\beta_0-\beta_1x_i)^2$$

4. 在这个点 x_0 处，x_0 的预测值由 $\hat{f}(x_0)=\hat{\beta}_0+\hat{\beta}_1x_0$ 给出
5. 选取新的 x_0 重复计算，直至拟合一条平滑的拟合曲线

> 这个模型和 KNN 很像，但是局部回归的曲线相比于 KNN 要更加平滑

选择合适的窗口大小决定了回归的平滑程度，当 $s=\frac{k}{n}$ 越大时，拟合曲线的波动就越小

局部回归可以进行扩展，例如：

- 部分变量上应用的局部回归。例如对于时间序列数据，人口 X_1 和 GDP X_2 对于放假的影响是长期稳定的，适合全局回归，但时间 X_3 却可能表现出短期趋势变化，适合局部回归。我们因此构造一个**变系数模型 (varying coefficient model)** 对数据进行拟合
- 高维度数据的局部回归。对于二维数据，我们可以利用二维空间中的最近点来拟合局部回归；但是对于 $p>3$ 或者 $p>4$ 以上的空间，这个模型的效果较差，这是因为稀疏的临近

数据

7.7 广义可加模型

广义可加模型 (generalized additive model, GAM) 在保持**可加性 (additivity)** 同时提供了一种广泛的扩展普通线性回归的框架，可用于定性和定量的变量，相当于多元线性回归模型的扩展

7.7.1 用于回归问题的 GAM

用函数代替多元回归中的不同特征项，我们得到：

$$\begin{aligned} y_i &= \beta_0 + \sum_{j=1}^{p} f_j(x_{ij}) + \epsilon_i \\ &= \beta_0 + f_1(x_{i1}) + f_2(x_{i2}) + \cdots + f_p(x_{ip}) + \epsilon_i \end{aligned}$$

这被称为可加性模型，因为我们对于每一个特征 X_j，都有对应的一个函数 f_j 并对它们进行加总

我们讨论过的大多数模型都可以用于构建 GAM。`pygam` 提供了 Python 中的 GAM 模型构建器，值得一提的是它关于拟合平滑样条的方法**回归自适应拟合 (Backfitting)**，它能够：

- 处理多个预测变量，即使它们非线性
- 不受到变量个数的限制，可通过可加性独立考察每个变量的影响
- 平滑方法灵活，包括局部回归，样条回归和核平滑

GAM 的局限性体现在模型仅限于可加性，不过我们可以手动添加交互项，或者加入地位相互作用函数

7.7.2 用于分类问题的 GAM

多元逻辑回归模型：

$$\log\left(\frac{p(X)}{1-p(X)}\right) = \beta_0 + \sum_{i=1}^{p} \beta_i X_i$$

扩展到 GAM：

$$\log\left(\frac{p(X)}{1-p(X)}\right) = \beta_0 + \sum_{i=1}^{p} f_i(X_i)$$

可用于分类问题

第八章 基于树的模型

基于树的方法是一种通过将预测变量空间划分为多个简单区域进行预测的模型，每个区域对应一个决策规则，可用树状结构表示。这也被称为**决策树 (decision tree)**

简单的决策树在分类和回归任务中不如第六和第七章描述的正则化方法或样条模型；但是，我们后续介绍的集成方法将会有更高的精度，尽管牺牲了一些可解释性

8.1 决策树基础

决策树可用于回归和分类两类问题

8.1.1 回归树

用于回归问题的决策树就是**回归树 (regression tree)**

在利用回归树预测运动员工资的例子中，我们发现回归树的如下特点：

1. 良好的分支结构，清晰明了
2. 相对于线性回归，树的分支、结点似乎更容易解释

构建决策树的过程，粗略的说可以分为：

1. 划分预测器空间，对于预测器向量组 $X_1, X_2, \cdots, X_p$，划分 J 个区域
2. 对于每个落入区域 R_j 的观测值，我们作同样的预测：简单的可以认为它们都在原始观测空间 R_j 所有数据的均值上

理论上，构造的空间 R_j 可以为任意形状，但我们倾向于将其划分为高维立方体，或者可以叫做**盒子 (boxes)**。寻找能使模型获得最小 RSS 的盒子遵循公式：

$$\sum_{j=1}^{J}\sum_{i\in R_j}(y_i-\hat{y}_{R_j})^2$$

其中 $\hat{y}_{R_j}$ 是第 j 个盒子所有观测值的均值

在计算上，对于全部的特征，考虑所有的划分方式是不可能实现的。为此，我们采用**递归二分法 (Recursive Binary Splitting)**，或者称为**自顶向上的贪婪方法 (Top-Down Greedy Approace)** 来划分盒子。这种方法从树的顶端开始，选择一个特征 X_j，选择能最小化 RSS 的阈值 s，划分出一片空间，再对所有的特征递归使用此方法，可以划分出所有的空间

> 这种算法的思想是贪婪。每步只考虑了当前的最优分发，但不能保证持续的划分在全局上都是最优

对于任意的j和s，我们定义**半平面对 (half-planes)**:

$$R_1(j,s)=\{X|X_j<s\} \text{ and } R_2(j,s)=\{X|X_j\leq s\}$$

然后我们寻找合适的j和s，使得最小化下式：

$$\sum_{i:x_i\in R_1(j,s)}(y_i-\hat{y}_{R_1})^2+\sum_{i:x_i\in R_2(j,s)}(y_i-\hat{y}_{R_2})^2$$

其中$\hat{y}_{R_i}$表示第i个平面训练值的平均值。当p不大时，寻找对应j和s的值相当快

划分完第一个特征后，我们继续进行划分，不同的是，我们不再对整个区域进行划分，而是选择划分好的一块区域，再根据新的特征，或对原来特征进一步约束来划分新区域

区域划分的终止条件可以设置为划分后剩下的叶子结点数量 (即最后的所有可能区域数)

我们拟合的树模型可能在训练集上表现较好，测试集的性能不佳，这是因为生成的复杂的树容易导致过拟合

一种解决过拟合的方案是设置分裂阈值，即到达某个阈值的 RSS 减小时，才对树进行分裂。然而，这种策略可能导致忽略前期的有用分裂 (这种分裂为后期的大量 RSS 减小创造了条件)

因此，一种更优的策略是生成一颗足够大的树，然后**剪枝 (prune)**，最后得到一颗它的**子树 (subtree)**。选择更好子树的指标是它的测试误差

> 交叉检验或验证集方法可以寻找这种子树，但可能过于繁琐

我们使用**代价复杂度剪枝 (cost complexity pruning)**，也叫**最弱连接剪枝 (weakest link pruning)** 来获取合适的子树。对于任意一颗子树，都有一个非负的惩罚项α对应，对α的每个值，子树$T \in T_0$，满足：

$$\sum_{m=1}^{|T|} \sum_{i:x_i \in R_m} (y_i - \hat{y}_{R_m})^2 + \alpha |T|$$

要使上式尽可能小。其中$|T|$表示树T的终端结点数，R_m是第m个终端结点对应的矩形 (预测空间的子集)，$\hat{y}_{R_m}$表示对应R_m中训练值的均值，调节参数α调控子树复杂度和拟合度的平衡

当子树的$\alpha = 0$时，子树T简单地等于T_0，因此上式只度量训练误差。然而，随着α增加，多个终端结点的树付出的代价增大，对于较小的树上式值反而较小。剪枝方法将从叶子结点向上，依次删除对于RSS影响最小但对于模型复杂度影响最大的分支

> 这种控制方法类似于 lasso 回归

对于α，我们可以用交叉检验的方式来选取

构建回归树的算法归结如下：

1. 使用递归二进制分裂在训练数据上生成一颗大树，直到某个区域内的观测数少于阈值
2. 应用代价复杂度剪枝，获取最优子树序列
3. 用K折交叉检验选择α，对于每个$k = 1, 2, \cdots, K$，用除了k的训练数据拟合模型，k用于检验误差，取平均值，找到一个使平均误差最小的α
4. 从步骤 2 中选择对应的子树

8.1.2 分类树

分类树 (classification tree) 与决策树最大的区别在于前者预测的是定性变量，而后者预测的是定量的变量。分类树的终端结点的预测值与盒子内的最多数量的种类有关，且我们关心它的比例

生成分类树比回归树更加简单。替代RSS的检验指标是**分类错误率 (classification error rate)**：

$$E = 1 - \max(\hat{p}_{mk})$$

其中$\hat{p}_{mk}$表示来自第k类且在第m区域的训练观测值比例。

> 事实上，分类误差对树木的生长不够敏感，需要更优的措施

基尼指数 (Gini index) 被定义为：

$$G = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk})$$

其中$\hat{p}_{mk}$的定义同上。当基尼指数很小时，$\hat{y}_{mk}$接近 1 或 0，可用于衡量一个节点分类的纯度，即大部分是否都为同一类

> 基尼指数还有一个定义是$G = 1 - \sum_{k=1}^{K} p_k^2$，等价，是上式的展开式

基尼指数的一个代替是**熵 (entropy)**:

$$D = -\sum_{k=1}^{K} \hat{p}_{mk} \log \hat{p}_{mk}$$

由于$0 \leq \hat{p}_{mk} \leq 1$，所以$D$是一个大于 0 的值。当$\hat{p}_{mk}$接近于 0 或 1 时，$D$会取一个很小值，跟基尼指数很类似

构建分类树可以用上面的三种指标，对于分割质量，我们采用后两种指标；若考虑更精确的分类效果，我们采用第一个指标分类错误率

注意到一颗完全生长的决策树，一个结点分支出的两个终端结点，可能会有相同的分类结果。出现这个的原因是此划分遵循了基尼指数和熵的优化，提升了这片空间的纯度，因而s被选中。纯度的提升有助于高置信预测，减少过拟合并提升解释性

> 例如，两个兄弟终端结点的分类都是 yes，但右侧的纯度为100%，若有后续预测变量分配到右侧，我们有很高的置信率去认为它也是 yes，便于解释

8.1.3 树和线性回归的比较

线性回归的预测遵循下面的式子：

$$f(X) = \beta_0 + \sum_{j=1}^{p} X_j \beta_j$$

而回归树则遵循：

$$f(X) = \sum_{m=1}^{M} c_m \cdot 1_{(X \in R_m)}$$

其中R_m表示划分的特征空间，$1_{(X \in R_m)}$是指示函数，表示$X \in R_m$时为 1，否则为 0，c_m表示对应区域内的预测值

选择模型的标准：

- 特征和响应之间高度线性，使用线性回归
- 特征和响应之间高度非线性，关系复杂，决策树方法优于传统方法
  除此之外，出于可解释性和可视化的考虑，决策树的预测更受欢迎

8.1.4 树模型的优缺点

优点：

1. 容易解释。有时候，甚至比线性回归更加容易解释
2. 比经典回归和分类方法更加真实地反映人类决策
3. 树可以图形化，非专业人士可以看懂
4. 树可以很容易处理定性预测器，不需要哑变量

缺点：

1. 树的预测精度在一般情况下不如其他一些分类和回归方法
2. 树的稳健性较差，数据的微小变化容易引起树模型的较大变化

聚合许多决策树、使用 Bagging、随机森林和 Boosting 等方法可以大幅提高预测性能

8.2 树的集成方法

集成 (ensemble) 的含义是组合许多简单的模块，以获得一个独立而非常有效的模型。这些简单的积木模型有时候被称为**弱学习器 (weak learners)**，因为它们本身可能得较弱的预测

8.2.1 Bagging

自助聚集法 (Boostrap Aggregating) 是一种使用了自助法来改进决策树模型的新模型

决策树的高方差导致了其容易过拟合。例如，将数据随机分为两类，分别拟合决策树，最后的决策结果可能很不同，这是因为决策树对数据细节非常敏感。决策树是一类**高方差模型 (high variance model)**，而使用 Bagging 方法可以降低它的方差

> 与之相反，当$n >> p$时，线性回归是一类**低方差模型 (low variance model)**

回顾方差的计算方法。对于n个独立观测值$Z_1, \cdots, Z_n$，每个观测值的方差为σ^2，则这些观察值的均值$\overline{Z}$的方差就是σ^2/n。因此，对于独立观察值，我们通过平均的方差可以降低方差。我们可以通过计算$\hat{f}^1(x), \hat{f}^2(x), \cdots, \hat{f}^B_(x)$来计算$B$个独立模型的输出值，然后取平均作为输出值，即：

$$\hat{f}_{avg}(x)=\frac{1}{B}\sum_{b=1}^{B}\hat{f}^{b}(x)$$

在实际中，这是不实际的，因为我们通常不拥有多个数据集。但是我们可以在单个数据集中使用 bootstrap 方法来生成多个小数据集，然后分别训练模型，再输出结果：

$$\hat{f}_{bag}(x)=\frac{1}{B}\sum_{b=1}^{B}\hat{f}^{*b}(x)$$

Bagging 可以改进许多回归方法的预测，但对决策树特别有用。通过继承数百甚至上千颗未经过修剪的决策树，然后取平均输出，Bagging 已被证明在准确性上有显著改进

> 树数量 B 的大小不是关键参数，不会导致模型的过拟合。实践中我们采用显著降低误差的 B 作为最终大小

对于分类问题，我们的 Bagging 方法采用**多数投票 (majority vote)** 来获得最终的结果。即得到所有子决策树的输出，然后取最多的那一类预测

事实证明，一个简单的估计 Bagging 模型测试误差的方法是**袋外误差估计 (out-of-bag Error Estimation)**。由于 boostrap 方法是放回抽样，每次约有$1-\frac{1}{e}$(约为$1/3$) 的数据没有被抽取到。我们利用袋外数据来估计单颗子树的测试误差，再把所有的误差平均。当B足够大时，OOB 估计实验误差等价于 LOOCV。OOB 估计对大数据集较为友好

Bagging 方法损失了可解释性，以获得模型准确率 (我们不能再通过单颗子树绘图了)。但是，我们可以考虑一些指标来评估每个特征的重要性，也即**变量重要性 (variable importance)**

我们可以这样收集变量重要性：

- 对于回归树，在每颗子树中，对于一个特征的每次分裂，记录 RSS 的减少总和，最后对所有B颗树进行平均，以显著降低 RSS 的特征为重要
- 对于分类树，在每颗子树中，对于一个特征的每次分裂，记录基尼指数的下降总和，最后对所有B颗树进行平均，以显著降低基尼指数的特征为重要

> 在教材中，变量重要性被映射到一个$[0,100]$的区间上

8.2.2 随机森林

随机森林 (random forest) 实现了对 Bagging 方法的改进。Bagging 方法每次构建树时，都是从同一个数据集中进行抽样再建立的，这将会导致树与树之间更加强的相关关系，从而弱化模型的泛化能力

与 Bagging 类似，随机森林也是在一个数据集中抽取多个 Bootstrap 样本，但有一个关键的改进：每次分裂时，从p个特征中随机选取m个，然后再从m个中选择最优的进行分裂。实践中，$m\approx\sqrt{p}$

> 例如，若p中存在一个高强的预测器，其他为中强，则大部分树都会选择此高强预测器，导致树之间高度相关。在这种情况下，平均所有树的输出不能显著降低模型的方差

随机森林迫使每个分裂只考虑预测变量的一个子集，这可以看作是对树的去相关处理，减小树输出的平均值变动。随机森林的关键即是选择m的大小，当$m = p$时，等同于 Bagging；当大量相关的预测变量存在时，我们使用很小的m值会有很大帮助

8.2.3 Boosting

Boosting是一种集成学习方法，通过组合多个弱学习器来构建一个强学习器 (就像 Bagging)，可用于多种模型中提升效率。下面的讨论将仅限于树模型

Boosting 同样生成一系列的子树，不同的是，每棵树都是利用先前的树的信息生长的。Boosting 不涉及 bootstrap 抽样，而是每棵树在原始数据集的修改版本上进行拟合

与直接对Y来拟合一颗决策树不同，Boosting 方法对模型的残差进行拟合。然后，将一颗新的决策树加入拟合函数中，更新残差，这样算法中的每一颗树都很小 (少量终端结点，由算法中的d决定)，通过残差对小树拟合，在表现不好的区域慢慢提升$\hat{f}$，允许更多不同的形状来削减残差

> 注意 Boosting 与 Bagging 不同在每颗树都依赖于前面的树，不是独立

Boosting 算法应用于回归树可以如下描述：

1. 设置初始模型$\hat{f}(x) = 0$，初始化残差$r_i = y_i$(容易从$\hat{f}(x) = 0$推出)
2. 对于每一颗子树，从$1, 2, \cdots, B$，重复
   2.1 拟合一棵树$\hat{f}^b$，并划分d个分支 (最终得到$d + 1$个终端结点) 到训练数据(X, r)
   2.2 通过添加一个缩小版本的新子树，更新总体模型$\hat{f}$：

$$\hat{f}(x) \leftarrow \hat{f}(x) + \lambda \hat{f}^b(x)$$

   2.3 同时更新残差

$$r_i \leftarrow r_i - \lambda \hat{f}^b(x_i)$$

3. 输出总的 Boosted 模型：

$$\hat{f}(x) = \sum_{b=1}^{B} \lambda \hat{f}^b(x)$$

> 2.1 中提到的训练数据(X, r)表示一个聚合矩阵，其中X表示特征矩阵，r在此处表示残差向量。每个模型都试图通过这两份数据更新

在决策树中应用 Boosting 方法，有三个调谐参数：

1. 子树数量B，不同于 Bagging 方法，过大的B会导致 Boosting 过拟合。通过交叉检验选择合适的 B

2. 收缩参数λ，也即是学习率。经典的值取0.01或者0.001，正确地选择取决于问题。非常小的λ需要一个非常大的B以获取较优的模型
3. 每颗子树的分裂数d，控制模型复杂性。$d=1$通常表现的较好，这时候树变为**树桩(stump)**，只有一组分裂，此时 Boosting 是加性模型，每次迭代只有一个特征；更广泛地，d被称为**交互深度**，多次的分裂可以学习变量之间的交互关系

> 单颗树的加性模型有助于解释模型的作用

8.2.4 贝叶斯加性回归树

现在我们讨论**贝叶斯加性回归树 (Bayesian additive regression trees, BART)**，这也是一种决策树的集成模型，它综合了前面两类集成学习的特点，新颖之处在于生成树的方式

定义K表示回归树数量，B表示模型的迭代次数，$\hat{f}_k^{(b)}(x)$表示在第b论迭代中，第k颗树对输入x的预测

在 BART 的第一轮中，每棵树都只有一个根节点 (即不做任何划分)，预测值为所有目标值y_i的均值，除以树的数量k：

$$\hat{f}^1(x)=\sum_{k=1}^{K}\hat{f}_{k=1}^1(x)=\frac{1}{n}\sum_{i=1}^{n}y_i$$

在 BART 的后续迭代中，它将逐步更新每棵树 (类似梯度提升树 GBDT)。对于第b轮需要更新的第k棵树，计算**部分残差 (partial residual)**：

$$r_i=y_i-\sum_{k'<k}\hat{f}_{k'}^b(x_i)-\sum_{k'>k}\hat{f}_{k'}^{b-1}(x_i)$$

> 上式的负数项可以分解为此轮已被更新的树模型预测值和未被更新的树模型的预测值；表示T_k的残差

然后根据所有观测值的残差，更新树。BART 不是将一颗新树来拟合这部分残差，而是从一组可能的**扰动 (perturbation)** 来更新，这包括：

1. 改变树的结构。包含分裂扩展和剪枝分支
2. 改变这棵树每个终端结点的预测值

通常，舍弃迭代轮次前面的模型，因为这些模型的拟合效果可能不太好，处于所谓的**预热期(burn-in)**。最终模型的预测均值将不包含预热期的值，记为：

$$\hat{f}(x)=\frac{1}{B-L}\sum_{b=L+1}^{B}\hat{f}^b(x)$$

其中L表示预热期模型数量

BART 算法可以概述为：

1. 拟合初始模型$\hat{f}_1^1(x)=\hat{f}_2^1(x)\cdots=\hat{f}_K^1(x)=\frac{1}{nL}\sum_{i=1}^{n}y_i$

2. 计算第一次预测值$\hat{f}^1(x)=\sum_{k=1}^K \hat{f}_k^1(x)=\frac{1}{n}\sum_{i=1}^n y_i$
3. 对于后续的迭代次数d：
   3.1 对于$k=1,2,\dots,K$：
   3.1.1 对于$i=1,2,\dots,n$，计算部分残差$r_i=y_i-\sum_{k'<k}\hat{f}_{k'}^b(x_i)-\sum_{k'>k}\hat{f}_{k'}^{b-1}(x_i)$
   3.1.2 用随机扰动拟合新树$\hat{f}_k^b(x)$对r_i，选择提升最好的那颗树
   3.2 计算本次预测值$\hat{f}^b(x)=\sum_{k=1}^K \hat{f}_k^b(x)$
4. 计算预热期L之后的预测均值：

$$\hat{f}(x)=\frac{1}{B-L}\sum_{b=L+1}^{B}\hat{f}^b(x)$$

> 关键步骤是 3.1.2，即采用随机扰动修改树，不是完全重新拟合一颗新的树，避免过拟合

实验表明，BART 的训练和测试误差在K增大到一定程度后趋于平缓，而 Boosting 的训练误差随着B增大一直减小，测试误差则下降到一定程度趋于平缓。这表明前者没有产生过拟合，后者产生了。BART 防止过拟合主要归功于随机扰动的树修改方法

随机扰动对树的修改使用了拟合树集合的贝叶斯方法，每次的修改实际上是从后验分布中绘制一颗新的树。上述算法可以看做拟合 BART 的**马尔科夫链蒙特卡洛 (Markov chain Mont Carlo)** 算法

在实际应用中，我们通常选择较大的B和K值，适中的L值。BART 拥有很好的箱外表现，也即在最小的参数调优代价下的更优表现

8.2.5 集成树模型的总结

决策树是很适用于集成方法的，原因在于其灵活性和处理多个特征的能力。下面对上面提到的集成树模型进行总结：

- Bagging 的树木在随机观测样本上独立生长，树木之间往往非常相似，容易陷入局部最优，不能彻底探索模型空间
- 随机森林中的树木同样独立生长在观测的随机样本上，然而每次的分裂都是使用特征的随机子集，去除了部分相关性
- Boosting 方法只使用原始数据而不再次抽样，树木依次生长，采用缓慢的学习方法，每颗树都适应早期树木留下来的信号，并在使用前收缩
- BART 同样使用原始数据，树木依次生长。为了避免局部极小值，每颗树的扰动对模型进行了更加彻底的探索

第九章 支持向量机

支持向量机 (support vector machine) 是一类上世纪 90 年代发展起来的分类器，被认为是最好的“开箱即用”的分类器之一

本章讨论的概念涉及**最大间隔分类器 (maximal margin clssifier)**(不能用于大数据集)、**支持向量分类器 (support vector classifier)** 和支持向量机及其扩展。人们通常把上面三类分类器统称为支持向量机，本章对它们作出了详细区分

9.1 最大间隔分类器

9.1.1 超平面

在p维空间中，一个**超平面 (hyperplane)** 是一个维度为$p-1$的平坦仿射子空间。当$p=2$时，超平面是一条直线；$p=3$时，超平面是一个平面；而当$p>3$时，超平面很难可视化

二维空间中的超平面可以这样定义：

$$\beta_0+\beta_1X_1+\beta_2X_2=0$$

也就是说，所有$X=(X_1,X_2)^T$满足上面方程的点都在这个超平面上 (实际上是一条直线)。上面的式子可以扩展为一个p维的超平面：

$$\beta_0+\sum_{i=1}^{p}\beta_iX_i=0$$

对于不满足超平面的点，例如：

$$\beta_0+\sum_{i=1}^{p}\beta_iX_i>0$$

表示点在超平面的一侧，而：

$$\beta_0+\sum_{i=1}^{p}\beta_iX_i<0$$

则表示点在超平面的另一侧。通过比较不等符号，我们可以区分点在超平面的哪一侧

> 超平面的特性是，可以将对应的空间分为两个不相交的子空间

9.1.2 用分隔超平面分类

对于一个$n\times p$的数据矩阵$\mathbf{X}$，它包含n个p维向量作为观测值：

$$x_1=\begin{pmatrix}x_{11}\\\vdots\\x_{1p}\end{pmatrix},\cdots,x_n=\begin{pmatrix}x_{n1}\\\vdots\\x_{np}\end{pmatrix}$$

另有一个测试数据$x^*=(x_1^*,\cdots,x_p^*)^T$。现在，我们将用一种基于**分隔超平面 (separating hyperplane)** 的方法对数据进行分类

一个分隔超平面满足下面的特性。对于分别被标记为 1 和-1 的两类，分隔超平面有：

$$\beta_0+\sum_{j=1}^{p}\beta_jx_{ij}>0\quad if\ y_i=1$$

和：

$$\beta_0 + \sum_{j=1}^{p} \beta_j x_{ij} < 0 \quad if\ y_i = -1$$

等价的，在上面的分类条件下，一个分隔超平面满足：

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip}) > 0$$

对于任意$i = 1, 2, \cdots, n$成立

假设我们能找到这样的一个超平面，则可以根据：

$$f(x^*) = \beta_0 + \sum_{i=1}^{p} \beta_i x_i^*$$

的正负来判断它在超平面的哪个方向，且这个函数值的大小可以判断它距离超平面的距离。显然，超平面分类器会导致一个线性的分类边界

9.1.3 最大间隔分类器

如果数据能用超平面完美分开，那么实际上存在无穷个这样的超平面

为了选择唯一的超平面，一个自然的方法是**最大化间隔超平面 (maximal margin hyperplane)**，也被称为**最佳分隔超平面 (optimal separating hyperplane)**。称训练数据上的点到超平面的垂直距离为**间隔 (margin)**，则最大化间隔超平面会最大化所有训练数据到其的间隔，使得分类的准确性最大化。这样的分类器就是**最大化间隔分类器 (maximal margin classifier)**。然而，当p过大时，最大化间隔超平面容易导致过拟合

> 某种意义上来说，最大化间隔就是我们能够插入的最小超平面 (刚好分开某类数据的超平面) 的中线

对于位于分割超平面间隔附近的数据点 (即最小分隔超平面内的点)，我们称这些观测值为**支持向量 (support vetors)**，他们位于的超平面被称为**间隔边界 (margin)**。这是因为它们固定了最大化间隔超平面，如果它们略微移动，则这个超平面也会跟着移动 (其他的点则不会)

> 这个分类的一个特点是，最大间隔超平面直接依赖于观测值的一小部分子集

9.1.4 建立最大间隔分类器

建立最大化间隔分类器实际上是一个优化问题：

$$\underset{\beta_0, \beta_1, \ldots, \beta_p, M}{\text{maximize}}\ M \tag{1}$$

$$\text{subject to} \sum_{j=1}^{p} \beta_j^2 = 1 \tag{2}$$

$$y_i(\beta_0 + \sum_{j=1}^{p} \beta_j x_{ij}) \geq M,\ \forall i = 1, \ldots, n \tag{3}$$

这个式子没有它看起来那么困难。首先，对于式 (3)：

$$y_i(\beta_0 + \sum_{j=1}^{p} \beta_j x_{ij}) \geq M, \ \forall i = 1, \ldots, n$$

它保证了所有的数据点都位于超平面的某侧，存在一定的缓冲距离

而式子 (2) 则为 (3) 添加了意义，在此约束条件下，第i哥观测值到超平面的距离就是：

$$y_i(\beta_0 + \sum_{j=1}^{p} \beta_j x_{ij})$$

最后，式子 (1) 要求最大化这个平面到所有点的距离

> 优化问题的细节涉及拉格朗日乘数法，本书未提及

9.1.5 不可分情况

许多问题中的观测值不能找到可分的超平面，也就是说，不存在一个式子，能够满足所有的解都有$M > 0$

不可分情况下的最大化间隔分类器不可用。后面的模型将使用**软间隔 (soft margin)** 来进行几乎可分的分类。这种分类器就是支持向量分类器

9.2 支持向量分类器

9.2.1 支持向量分类器简介

最大间隔分类器存在如下的问题：

1. 观测数据不一定线性可分
2. 对单个数据变化敏感，容易发生过拟合

本章开发的分类器有如下改进：

1. 对少量观测值的更高稳健性
2. 对大多数观测值的更好分类效果

为了实现这些改进，对于少量的观测值，误分类是值得的

支持向量分类器，又称为**软间隔分类器 (soft margin classifier)**，能顾实现这些要求。对于部分数据，它可以存在于间隔的错误一侧，也可以存在于超平面的错误一侧

> 这就是说数据可能靠近边界但未被正确分类，也可能完全错误分类 (远离超平面)

9.2.2 支持向量分类器的细节

为了实现上一节的改进，我们对式子进行如下修改：

$$\underset{\beta_0,\ldots,\beta_p,\epsilon_0,\ldots,\epsilon_n,M}{\text{maximize}} M \tag{1}$$

$$\text{subject to} \sum_{j=1}^{p} \beta_j^2 = 1 \tag{2}$$

$$y_i(\beta_0 + \sum_{j=1}^{p} \beta_j x_{ij}) \geq M(1-\epsilon_i) \tag{3}$$

$$\epsilon_i \geq 0, \quad , \sum_{i=1}^{n} \epsilon_i \leq C \tag{4}$$

其中C是一个非负调谐参数；ϵ_i是一个非负的**松弛变量 (slack variable)**，允许数被误分类 ($\epsilon_i = 0$时，表示x_i被正确分类，且到超平面距离至少为 M)。当上式被确定下来后，就可用于分类观测数据x^*了

松弛变量告诉我们第i个观测值位于何处，$\epsilon_i > 0$则说明它被误分类了。参数C用于控制我们对误分类的容忍程度，实践中常用交叉检验选择

优化式子$(1) \sim (4)$指出，严格位于正确分类的边缘一侧的数据不影响支持向量分类器，位于超平面上，或者被错误分类的观测值 (被称为支持向量) 才会对支持向量分类器产生影响

> 这意味着支持向量分类器对远离超平面的观测值的行为具有相当的稳健性

9.3 支持向量机

9.3.1 非线性决策边界的分类

支持向量分类器无法简单地处理非线性决策边界。为了解决这个问题，我们考虑使用：

- 添加交互项，如X_1X_2
- 引入高次特征，如X_1^2

来扩展特征空间，并处理非线性决策边界。对于引入二次特征，我们的约束函数组变为：

$$\underset{\beta_0,\beta_{11},\beta_{12},\ldots,\beta_{1p},\beta_{2p},\epsilon_1,\ldots,\epsilon_n,M}{\text{maximize}} M$$

$$\text{subject to} y_i \left(\beta_0 + \sum_{j=1}^{p} \beta_{j1} x_{ij} + \sum_{j=1}^{p} \beta_{j2} x_{ij}^2 \right) \geq M(1-\epsilon_i)$$

$$\sum_{i=1}^{n} \epsilon_i \leq C, \epsilon_i \geq 0, \ , \sum_{j=1}^{p} \sum_{k=1}^{2} \beta_{jk}^2 = 1$$

当我们将这些扩展后的特征映射回原来的函数中时，高次项和交互项会导致决策边界的非线性

然而，当p很大时，引入过多的高次项和交互项使计算变得异常复杂。下面提到的支持向量机通过核函数技术，可以在不显示扩展特征空间的情况下，高效处理非线性问题

9.3.2 支持向量机

支持向量机使用**核函数 (kernels)** 进行特征空间的拓展化，其细节具有相当的技术性

首先，我们引入内积的概念：

$$< x_i, x_{i'} >= \sum_{j=1}^{p} x_{ij} x_{i'j}$$

内积就是两个向量的点积。则线性支持向量分类器的一个输出函数则可以表述为：

$$f(x) = \beta_0 + \sum_{i=1}^{p} \alpha_i < x, x_i >$$

其中参数α_i对应每个训练数据向量和观测向量之间的内积。为了估计α_i和β_0，我们需要计算C_n^2对向量$< x_i, x_{i'} >$之间的内积

数学上证明，为了计算$f(x)$，我们计算点x和训练数据x_i之间的内积有如下性质：若x_i不是支持向量，则它的观测$\alpha_i = 0$。这些系数α_i非 0 的点就是支持向量。我们设支持向量集合为S，则输出函数可以变为：

$$f(x) = \beta_0 + \sum_{i \in S} \alpha_i < x, x_i >$$

这比之前的形式减少了很多项

现在，我们将式中的内积替换为广义内积：

$$K(x_i, x_{i'})$$

其中K就是核函数。核函数以一种刻画两个观测值之间相似性的函数，例如一种核函数：

$$K(x_i, x_{i'}) = \sum_{j=1}^{p} x_{ij} x_{i'j}$$

这被称为线性核，与原来的支持向量分类器等价，它使用皮尔逊相关系数来刻画两个向量之间的相关性。我们可以替换为另一种核：

$$K(x_i, x_{i'}) = (1 + \sum_{j=1}^{p} x_{ij} x_{i'j})^d$$

这被称为**多项式核 (polynomial kernel)**，其中d是一个正整数。使用这个$d > 1$的核，我们可以刻画非常灵活的决策边界，即通过把特征转换到高维空间中

当支持向量分类器使用非线性核时，它就成为了所谓的支持向量机，此时它的输出函数具有如下形式：

$$f(x) = \beta_0 + \sum_{i \in S} \alpha_i K(x, x_i)$$

一种流行的核叫做**径向核 (radial kernel)**，它的形式如下：

$$K(x_i, x_{i'}) = \exp(-\gamma \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2)$$

其中γ是一个正常数。这个核使用指数衰减的形式来刻画一个训练数据对预测数据的影响，当二者的欧氏距离很近时，这个影响很大，否则很小。这个特性使得它能捕捉数据的局部结构

使用核函数相比于简单的原始特征扩大的好处在于：

- 不显示地扩大特征空间，这意味这我们不需要计算和存储特征的高维表示
- 对于n个训练样本，我们计算的核函数值C_n^2相比于许多高维度的数据扩展来说已经很小，尤其是某些扩展后维度为无限的数据

9.4 多分类下的 SVM

目前我们的讨论仅局限于二分类。虽然 SVM 基于的分离超平面的概念并不自然的适用于两个以上的类别，但是目前存在一些扩展分类的方法

9.4.1 一对一分类

设$K > 2$个类别，我们构造C_n^2个 SVM 来对这个数量的类别进行**一对一 (one-versus-one)** 分类，最终以最多被分类的类为最终分类。这个方法有些繁琐

9.4.2

一对全 (one-versuis-all) 方法，也称为**一对余 (one-versus-rest)** 方法，一共拟合K个 SVM 模型，每个模型将一个类别 (编码为+1) 和其他所有类别 (编码为-1) 进行比较。令$\beta_{0k}, \beta_{1k}, \cdots, \beta_{pk}$表示第$k$类 (+1) 与其他类 (-1) 拟合 SVM 得到的参数，设实验值x^*，则输出函数$f(x^*) = \beta_{0k} + \beta_{1k}x_1^* + \cdots + \beta_{pk}x_p^*$最大的分类为最终分类

> 这种方法在类别不平衡时表现不佳

9.5 SVM 与逻辑斯蒂回归的关系

支持向量分类器的输出函数可以重写为：

$$\underset{\beta_0,\beta_1,\ldots,\beta_p}{minimize} \left\{ \sum_{i=1}^{n} \max[0, 1 - y_i f(x_i)] + \lambda \sum_{j=1}^{p} \beta_j^2 \right\}$$

其中λ是一个非负调谐参数，当λ足够大时可以容纳更高的偏差。其中，小λ值与支持向量分类器中的C值相等。注意，$\lambda \sum_{j=1}^{p} \beta_j^2$是我们之前提到过的正则化项

这种损失+乘法的组合重复出现了多次：

$$\underset{\beta_0,\beta_1,\ldots,\beta_p}{minimize} \{L(\mathbf{X}, \mathbf{y}, \beta) + \lambda P(\beta)\}$$

其中L是用于刻画模型与训练数据的损失函数，量化了以β为参数的模型对数据(X, y)的拟合程度，$\lambda P(\beta)$是对这组参数的惩罚项

> 岭回归和 lasso 回归都采用了上面的形式

对于支持向量分类器的损失函数，我们可以写为：

$$L(\mathbf{X}, \mathbf{y}, \beta) = \sum_{i=1}^{n} \max[0, 1 - y_i(\beta_0 + \sum_{j=1}^{p} \beta_j x_{ij})]$$

这也被称为**合页损失 (hinge loss)**。合页损失与逻辑斯蒂回归使用的损失函数非常近似

支持向量分类器的特性是位于分类边缘的正确分类数据对模型没有影响，损失函数为 0；而逻辑斯蒂回归的损失函数在任何地方都不为 0，即是远离决策边界的观测值影响非常小。因此二者常给出相似的分类结果

> 当类间分类较好时，支持向量机的效果好于逻辑回归；在更多重叠的区域中，逻辑回归更受青睐

SVM 的超参数C在它出现初期被认为不重要，但是现在已经证明它用于控制模型的偏差-方差权衡，有必要通过交叉检验等方式进行验证

SVM 并不以使用核函数扩大特征维度来显得特殊，事实上，我们之前介绍的许多方法都可以用核函数。只是由于历史原因，SVM 使用非线性核的情况广泛的多

支持向量回归 (support vector regression) 是 SVM 在回归领域中的扩展，它寻求使不同类型损失最小化的系数，只有绝对值大于某个常数的残差对损失函数有贡献

第十章 深度学习

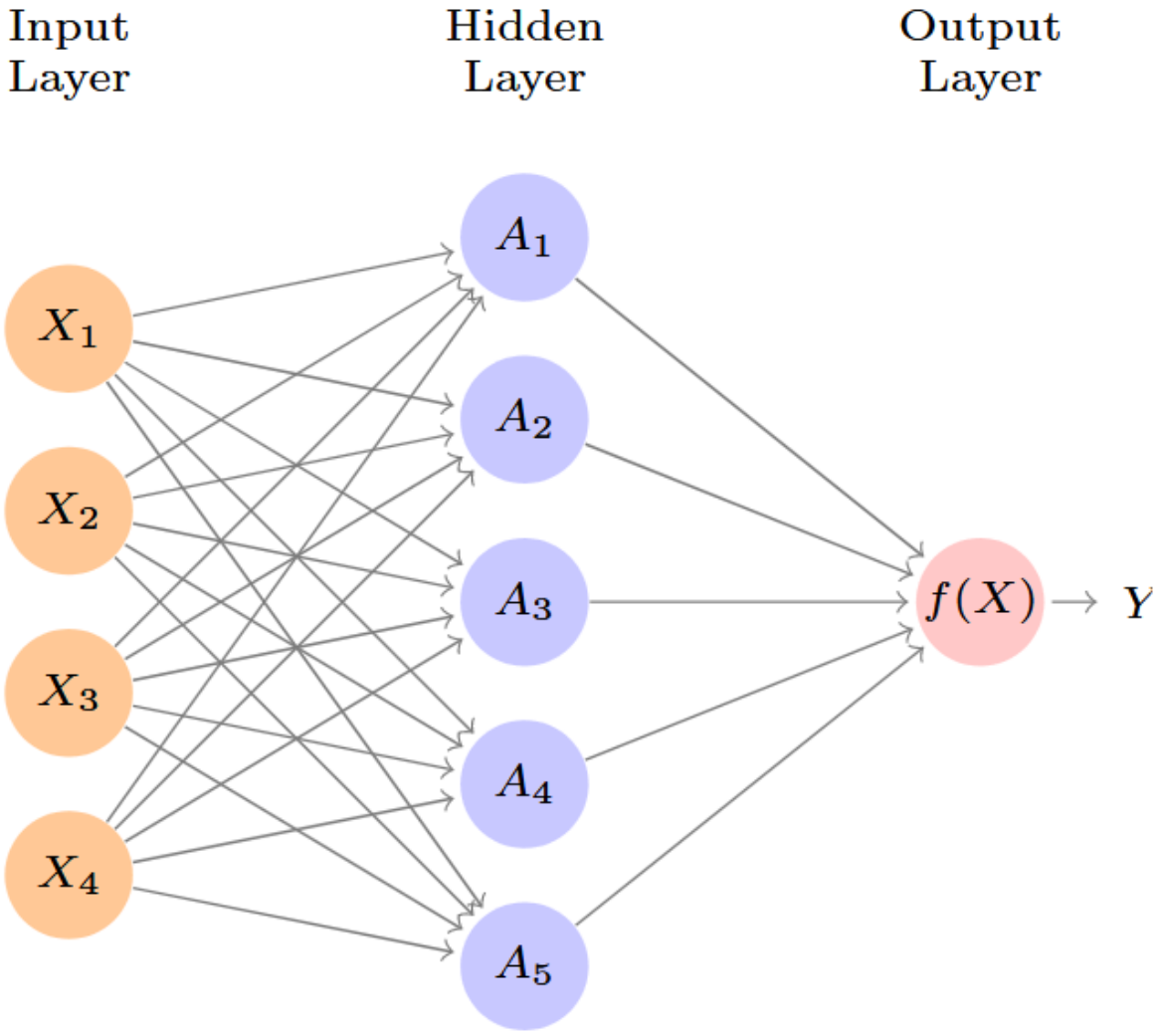
深度学习 (deep learning) 的基石是**神经网络 (neural network)**。神经网络在上世纪 80 年代被提出，中间由于 SVMs、Boosting 和随机森林等方法而沉寂、2010 年之后深度学习带来了新的技术，得益于现代更加庞大的数据集，深度学习成为了当下最活跃的研究对象

本章讨论用于图像领域的卷积神经网络 (CNNs)、用于时间或其他序列的循环神经网络 (RNNs) 等基础知识，并使用 `Python torch` 包来实现这些模型

10.1 单层神经网络

神经网络接受一个包含p个变量的输入向量$X = (X_1, X_2, \ldots, X_p)$并构造一个非线性函数$f(X)$来预测响应变量$Y$

神经网络与之前的非线性模型的区别在于它的结构，一个**前馈神经网络 (feed-forward neural network)** 的结构如下：



用神经网络的术语描述，4 个特征$X_1, \ldots, X_4$组成了**输入层 (input layer)**，输入层被连接到K个**隐藏单元 (hidden units)**。神经网络的数学描述如下：

$$
\begin{aligned}
f(X) &= \beta_0 + \sum_{k=1}^{K} \beta_k h_k(X) \\
&= \beta_0 + \sum_{k=1}^{K} \beta_k g(\omega_{k0} + \sum_{j=1}^{p} \omega_{kj} X_j)
\end{aligned}
$$

其中β_0是偏置项，β_k是输出层的权重，$h_k(X)$是第k个隐藏单元的输出。g表示**激活函数 (activations)**，ω_{k0}是第k个隐藏单元的偏置项，ω_{kj}是第k个隐藏单元与输入X_j的权重

我们这样理解这个单层神经网络：

1. 首先，对于K个激活函数A_k，在隐藏层中，我们计算每个特征的输入：

   $$A_k = h_k(X) = g(\omega_{k0} + \sum_{j=1}^{p} \omega_{kj} X_j)$$

   其中的每个$g(z)$都是非线性的激活函数，它们被事先设置好了。
2. 当我们得到A_k，作为之前特征的一个转换时 (就像第七章的样条)，然后将K个激活值输入到输出层，输出：

$$f(X) = \beta_0 + \sum_{k=1}^{K} \beta_k A_k$$

这是一个线性回归模型，拥有K个输入。所有的参数$\beta_0, \ldots, \beta_K$和$\omega_1 0, \ldots, \omega_{Kp}$都需要用训练数据估计

在早期的神经网络中，**sigmoid 激活函数 (sigmoid activation function)** 比较常用：

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

这实际上同逻辑回归中使用的函数相同，它将一个线性函数转换到 0 和 1 之间的一个概率。现代神经网络常用的激活函数是**修正线性单元 (rectified linear unit, ReLU)**，形式如下：

$$g(z) = (z)_+ = \begin{cases} 0, & \text{if } z < 0 \\ z, & \text{otherwise} \end{cases} = \max(0, z)$$

尽管 ReLU 激活函数有一半的值是 0，但是引入偏置项ω_{k0}可以改变这个阈值

神经网络得名于大脑的神经元，激活函数A_k接近 1 时，认为这个神经元被激发，就类似大脑中的活跃神经元；当激活值接近 0 时，模拟大脑中不活跃的神经元

> 因此，Sigmoid 激活函数适合模拟神经元，因为它将输入值映射到 $[0, 1]$ 区间上

非线性激活函数$g(\cdot)$非常重要，没有它$f(X)$就会退化为一个简单的关于X_i的线性模型。非线性激活函数使我们能够捕捉这些特征之间复杂的相互作用和非线性关系

拟合神经网络需要估计未知参数，他们通常用最小方差损失估计：

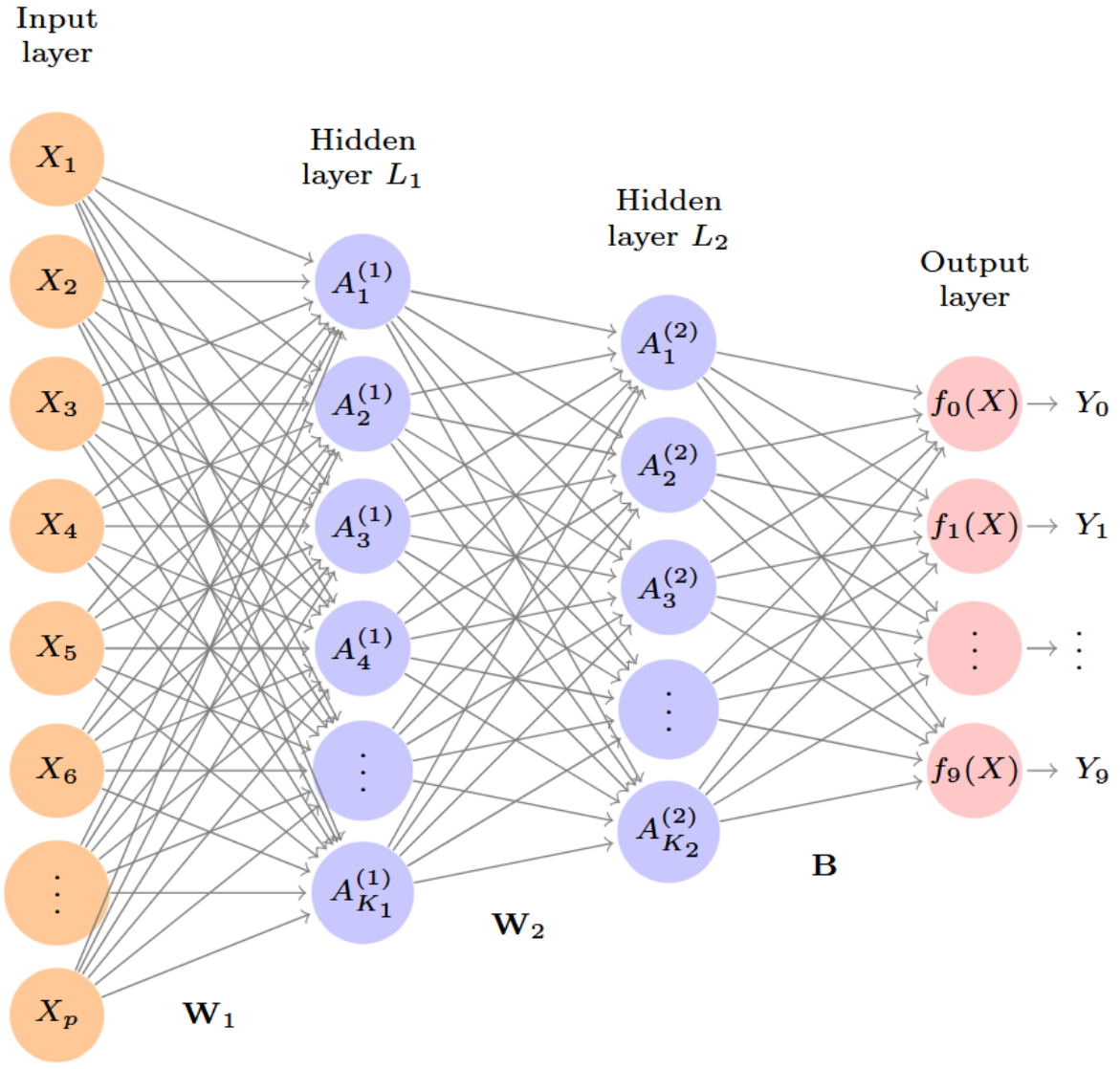
$$\text{minimize} \sum_{i=1}^{n} (y_i - f(x_i))^2$$

10.2 多层神经网络

现代神经网络通常具有一个以上的隐含层，每个层大小适中。这里理论上的大量单元的单隐藏层要好

本节使用著名的手写数字数据集 `MNIST` 来研究多层神经网络。这个数据集的特点是拥有 6 w 张训练照片和 1 w 张测试照片，每张照片有$p = 28 \times 28 = 784$个像素，每个像素是一个取值在$[0, 255]$的灰度，表示该像素的颜色有多深，响应变量$Y = (Y_0, \ldots, Y_9)$是是个哑变量，使用了独热编码

下面是一个两层神经网络，用于执行数字分类任务：



> 将截距 (机器学习中称为**偏差 (biases)**) 计算在内，此神经网络拥有 235,146 个参数 (机器学习中称为**权重 (weights)**)

它的特点是：

1. 有两层，分别有$L_1 = 256$个$L_2 = 128$个单元
2. 拥有十个定性输出变量，这些变量之间高度相关。在**多任务学习 (multi-task learining)** 中，单个神经网络可同时预测不同的想要，这些响应都会影响隐藏层的形成
3. 为多任务学习制定了相应的损失函数

第一个隐藏层的激活函数如下：

$$A_k^{(1)} = h_k^{(1)}(X) \qquad = g(\omega_{k0}^{(1)} + \sum_{j=1}^{p} \omega_{kj}^{(1)} X_j)$$

其中$k = 1, 2, \ldots, K_1$。第二个隐藏层接受输入$A_k^{(1)}$，激活函数如下：

$$A_l^{(2)} = h_l^{(2)}(X) \qquad = g(\omega_{l0}^{(2)} + \sum_{k=1}^{K_1} \omega_{lk}^{(2)} A_k^{(1)})$$

其中$l = 1, \ldots, K_2$

> 第二层隐藏层也是X的函数，这表明多层神经网络能够得到相当复杂的X交互项的输出

图示记号$\mathbf{W}_1$表示从输入层到第一个隐藏层L_1的整个权重矩阵。这个矩阵将会包含$785 \times 256 = 200960$个元素，785而不是784是因为计入了截距或称偏差项 (机器学习社区)

> 计算方式$(p+1) \cdot L_1$

在输出层，我们计算：

$$\begin{aligned} X_m &= \beta_{m0} + \sum_{l=1}^{K_2} \beta_{ml} h_l^{(2)}(X) \\ &= \beta_{m0} + \sum_{l=1}^{K_2} \beta_{ml} A_l^{(2)} \end{aligned}$$

这是一个线性模型。其中$m = 0, 1, \ldots, 9$，这个权重矩阵则包括$129 \times 10 = 1290$个参数

对于定量的输出，为了保证我们分类的输出情况类似于概率形式，我们还必须对输出层进行 softmax 激活：

$$f_m(X) = Pr(Y = m|X) = \frac{e^{Z_m}}{\sum_{l=0}^{9} e^{Z_l}}$$

为了训练这个网络，我们估计能使最小化：

$$-\sum_{i=1}^{n} \sum_{m=0}^{9} y_{im} \log(f_m(x_i))$$

负数多项对数似然值的参数，这也被称为**交叉熵 (corss-entropy)**，这是二元分类逻辑回归准测的推广。其中，y_{im}是一个指示函数

> 熵的计算公式是：$H(p) = -\sum_x p(x) \log p(x)$，表示一个随机变量的不确定性；交叉熵被定义为衡量用分布q表示真实分布p所需的平均编码长度：$H(p, q) = -\sum_x p(x) \log q(x)$

神经网络的参数如此之多，以至于 6 w 个训练值不足以估计所有的参数。因此，为了避免过拟合，我们使用了两种形式的正则化：

- 岭正则化，类似于岭回归
- **Dropout**正则化

10.3 卷积神经网络

本章我们用 `CIFAR 100` 数据集研究**卷积神经网络 (Convolutional Neural Networks, CNNs)**。这个数据集包含分为 100 个细分子类，20 个超类的动物图像，每个图像包含32×32个像素，每个像素由 3 个 8 bits 数组成，分别表示红绿蓝，这些值存储在一个叫做**特征图 (feature map)** 的三维数组中，前两维表示空间信息，即该像素的高度和宽度，第三维表示**颜**

色通道 (channel) ，这是一个三维数组，分别包含 RGB 的取值$[0, 255]$。数据包含 5 w 训练样本和 1 w 测试样本

CNNs 被发明用于这种任务，它通过识别图像任何位置的特征或者模式来区分每个对象的分类。它首先识别输入图像的低级特征，如小的边缘、颜色块等，然后用这些低级特征组合为高层次特征，如动物的不同部位，根据这些高层特征的存在和缺失来提高输出正确类的概率

CNNs 有两个特殊的隐藏层，分别是**卷积 (convolution)** 层和**池化 (pooling)** 层。卷积层在图像中搜索小型局部模式，不同层次的卷积层识别的模式不同；池化层则对卷积层的特征图进行**降采样 (downsample)** 以获取最突出的信息，提高稳健性

> 池化的常见方法包括最大池化 (保留窗口内的最大值) 和平均池化 (保留窗口内的平均值)

10.3.1 卷积层

卷积层是由许多**卷积滤波器 (convolutional filter)** 组成的。卷积滤波器依赖于一个非常简单的操作，即卷积，它可以认为是重复相乘矩阵元素，然后将结果相加

假设一个4×3的图像：

$$\text{Original Image} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \\ j & k & l \end{bmatrix}$$

我们考虑一个2×2的卷积核 (也就是卷积滤波器)：

$$\text{Convolution Filter} = \begin{bmatrix} \alpha & \beta \\ \gamma & \delta \end{bmatrix}$$

则我们用滤波器卷积得到的结果是：

$$\text{Convolved Image} = \begin{bmatrix} a\alpha + b\beta + d\gamma + e\delta & b\alpha + c\beta + e\gamma + f\delta \\ d\alpha + e\beta + g\gamma + h\delta & e\alpha + f\beta + h\gamma + i\delta \\ g\alpha + h\beta + j\gamma + k\delta & h\alpha + i\beta + k\gamma + l\delta \end{bmatrix}$$

卷积核用于模式匹配和特征提取。若一个子矩阵的结构和卷积核很相似，则计算输出的值就较大；不同的卷积核，可以分别识别图像的不同特征。卷积核的局部作用和大小的灵活变化使它在识别图像上非常好用

> 卷积特征图的值大小表示相似性，是因为卷积核的卷积过程相当于两个矩阵的点乘，如$(a, b, c, d)^T \cdot (\alpha, \beta, \gamma, \delta)^T$。如果这个值较大，说明向量的方向约接近，即越相似

卷积特征图的输出高度等于输入高度减去卷积核高度加一，输出宽度等于输入宽度减去卷积核宽度加一

以竖向卷积核和横向卷积核为例，对于一副图像，它们分别生成凸显竖线特征和横线特征的特征图。在传统方法中，这是通过预定义的滤波器，如 Sobel 提取边缘，Gabor 提取纹理实现

的，需要人工组合多种滤波器。CNNs 采用不同的权重来使用滤波器，能够针对特定分类任务进行优化

以下是一些更详细的介绍：

- 对于一张 RGB 图，其是一个$32\times32\times3$的张量，我们同样设计一个$3\times3\times3$的卷积核，来对图像特征进行卷积，这样能得到 3 副30×30的 R、G、B 特征图，接着对这些特征图对应求和，就能综合 RGB 三种颜色特征，得到最终的一张特征图

  > 此时，颜色作为特征已经全部使用，将不会传入到后续的卷积层中

- 在 CNN 的第一层，通常拥有K个卷积核，每个卷积核都会与输入图像进行卷积，共产生K个 2 D 图。这K个 2 D 图堆叠在一起，就形成了一个新的 3 D 特征图，然后继续应用卷积核

  > 卷积产生的 2 D 图就好像传统神经网络中隐藏层的激活值，不过区别在于这里的激活值是一个矩阵，保留了空间信息

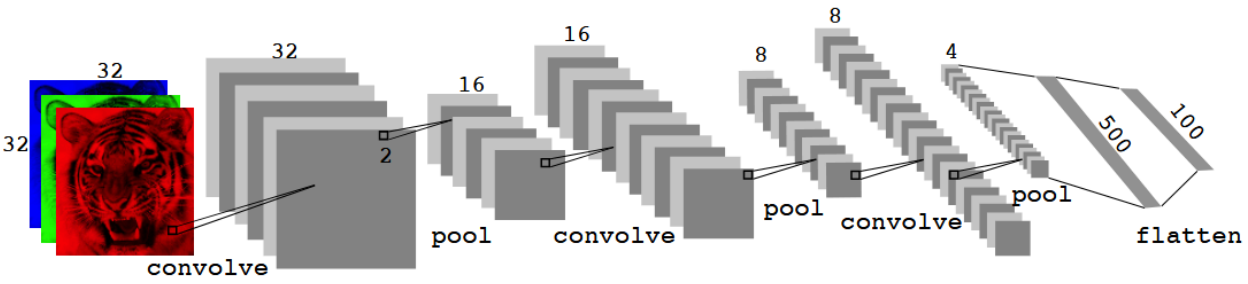
- 为了引入非线性操作 (相对于卷积的线性操作)，卷积后通常施加$ReLU$激活函数以去掉负数值，增加稀疏性便于计算。这样看来，$ReLU$就像一个特征过滤器，对于高响应的值保留，低响应的则置零。因此又将它视为一个**检测层 (detector layer)**

10.3.2 池化层

池化层在 CNN 中执行浓缩的功能。具体来说，**池化 (pooling)** 保留原有特征值的重要特征，降低后续运算量，例如：

$$\text{Max pool}\begin{bmatrix}1&2&5&3\\3&0&1&2\\2&1&3&4\\1&1&2&0\end{bmatrix}\rightarrow\begin{bmatrix}3&5\\2&4\end{bmatrix}$$

10.3.3 卷积神经网络架构



一个典型的 CNN 包含多个卷积层和池化层。其中，每个卷积层包含多个卷积核，生成多个 2 D 图像；池化层提取这些图像的重要特征 (降低图像的宽度高度维数) 后，再次进行卷积，为了补偿池化层降维的部分，这里的卷积核通常比之前更多。经过多层的特征提取后，当空间维度足够低时，将所有的像素扁平化，视为单独的特征，接入 MLP 进行非线性操作，最后输出结果

> 示例图中卷积后图像空间维度不变，是因为使用了**填充 (padding)** 方法
> 有时候在池化层前接入多个卷积层，可以在不降低空间分辨率的前提下提取更复杂的特征。这等效于使用更大感受视野的卷积核

CNNs 的参数如此多，为了避免过拟合，我们需要调优的超参数包括：

- 卷积核个数，大小，层次
- 池化的方式，步长等
- 全连接层大小、Dropout 方法、L_2正则化

10.3.4 数据增强

数据增强 (data augmentation) 是一种增大数据集的技术。CNNs 一来大规模数据，但真实的标注数据有限，为了减少过拟合，我们可用数据增强的方法扩大数据集。常见的有：

1. 几何变换。如旋转、平移、缩放等
2. 颜色变换。如亮度、对比度和噪声等
3. 伪造数据。如随机遮挡图像、混合图片和裁剪拼接等

数据增强可以看做是一种正则化的形式

10.3.5 预训练分类器

CNNs 的核心是学习卷积滤波器，这一学习过程通过反向传播学习实现。当 CNN 在大规模数据集上训练后，他学习到的卷积滤波器可以快速泛化到其他视觉任务上。这是因为低级和中级特征在不同任务中是通用的，只有最后的分类层要根据任务具体调整

迁移学习是指在大数据集上训练好的 CNN 作为特征提取器，然后在小数据集上微调以适应小数据任务的过程

权重冻结 (Weight Freezing) 的意思是不更新 CNN 预训练好的前几层权重，只训练最后几层 (通常是全连接层)。这可以减少训练量，并避免过拟合

10.4 文档分类

文档分类时一种在工业和生活中常见的场景，例如对医学期刊评级、对新闻稿、邮件进行分类等。下面使用 `IMDb` 数据集演示，这是一个关于电影评价和最终得分的数据集，一部电影的评价可能是积极的或者消极的

每个评论的长度不同，用语不同 (俚语甚至非词语)，我们需要寻找方法来对文档进行**特征提取 (featurize)**

最常见和简单的方案是使用**词袋 (bag-of-words)** 模型。我们用一个单词的出现与否对文档进行评分；若一个文档包含 M 个单词，那意味着他们都是一个M维度的向量，其中 1 表示单词存在，0 表示不存在。我们通过限定最常用的 1 w 个词来限定向量维度，其中不在这些词的字典中的词将会被标记为如 `<UNK>` 的样式

通过这种方法构造的向量特征很多，但是作为训练集的矩阵非零元素也很多，这被称为**稀疏矩阵 (sparse matrix format)**，可以用特别的方法存储以节省空间

实验中使用 lasso 和两个隐藏层的二分类神经网络来进行预测，注意到二分类神经网络可视为一个非线性逻辑回归模型：

$$\begin{aligned}\log\left(\frac{\Pr(Y=1|X)}{\Pr(Y=0|X)}\right) &= Z_1 - Z_0 \\ &= (\beta_{10}-\beta_{00}) + \sum_{l=1}^{K_2}(\beta_{1l}-\beta_{0l})A_l^{(2)}\end{aligned}$$

> Softmax 函数具有冗余性，因为K分类问题只需要$K-1$组系数，最后一个可以通过概率相减得到

词袋模型只考虑的文章中是否出现单词，但是没考虑上下文，下面是另外两种常用的改进方案：

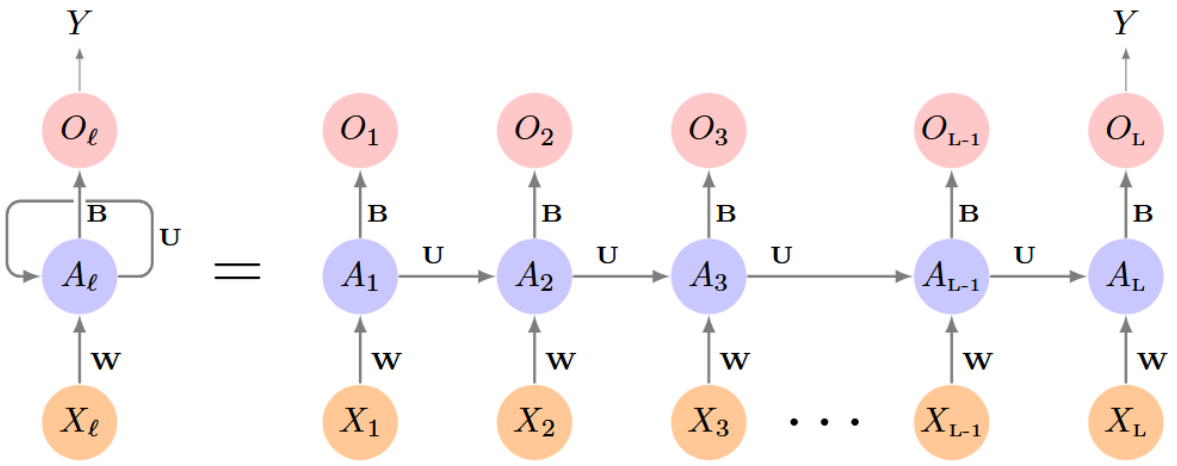
1. **bag-of-n-grams**模型。例如，我们可以每次将两对临近单词进行组合，联系上下文
2. 将文档视为一个序列，考虑单词出现的同时考虑上下文单词的位置

10.5 循环神经网络

许多数据实际上是序列化的，构建模型时需要特殊处理，例如：

- 文档，如书籍和电影评论。它们单词的顺序和相对位置捕捉了叙述主题和语气等特征
- 温度、降雨量等时间序列
- 金融时间序列
- 语音、录音等。文本转录和语言翻译需要考虑序列化
- 手写体

在**循环神经网络 (recurrent neural network, RNN)** 中，输入矩阵X就是一个序列。RNN 被设计用来处理这类有序数据，这就像 CNN 被设计得适用于图像数据一样。这些输入数据$X=(X_1,X_2,\ldots,X_L)$，元素之间的排列顺序和近邻程度可以衡量它们之间的关系。RNN 的输出可以是序列 (如文本生成)，也可以是标量 (分类)



对于数据序列$X = \{X_1, X_2, \ldots, X_L\}$，其中每个$X_l$都是一个向量，隐藏层神经元序列$\{A_l\}_1^L = \{A_1, A_2, \ldots, A_L\}$。用文本举例，则每个$X_l$都是一个单词的独热编码 (例如$X_l = (0, 0, 1, 0, 0)$)。RNN 每次接受一个数据序列中的向量$X_l$，并根据之前序列得到的激活值$A_{l-1}$输出当前序列的激活值$A_l$，同时输出一个预测值$O_l$。对于最后一次输出$O_L$，它是和我们预测的目标最接近的

具体来说，我们考虑输入序列中的一个向量X_l，它拥有p个元素$X_l^T = (X_{l1}, X_{l2}, \ldots, X_{lp})$，隐藏层由$K$个神经元组成：$A_l^T = (A_{l1}, A_{l2}, \ldots, A_{lp})$。我们可以用矩阵表示权重：

- 输入层的权重矩阵$\mathbf{W}$，维度$K \times (p+1)$
- 隐藏层到隐藏层的权重矩阵$\mathbf{U}$ 共有$K \times K$维度，其中每个元素u_{ks}表示从隐藏层中的神经元k到另一个隐藏层中的神经元s的权重
- 输出层的权重向量$\mathbf{B}$，有$K+1$个权重β_k到输出结果上

因此，隐藏层的计算公式 (激活值) 为：

$$A_{lk} = g\left(\omega_{k0} + \sum_{j=1}^{p} \omega_{kj} X_{lj} + \sum_{s=1}^{K} u_{ks} A_{l-1,s}\right)$$

每次的输出O_l可以计算为：

$$O_l = \beta_0 + \sum_{k=1}^{K} \beta_k A_{lk}$$

对于定量变量等可以接入 Sigmoid 激活函数

上面的计算式中，$g(\cdot)$是一个激活函数，类似于 ReLU。注意权重矩阵$\mathbf{W}, \mathbf{U}, \mathbf{B}$不是$l$的函数，它们在每次接受序列数据时保持一致。这是 RNNs 的特性，即**权重共享 (weight sharing)**

对于回归问题，每对观测值(X, Y)的损失函数为：

$$(Y - O_L)^2$$

其中$O_L = \beta_0 + \sum_{k=1}^{K} \beta_k A_{Lk}$。注意这里只用到了最后一次的输出$O_L$，而忽略了前面的输出。因为共享权重，每次的输入数据简介地对权重矩阵产生贡献。对于n对观测值(x_i, y_i)，我们的参数需要最小化：

$$\sum_{i=1}^{n} (y_i - o_{iL})^2 = \sum_{i=1}^{n} \left(y_i - \left(\beta_0 + \sum_{k=1}^{K} \beta_k g\left(w_{k0} + \sum_{j=1}^{p} w_{kj} x_{iLj} + \sum_{s=1}^{K} u_{ks} a_{i,L-1,s}\right)\right)\right)^2$$

为什么保留每次的输出O_l，尽管我们只用到了O_L？这是因为它们是必要计算的，而且有时候我们希望得到的结果也是一个变化的序列$\{O_1, O_2, \ldots, O_L\}$

10.5.1 用于文档分类的序列模型

使用独热编码来编码各类单词存在维度灾难的问题，因为每个单词都会表示为一个非常稀疏的向量。一种解决方案是使用**嵌入 (embedding)** 。词嵌入是指用一个低维度的向量 (维度为

$m << n$) 来表示之前的独热编码，这需要用到一个嵌入矩阵$E_{m\times n}$，其中每列表示第i个单词转换后的嵌入向量e_i，他们之间的关系满足：

$$\mathbf{e}_i = E\mathbf{v}_i$$

其中$\mathbf{v}_i$表示单词i的独热编码向量

为了获取矩阵E，我们首先需要有大规模标注的语料集，然后将E作为神经网络优化的一部分。在这种情况下，E就成为了**嵌入层 (embedding layer)**。或者我们可以在嵌入层中加入一个预训练的矩阵，也即是**权重冻结 (weight freezing)** 的一个过程。两个广泛使用的预训练矩阵是 `word 2 vec` 和 `GloVe`，它们是主成分分析 PCA 的变体，要求同义词需要出现在映射空间的附近

下一步，我们将每个文档限制到最后的L个单词，然后将$< L$的文档前置补 0，这样每个文档都是由L个向量$X = \{X_1, X_2, \ldots, X_L\}$组成的序列了，每个$X_l$拥有$m$个分量

我们用基础 RNN 拟合这个模型，它的参数如下：

- 常规的权重矩阵W，拥有$K\times(m+1)$个参数
- 矩阵U拥有$K\times K$个参数
- 向量B拥有$2(K+1)$个参数 (2 分类问题下)
- 如果要训练嵌入矩阵E，它有$m\times D$的参数，D表示词典单词数，是最大的一个矩阵

实验表明，简单 RNN 的训练正确率为76%。一种改进的 RNN 模型是**长短期记忆模型 (long term and short term memory， LSTM)**，一定程度上克服了简单 RNN 的梯度下降和爆炸问题。LSTM 模型在此数据集上的性能提高到了87%，与词袋模型88%相当

对 LSTM 进行调参是一个很花时间的工作。RNNs 是一系列模型的基础，在此基础上，一个领先的 RNNs 配置在 `IMDb` 数据集上报告了95%以上的准确率。本书未提及细节

10.5.2 时间序列预测

下面用一份纽约股票交易所的交易数据来演示时间序列预测。它的三份时间序列如下：

- 对数交易量。表示当天交易流通股票相对于过去 100 天交易量的移动平均值的对数，计算公式为$\log(\frac{\text{当日交易量}}{\text{100天平均交易量}})$
- Dow Jones 指数汇报。表示连续交易日之间 Dow Jones 工业指数对数值的差异，计算公式为$\log(\text{today}) - \log(\text{yesterday})$
- 对数波动率，表示每日价格变动绝对值的对数

日期t的观测值(v_t, r_t, z_t)分别代表对数交易量，道琼指数和对数波动率，共有$T = 6051$个这样的观测对。在这种情况下，观测值们倾向于拥有相似的表现，这与其他训练数据之间相互独立的假设不同。我们假设观测对$(v_t, v_{t-l}$，其中l表示l天的**滞后 (lag)**，获得连续的这样的观测对，我们可以计算它们之间的自相关性

> 时间序列的自相关性使我们能够直接利用它的历史数据来预测未来的数据

我们希望通过v_t的过去值$v_{t-1}, v_{t-2}, \ldots$来预测v_t，同时也要考虑r_{t-1}, r_{t-2}和$z_{t-1}, z_{t-2}, \ldots$等历史数据。这与文本预测不同的是，我们只有一条时间序列 (不同于文本的n条评论时间序列)，因此我们考虑利用滞后L来划分历史数据：

$$X_1 = \begin{pmatrix} v_{t-L} \\ r_{t-L} \\ z_{t-L} \end{pmatrix}, X_2 = \begin{pmatrix} v_{t-L+1} \\ r_{t-L+1} \\ z_{t-L+1} \end{pmatrix}, \ldots, X_L = \begin{pmatrix} v_{t-1} \\ r_{t-1} \\ z_{t-1} \end{pmatrix}$$

我们的预测目标$Y = v_t$。对于时间t的变化，我们的数据可以划分出相当多的训练数据对。注意L在这里是一个关键参数，可能需要通过调优获得。用 RNN 拟合的模型$R^2 = 0.42$，是一个不错的效果

刚才的 RNN 模型与传统的**自回归模型 (autoregression, AR)** 有许多共同指出，我们将历史数据改造为响应值向量和设计矩阵：

$$y = \begin{bmatrix} v_{L+1} \\ v_{L+2} \\ \vdots \\ v_T \end{bmatrix}, M = \begin{bmatrix} 1 & v_L & v_{L-1} & \cdots & v_1 \\ 1 & v_{L+1} & v_L & \cdots & v_2 \\ 1 & v_{L+2} & v_{L+1} & \cdots & v_3 \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & v_{T-1} & v_{T-2} & \cdots & v_{T-L} \end{bmatrix}$$

这与用滞后值来预测当前值的思路一致，我们用y和M拟合回归模型：

$$\hat{v}_t = \hat{\beta}_0 + \hat{\beta}_1 v_{t-1} + \beta_2 v_{t-2} + \cdots + \hat{\beta}_L v_{t-L}$$

这被称为顺序L自回归模型，缩写为 AR (L)，将特征改为包括另外两个时间序列的向量，我们就能得到 AR 版本的预测模型，其$R^2 = 0.41$，仅比 RNN 少0.01

与 RNN 类似，AR 也接受$L = 5, p = 3$的数据，但是它将L个元素同等对待，作为一个$L \times p$的预测变量的向量，这个过程在神经网络中称为**展平 (flatening)**。RNN 理论上能够记住前面每次预测的信息，并作用到后续的预测结果上，但实际上受到梯度的限制

现代常用的手段是扩展 AR 模型，如将滞后变量作为输入向量，输入到一个普通神经网络中，增强模型灵活性；或者增加模型的特征，如星期，可用独热编码表示；也可以用 RNN 的改进版本 LSTM 来预测

10.5.3 RNNs 的总结

上面的讨论仅仅涉及了 RNNs 的很小部分，RNN 的扩展性非常强

一维卷积神经网络可以用于处理序列，例如，我们将单词表示为嵌入空间的向量组，并用一维 CNN 滑动处理，也能识别短语和时间序列的趋势

在 RNN 中添加多个隐藏层，可以增强模型的表达能力和学习能力，例如第一个隐藏层的输出序列A_l再输入到下一个隐藏层中；还可以扩展为**双向 RNN (bidirectional RNNs)**，从两个方向扫描序列，缓解梯度问题

在语言翻译中，目标也是一个词序列，但是它使用的语言与输入序列的语言不同。在**序列到序列 (Seq 2 Seq)** 学习中，隐藏单元被认为能够捕捉句子的语义含义。语言建模和翻译中的一些重大突破源于这类 RNN（循环神经网络）相对较近的改进

> 这种模型能解决$n \to m$的问题

10.6 何时使用深度学习

深度学习在现代的多种任务中取得了相当好的性能，真正成为了当代的一项革新技术。然而，它本质上是一个黑箱模型，解释性相比我们之前学习的方法差了很多

尽管深度学习在恰当的参数下表现得十分优异，但是这往往十分耗时。**Occam's razor**指出，当模型的效果相近时，选择最简单的那个

通常来说，当训练集的样本量非常大，且模型的可解释性不重要时，我们选择深度学习模型

10.7 神经网络的拟合

我们从简单神经网络开始。对于 10.1 节的简单神经网络，我们的目的是最小化参数$\beta = (\beta_0, \beta_1, \dots, \beta_K)$和$\omega_{k0}, \omega_{k1}, \dots, \omega_{kp}$在$n$对观测值$(x_i, y_i)$下的损失函数：

$$\underset{\{\omega_k\}_1^K}{\text{minimize}} \frac{1}{2} \sum_{i=1}^{n} (y_i - f(x_i))^2$$

其中：

$$f(x_i) = \beta_0 + \sum_{k=1}^{K} \beta_k g(\omega_{k0} + \sum_{j=1}^{p} \omega_{kj} x_{ij})$$

> 损失函数在参数上是非凸的，因此存在多个解。例如，对于最小值，我们可能得到**局部最小值 (local minimum)** 或者 **全局最小值 (global minimum)**

为了避免最小化损失函数过程中的过拟合，常采用下面两种方法：

- **慢学习 (slow learning)** 缓慢地进行**梯度下降 (gradient descent)** 过程，在探测到过拟合时停止优化
- 正则化在损失函数中添加惩罚项

假设我们的所有参数都放在一个向量θ中，则上面的损失函数改写为：

$$R(\theta) = \frac{1}{2} \sum_{i=1}^{n} (y_i - f_\theta(x_i))^2$$

通过上式，我们很容易用梯度下降的方法来获取一个全局 (局部) 最小值作为参数向量的选择：

1. 猜测θ的一个子集元素θ^0，设置时间$t = 0$
2. 迭代，直到上式下降到合适的一个值

1. 找到一个向量δ作为θ的一个微小变化值，使得$\theta^{t+1}=\theta^t+\delta$能使优化函数下降，即$R(\theta^{t+1})<R(\theta^t)$
2. 修改$t\leftarrow t+1$

一般来说，我们在复杂问题中得到梯度下降的最小值是一个较优局部最小值

10.7.1 反向传播

为了找到最合适的θ向量，使得$R(\theta)$的值减小，我们定义目标函数的**梯度 (gradient)** 表示函数的偏导数向量：

$$\nabla R(\theta^m)=\frac{\partial R(\theta)}{\partial\theta}|_{\theta=\theta^m}$$

其中$\theta=\theta^m$表示当猜测完当前的θ后，再计算梯度值。梯度表示某个点$\theta=\theta^m$处函数增长最快的方向，然后我们希望减小$R(\theta)$的值，于是反方向移动θ：

$$\theta^{m+1}\leftarrow\theta^m-\rho\nabla R(\theta^m)$$

对于足够小的**学习率 (learning rate)**，这一步变动将会得到更小的$R(\theta)$值，即$R(\theta^{m+1})\leq R(\theta^m)$。当梯度向量为 0 时，我们达到了目标最小值

> 计算梯度的过程需要用到求偏导数的**链式法则 (chain rule)**

总的损失函数$R(\theta)$被分解为$R(\theta)=\sum_{i=1}^n R_i(\theta)=\frac{1}{2}\sum_{i=1}^n(y_i-f_\theta(x_i))^2$，对于每对观测值$(x_i,y_i)$，它的损失函数：

$$R_i(\theta)=\frac{1}{2}\left(y_i-\beta_0-\sum_{k=1}^K\beta_k g(\omega_{k0}+\sum_{j=1}^p\omega_{kj}x_{ij})\right)^2$$

为了简化上式，我们用$z_{ik}=\omega_{k0}+\sum_{j=1}^p\omega_{kj}x_{ij}$来代替激活函数$g(\cdot)$，则关于$\beta_k$的梯度：

$$\begin{aligned}\frac{\partial R_i(\theta)}{\partial\beta_k}&=\frac{\partial R_i(\theta)}{\partial f_\theta(x_i)}\cdot\frac{\partial f_\theta(x_i)}{\partial\beta_k}\\&=-(y_i-f_\theta(x_i))\cdot g(z_{ik})\end{aligned}$$

其中$g(z_{ik})$表示第k个隐藏层神经元的输出，这一项体现了第k个神经元对误差的贡献。 ω_{kj}的梯度：

$$\begin{aligned}\frac{\partial R_i(\theta)}{\partial\omega_{kj}}&=\frac{\partial R_i(\theta)}{\partial f_\theta(x_i)}\cdot\frac{\partial f_\theta(x_i)}{\partial g(z_{ik})}\cdot\frac{\partial g(z_{ik})}{\partial z_{ik}}\cdot\frac{\partial z_{ik}}{\partial\omega_{kj}}\\&=-(y_i-f_\theta(x_i))\cdot\beta_k\cdot g'(z_{ik})x_{ij}\end{aligned}$$

其中β_k表示输出层权重，调节残差分配到隐藏层神经元k的比例，激活函数的导数$g'(z_{ik})$衡量激活函数对输入的敏感度，输入值x_{ij}表示输入数据对隐藏层神经元k的贡献

反向传播 (backpropagation) 的起点是残差$(y_i-f(x_i))$，之后通过梯度逐层分解到每层上。通过逐层分解的方式，反向传播计算了所有参数的梯度，使得神经网络能够通过梯度下降优化

10.7.2 正则化和随机梯度下降

计算大量数据的梯度是非常麻烦的。在实践中，常使用**随机梯度下降 (Stochastic Gradient Descent, SGD)** 的方法进行加速。随机梯度下降不计算所有样本以进行反向传播，而是每次迭代只使用一个小样本，也称为**小批量 (minibatch)** 样本进行训练

考虑数字分类使用的双隐藏层神经网络，它的参数数量是训练数据的 4 倍，因此必须通过正则化来避免过拟合。我们引入L_2正则化的交叉熵公式如下：

$$R(\theta;\lambda)=-\sum_{i=1}^{n}\sum_{m=0}^{9}y_{im}\log(f_m(x_i))+\lambda\sum_j\theta_j^2$$

这里的λ可以通过设置一个较小值或者通过验证集方法找到。对于不同层的权重组，也可以使用不同的λ值；也可以引入 lasso 回归使用的L_1正则化项

> 文中的实验指出，SGD 对模型的作用效果近似于施加一个二次的L_2正则化项

SGD 在训练过程中将整个数据集划分成多个小批量，并对每个小批量计算损失函数和梯度，更新模型参数。当整个训练集被遍历时，我们说完成了一个**epoch**，通常训练多个 epoch 以获得最佳模型

当训练到一定的 epoch 后，使用验证集方法表明模型的验证误差出现了增加，表明潜在的过拟合发生。因此，**早停 (early stopping)** 策略常常被应用于及时停止迭代，防止过拟合

10.7.3 丢弃学习

丢弃 (dropout) 是一种正则化技术，它受到随机森林的启发，每次拟合模型时随机移除一层神经元中的某个部分Φ，这些神经元不参与计算，不影响梯度更新，因此每次反向传播仅对未被丢弃的神经元进行梯度更新。进行测试时则不进行丢弃，而对所有神经元的输出乘以丢弃概率p以保持期望输出一致

丢弃学习可以防止神经元变得过拟合。在实际应用中，“丢弃”的操作是将某些神经元激活值设置为 0 实现的

10.7.4 网络调优

在神经网络的训练过程中，我们有许多需要考虑的参数：

- 隐藏层的数量和每层的神经元数量。现代观点认为，每层隐藏层可以容纳大量的神经元
- 正则化参数。包括丢弃率Φ，正则化强度λ等，它们通常在不同隐藏层中单独设置
- SGD 的细节。包括批量的大小，训练的 epochs 数，可能被应用的数据增强的细节

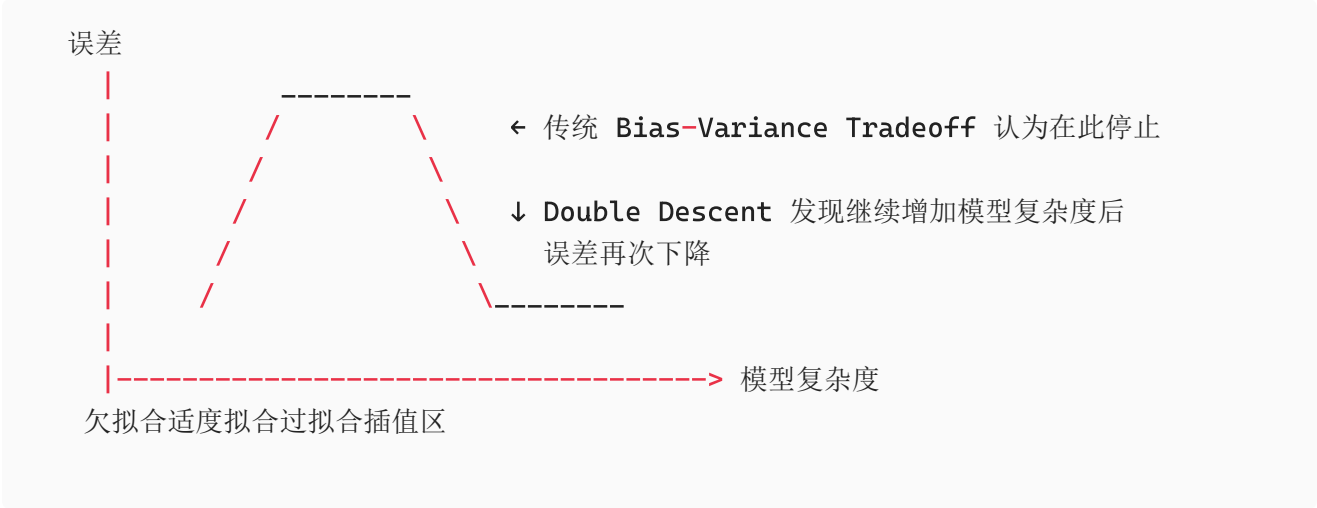
我们需要对这些超参数进行调优，因为它们的设置直接影响模型性能。网络调优的目的即是寻找最优策略提高模型准确率，减少过拟合和欠拟合并优化计算效率

10.8 差值和双下降

本书讨论的一个核心观点是偏差-方差权衡。这个观点认为，在训练误差接近零 (这一区域被称为**插值 (interpolate)** 区) 时，测试误差将会增大。一般来说，我们期望训练误差呈现下降的趋

势，而测试误差则会形成一个 U 型曲线

然而，事实证明，在某些特定的设置中，插值训练的统计学习方法可以表现得很好，这种现象就是**双下降 (double descent)**。



原书用自然样条进行了实验。结果表明，当自由度与训练数据相同时，拟合曲线震荡程度很大；而当自由度远超过训练数据时，拟合曲线的震荡反而减小，即出现了双下降现象。大量参数的神经网络也会出现双下降现象，这是因为 SGD 等技术倾向于选择一个平滑的插值模型，在这类问题上具有良好的测试集性能

> 自然样条选择过程中，提到了**最小范数解 (minimum norm solution)** 的概念，即在方程个数少于变量的方程组中，选择欧几里得范数$l = \sqrt{\sum_{i=1}^{n} x_i^2}$最小的解

下面是一些值得注意的点：

- 双下降与偏差-方差权衡不矛盾。在书中提到的例子中，双下降区域是因为横轴的显示为样条基函数的数量，这不能完全真实地表示模型的灵活性，因为在插值区域，后续模型的最小自然范数样条比之前更低，表明方差的再次下降
- 正则化方法同样适用。正则化方法不需要插值数据，就能获得优秀的泛化能力，获得更好的测试误差。正则化方法避免了双下降，且双下降不是所有学习方法的普遍特性
- SVM 等方法能够达到零训练误差，且泛化依然较好，是因为它们找到的是最小范数解。最小范数解兼具平滑性和低复杂度，但是零训练误差并不总是最优
- 零训练误差的效果取决于数据的信噪比。对于高信噪比的数据，训练数据噪声很少，做到零误差是可能的；但对于低信噪比的数据，模型非常容易过拟合。避免过拟合的方法包括正则化的早停等

第十一章 生存分析和删失数据

生存分析 (survival analysis) 的研究对象是尚未发生的活动。例如，在为其五年的医学研究中，部分患者可能存活超过 5 年，我们希望研究这些未被保留的数据，它们称为**删失数据 (censored data)**，这些数据携带了有用的信息；或者，对于取消订阅的客户，在我们研究的时段内没有取消，我们不知道将来何时会取消，则这些客户取消订阅事件的数据也是删失的

11.1 生存时间和删失时间

对于每个研究对象，我们可以得到它们的**生存时间 (survival time)** 和 **删失时间 (censoring time)**，分别记为T和C，其中删失时间又称为**失败时间 (failure time)** 或者 **事件时间 (event time)**。生存时间意味着此时我们感兴趣的事情发生了，删失时间则意味着此时数据丢失 (例如病人退出或未复发)

定义随机变量：

$$Y = \min(T, C)$$

为事件的发生，指示变量：

$$\delta = \begin{cases} 1, & if\ T \leq C \\ 0, & if T\ > C \end{cases}$$

现在我们可以获得观测变量对(Y, δ)

11.2 删失的具体描述

为了分析删失数据，我们首先要对数据删失的原因进行假设。例如，患者提前退出研究的可能是因为它病得非常厉害 (若无此假设，可能高估生存时间)，男性的患者在重病时更倾向于退出研究 (若无此假设，可能错误分析性别影响)

生存分析的一个重要假设是删失机制与事件发生独立，即删失不会倾向于某些数据 (大体重被删失违反这一假设)。判断独立性主要取决于数据搜集的过程，后续研究假设独立性成立

本章重点介绍右删失，即$T \geq Y$时发生删失，这保证删失发生时间至少与观测时间Y一样大。左删失和区间删失的思想同上

11.3 The Kaplan–Meier 生存曲线

生存曲线 (survival curve 或 **生存函数 (survival function)** 的定义如下：

$$S(t) = \Pr(T > t)$$

这是个递减函数，量化了在时间t时幸存下来的概率。$S(t)$的值越大，表明t之前发生事件的概率越小

用 `BrainCancer` 数据集举例，当我们试图估计$S(20)$时，我们计算$t = 20$时仍然存活患者的比例。然而，我们若用下面两种方式来处理删失患者：

1. 将删失患者认为不存活。可能低估了生存率
2. 只考虑非删失患者，即缩小分母。没有用上删失患者的信息

采用下面的方法处理删失数据。数据集中，在观测时间内死亡的人数记为K，则对于所有的死者，它们的唯一死亡时间点为$d_1 < d_2 < \cdots < d_k$，记录q_k为第k个时间点时的死亡人数。对于$k = 1, \ldots, K$，令r_k表示在某个死亡时间点d_k前存在于研究中的人数，它们称为风险患者，风险患者的集合被称为**风险集 (risk set)**

根据总概率定律：

$$\begin{aligned}\Pr(T > d_k) = \Pr(T > d_k | T > d_{k-1}) \Pr(T > d_{k-1}) + \\ \Pr(T > d_k | T \leq d_{k-1}) \Pr(T \leq d_{k-1})\end{aligned}$$

而事件$\{T > d_k | T \leq d_{k-1}\}$不可能发生，因此：

$$S(d_K) = \Pr(T > d_k) = \Pr(T > d_k | T > d_{k-1}) \Pr(T > d_{k-1})$$

代入公式$S(t) = \Pr(T > t)$，我们有：

$$S(d_K) = \Pr(T > d_k | T > d_{k-1}) S(d_{k-1})$$

这意味着：

$$S(d_k) = \Pr(T > d_k | T > d_{k-1}) \dots \Pr(T > d_2 | T > d_1) \Pr(T > d_1)$$

为了估计每一项的值，我们有：

$$\hat{\Pr}(T > d_j | T > d_{j-1}) = (r_j - q_j)/r_j$$

这表示在时间点d_j后，幸存人数的比例，则**Kaplan-Meier 估计器 (Kaplan–Meier estimator)**估计的生存曲线如下：

$$\hat{\mathrm{S}}(d_k) = \prod_{j=1}^{k} \left(\frac{r_j - q_j}{r_j} \right)$$

对于在两个死亡时间点d_k, d_{k+1}中的时间t，我们令$\hat{S}(t) = \hat{S}(d_k)$，因此 Kaplan-Meier 生存曲线呈阶梯状

11.4 对数秩检验

为了检验存在删失数据的两组生存曲线的风险是否有显著差异 (不能用平均值比较)，我们引入**对数秩检验 (log-rank test)**，它是一种按时间顺序检验事件的方法

设两组样本在死亡时间d_k前的未死亡且在试验中患者数分别为r_{k1}, r_{k2}，d_k时的死亡人数分别为q_{k1}, q_{k2}。对于每个死亡时间d_k，若没有同时死亡患者，则q_{k1}和q_{k2}中一个为 1，另一个为 0。注意到$r_{k1} + r_{k2} = r_k, q_{k1} + q_{k2} = q_k$。在每个死亡时间点，我们作形如下表的$2 \times 2$表格：

	Group 1	Group 2	Total
Died	q_{1k}	q_{2k}	q_k
Survived	$r_{1k} - q_{1k}$	$r_{2k} - q_{2k}$	$r_k - q_k$
Total	r_{1k}	r_{2k}	r_k

对数秩检验的核心思路如下：为了检验假设$H_0 : E(X) = \mu$，我们对随机变量X构造一个检验统计量：

$$W = \frac{X - \mu}{\mathrm{Var}(X)}$$

构造对数秩统计量我们则计算$X = \sum_{k=1}^{K} q_{k1}$，其中$q_{k1}$通过查上面的表得到。若两组数据中没有显著差异，则下公式成立：

$$\mu_k = \frac{r_{k1}}{r_k} q_k$$

因此$X = \sum_{k=1}^{K} q_{k1}$的期望$\mu = \sum_{k=1}^{K} \frac{r_{k1}}{r_k} q_k$。进一步地$q_{k1}$的方差：

$$\mathrm{Var}(q_{k1}) = \frac{q_k (r_{k1}/r_k)(1 - r_{k1}/r_k)(r_k - q_k)}{r_k - 1}$$

尽管q_{ki}可能相关，我们依然估计：

$$\mathrm{Var}(\sum_{k=1}^{K} q_{k1}) \approx \sum_{k=1}^{K} \mathrm{Var}(q_{k1}) = \sum_{k=1}^{K} \frac{q_k (r_{k1}/r_k)(1 - r_{k1}/r_k)(r_k - q_k)}{r_k - 1}$$

这一步我们得到了检验统计量中的$\mathrm{Var}(X)$，因此最终：

$$W = \frac{\sum_{k=1}^{K} (q_{k1} - \mu_k)}{\sqrt{\sum_{k=1}^{K} \mathrm{Var}\left(q_{k1}\right)}} = \frac{\sum_{k=1}^{K} \left(q_{k1} - \frac{q_k}{r_k} r_{k1}\right)}{\sqrt{\sum_{k=1}^{K} \frac{q_k (r_{k1}/r_k)(1 - r_{k1}/r_k)(r_k - q_k)}{r_k - 1}}}.$$

当样本量较大时，对数秩检验得到的统计量W大致呈现标准正态分布，因此计算零假设的p值可以比较两组生存曲线之间的差异

11.5 回归模型与生存数据

考虑回归模型在生存数据中的应用。每对观测值(Y, δ)，其中$Y = \min(T, C)$，指示变量δ在$T \leq C$时为 1，否则为 0；特征向量$X \in R^p$，表示有p个特征的向量，预测目标为真实生存时间T

我们的目的是预测T而不是Y，因此需要模仿之前的操作使用顺序结构

11.5.1 风险函数

风险函数 (hazard function)，也称为**及时风险率 (hazard rate)** 定义为：

$$h(t) = \lim_{\Delta t \to 0} \frac{\Pr(t < T \leq T + \Delta t | T > t)}{\Delta t}$$

它衡量了在时间t时，个体未发生事件的条件下，瞬间发生事件的速率。将生存数据建模为协变量函数的关键方法在于风险函数

由条件概率公式得：

$$
\begin{aligned}
h(t) &= \lim_{\Delta t \to 0} \frac{\Pr\left((t < T \leq t + \Delta t) \cap (T > t)\right)/\Delta t}{\Pr(T > t)} \\
&= \lim_{\Delta t \to 0} \frac{\Pr(t < T \leq t + \Delta t)/\Delta t}{\Pr(T > t)} \\
&= \frac{f(t)}{S(t)},
\end{aligned}
$$

其中：

$$
f(t) = \lim_{\Delta t \to 0} \frac{\Pr(t < T \leq t + \Delta t)}{\Delta t}
$$

表示T的**概率密度函数 (probability density function)**，也即死亡时间T的瞬时变化率，第一步消去交集的过程是显然的

> 上式的三个等号实际上是表述T分布的等效方式

第i对观测值的似然函数为：

$$
\begin{aligned}
L_i &= \begin{cases} f(y_i) & \text{if the } i\text{th observation is not censored} \\ S(y_i) & \text{if the } i\text{th observation is censored} \end{cases} \\
&= f(y_i)^{\delta_i} S(y_i)^{1-\delta_i}.
\end{aligned}
$$

其中y_i表示某个时间，$f(y_i)$表示概率密度函数，$S(y_i)$表示生存函数，δ_i是当前时间的删失指示变量，值为 1 时表示事件发生

这个公式的直觉是：若$Y = y_i$时，第i个观测值未被删失，则似然函数应为事件刚好发生在y_i的概率，即密度函数$f(y_i)$；如果第i个观测值被删失，则说明其至少存活到y_i，则似然函数应为个体至少活到y_i的概率，即生存函数$S(y_i)$

假设n对观测值独立，极大似然函数的形式则是：

$$
L = \prod_{i=1}^{n} f(y_i)^{\delta_i} S(y_i)^{1-\delta_i} = \prod_{i=1}^{n} h(y_i)^{\delta_i} S(y_i),
$$

得到似然函数后，我们就可以对原函数进行估计了。可采用两种形式：

1. 假设生存函数是形如$f(t) = \lambda e^{-\lambda t}$的指数形式，或者来自$\Gamma$或者 Weibull 分布族
2. 用非参数化的估计，即类似于 Kaplan-Meier 估计器的阶梯形式

使用风险函数来估计生存函数比直接用概率密度函数好，这是因为前者反应了生存函数的变化率，更能展示协变量与生存函数的关系。我们可以假设一个指数型的风险函数：

$$
h(t|x_i) = e^{\beta_0 + \sum_{j=1}^{p} \beta_j x_{ij}}
$$

其中指数函数保证风险函数始终为正。上式的$\beta_0, \beta_1, \ldots, \beta_p$可用极大似然法估计，然而指数函数形式的风险函数在每时每刻的值恒定，不符合风险逐渐增大的实际 (即生存函数的导数的导数为 0，实际应为正数)

11.5.2 比例风险

比例风险假设 (proportional hazards assumption) 指出：

$$h(t|x_i) = h_0(t)\exp\left(\sum_{j=1}^{p} x_{ij}\beta_j\right)$$

其中$h_0(t) \geq 0$是一个未知函数，也即**基础风险 (baseline hazard)**。基础风险衡量了当$x_{i1} = \cdots = x_{ip} = 0$时的风险。“比例风险”的名字来源于$h_0(t)$对每一类特征向量$X_i = (x_{i1}, x_{i2}, \ldots, x_{ip})$的图像成比例 (平移且不交叉)。项$\exp(\sum_{j=1}^{p} x_{ij}\beta_j)$被称为关于特征$X_i = (x_{i1}, x_{i2}, \ldots, x_{ip})$的**相对风险 (relative risk)**

未指明的$h_0(t)$实际上提供了任意满足数据的函数，这使得风险函数的形式相当灵活，这里关键假设是协变量x_{ij}是以指数形式影响风险率$\exp(\sum_{j=1}^{p} x_{ij}\beta_j)$的

> 比例风险假设可能在实际上并不容易成立，因为它意味着不同组的风险比例恒定，生存曲线永不交叉。然而事实上，随着时间的变化，风险比可能变动，生存曲线可能交叉

Cox 比例风险模型 (Cox's proportional hazards model) 提供了一种在无法使用极大似然法估计不确定$h_0(t)$条件下获得参数$\beta = (\beta_1, \ldots, \beta_p)^T$的估计，我们同样使用时间顺序的方法来实现这种估计。首先，假设每个个体的死亡发生在各不相同的时间，且$\delta_i = 1$，因此y_i是真正的死亡时间。则对于y_i时刻的第i对观测值其风险函数值为$h(y_i|x_i) = h_0\exp(\sum_{j=1}^{p} x_{ij}\beta_j)$，$y_i$时的总体风险 (处于风险集中，即为发生死亡的其他个体的风险) 为：

$$\sum_{i':y_{i'}\geq y_i} h_0(y_i)\exp\left(\sum_{j=1}^{p} x_{i'j}\beta_j\right)$$

则第i对观测值相对于总体的风险函数值为：

$$\frac{h_0(y_i)\exp\left(\sum_{j=1}^{p} x_{ij}\beta_j\right)}{\sum_{i':y_{i'}\geq y_i} h_0(y_i)\exp\left(\sum_{j=1}^{p} x_{i'j}\beta_j\right)} = \frac{\exp\left(\sum_{j=1}^{p} x_{ij}\beta_j\right)}{\sum_{i':y_{i'}\geq y_i}\exp\left(\sum_{j=1}^{p} x_{i'j}\beta_j\right)}$$

因为 Cox 模型是成比例的，第i个个体的失败概率通过上式给出 (表示在所有仍然存活的个体中，第i个成为下一个死亡者的概率)。在这里，基础风险函数$h_0(y_i)$被抵消了

偏极大似然 (partial likelihood) 函数即是这些相对于所有个体发生风险的比例的乘积：

$$PL(\beta) = \prod_{i:\delta_i=1}\frac{\exp\left(\sum_{j=1}^{p} x_{ij}\beta_j\right)}{\sum_{i':y_{i'}\geq y_i}\exp\left(\sum_{j=1}^{p} x_{i'j}\beta_j\right)}$$

偏似然函数忽略了$h_0(t)$的具体形式，但可以证明其提供了对β的良好估计。此外，一些和回归模型有关的其他信息，如零假设$H_0 : \beta_j = 0$也可以被验证并显示

> 偏似然函数一般是对不好直接使用极大似然估计的模型使用的，为了最大化偏极大似然函数 (没有封闭解)，我们需要用到第四章中的迭代算法

当协变量只有一个，且取值为二元时，我们使用对数秩检验和直接用 Cox 模型检测β等价。一般来说，如果单纯比较两组生存时间的差异，使用对数秩检验更便捷；若需要量化影响，则使

用 Cox 模型可以提供调整协变量的估计

一些关于 Cox 模型的细节如下：

- Cox 比例风险模型的指数项中没有β_0，这是因为它放入了$h_0(t)$中
- 假设死亡时间是唯一的，即无同时死亡；若有，则偏似然函数形式发生改变，计算更复杂
- 偏极大似然不是极大似然，是对极大似然的近似
- 通常，我们只关注β的估计，对$h_0(t)$估计的数学过程超出本书范围

11.5.3 Brain Cancer 数据集研究实例

本节用比例风险模型拟合 `BrainCancr` 数据集。结果显示，男性比女性在任意时间发生癌症的风险高出$e^{0.18}=1.2$倍，但是p值为0.61，表明结果并不显著

医学中的一种指数 (Karnofsky 指数) 在模型中的系数β_i为-0.05，$p=0.0027$，表明越高的指数显示越低的患病风险，结果显著

11.5.4 Publication 数据集研究实例

本节拟合 `Publication` 数据集，目的是研究论文发表时间与临床医学上的各种协变量的关系。首先用 Kaplan-Meier 生存曲线拟合了关于阴性实验结果和阳性实验结果的论文发表时间，并用对数秩检验检验其显著性，对数秩检验表明无显著差异

接着，考虑 Cox 风险比例模型，对比阴性实验结果和阳性实验结果论文发表时间的差异，此时检验认为差异显著。出现两种不同结果的原因是只有后者考虑了全部协变量影响 (调整其他协变量一致)

11.6 Cox 模型的收缩方法

受到损失函数+惩罚项的启发，我们可对 Cox 模型的偏极大似然函数使用收缩方法：

$$-\log\left(\prod_{i:\delta_i=1}\frac{\exp\left(\sum_{j=1}^p x_{ij}\beta_j\right)}{\sum_{i':y_{i'}\geq y_i}\exp\left(\sum_{j=1}^p x_{i'j}\beta_j\right)}\right)+\lambda P(\beta)$$

此处我们还需要考虑$P(\beta)=\sum_{j=1}^p\beta_j^2$对损失函数的影响，当然也可改为$P(\beta)=\sum_{j=1}^p|\beta_j|$惩罚项，二者分别对于岭回归和 lasso 回归

将 lasso 惩罚项应用到 Cox 模型对 `Publication` 数据集的拟合中。随着正则化强度的加强，模型在验证集中的偏似然误差 (partial likelihood deviance) 呈现 U 型曲线，其值是负对数部分似然的两倍。这显示了模型复杂度和误差之间的平衡关系

在测试集中，应用 Cox 模型存在两个问题：

1. 删失数据影响我们对其真实生存曲线的估计
2. Cox 模型不是用特征X来估计某个个体的生存时间T，而是用X对它的生存曲线$S(t|X)$进行建模，是时间t的函数

因此，为了评估模型，我们对每个测试观测值定义其风险评分：

$$\text{risk}_i = \text{budget}_i \cdot \hat{\beta}_{\text{budget}} + \text{impact}_i \cdot \hat{\beta}_{\text{impact}}$$

其中$\hat{\beta}_{\text{budget}}$和$\hat{\beta}_{\text{impact}}$是训练集中估计出来的两个特征参数，我们用这个两个值的特征组合来表示“风险”，例如高风险组中包含这对值最大的观测值。在 `publication` 中数据集中，观测结果被分为高中低三层，这三层之间的生存曲线存在明显区分，排序正确

11.7 其他主题

11.7.1 生存曲线的曲边形面积

类比二分类问题使用的 ROC 曲线，得到 AUC (曲线下面积) 来获得分类器效果比较的方法，我们试图分析生存曲线的曲线下面积的意义

考虑设计每对观测值的风险评分$\hat{\eta}_i = \hat{\beta}_1 x_{i1} + \cdots + \hat{\beta}_p x_{ip}$，我们可能会尝试判断风险评分和实际生存时间$t_i$都符合预测的观测对有多少来评价模型性能 (以风险评分较大，且实际生存时间较短为判断正例)。然而删失数据将会导致我们无法获得实际的生存时间$t_{i'}$

因此，需要用**Harrell's 一致性系数 (Harrell's concordance index)**，也被称为**C 系数 (C-index)** 来计算满足$\hat{\eta}_{i'} > \hat{\eta}_i$且$\hat{y}_i > y_{i'}$的比例：

$$C = \frac{\sum_{i,i':y_i>y_{i'}} I(\hat{\eta}_{i'} > \hat{\eta}_i)\delta_{i'}}{\sum_{i,i':y_i>y_{i'}} \delta_{i'}}$$

其中指示变量$I(\hat{\eta}_{i'} > \hat{\eta}_i)$在$\hat{\eta}_{i'} > \hat{\eta}_i$满足时为 1。这个指数的分子表示所有满足实际生存时间较长的个体中，模型也能正确排序的观测对数量；分母表示所有满足实际生存时间较长的个体中，实际没有被删失的观测对数量

> 当C越接近 1 时，表明模型效果越好；若$C = 0.5$，则模型等同于随机猜测

11.7.2 时间刻度的选择

时间刻度的选择至关重要。例如，考察协变量对患者治疗效果的影响，我们可以选择两种时间零作为开始时间：

1. 以患者的出生日期作为时间零点。此时$h_0(t)$隐含量患者的年龄，即年龄被时间刻度吸收，无需额外调整年龄
2. 以治疗日期作为时间零点。此时年龄是独立于时间变量的协变量，需要额外加入模型

不同的选择取决于研究背景，包括我们是否要单独研究年龄的影响

11.7.3 随时间变化的协变量

比例风险模型的强大之处在于它能处理随时间变化的协变量。例如，测量血压时，我们不在将血压值视为静态特征x_i，而是视为一个随时间变化的动态特征$x_i(t)$

因为偏似然本身就是随着时间序列构造的，因此它处理随时间变化的协变量更加直接。我们只需要将计算$PL(\beta)$式子中的$x_{ij}, x_{i'j}$改成$x_{ij}(y_i), x_{i'j}(y_i)$。一个实例是比较心脏移植患者能否获得更长的生存期，若使用固定的协变量表示移植状态，则会忽略患者必须活得足够长才能接受心脏移植的事实。因此，需要用随时间变化的协变量来建模移植状态，即患者在t前接受移植，则$x_i(t) = 1$，否则$x_i(t) = 0$

11.7.4 比例风险假设的检验

Cox 模型依赖于比例风险假设。在定性特征情况下，我们可以绘制每个级别的对数风险函数，若假设成立，则这些函数应该只差一个常数；对于定量的特征，我们可以用特征分层方法来实现验证

> Cox 模型对违反此假设的数据依然具有相当的稳健性

11.7.5 生存树

决策树的思想可以用于生存数据分析，同理，也可以用随机森林的方式来优化

第十二章无监督学习

监督学习和无监督学习的特点：

- 监督学习通常使用一组特征$X_1, X_2, \ldots, X_P$来预测目标变量Y
- 无监督学习没有目标变量Y，我们的目的是找到$X_1, X_2, \ldots, X_p$之间的一些潜在关系

无监督学习尝试可视化，发现特征$X_1, X_2, \ldots, X_p$中的子组。本章讨论主成分分析和聚类两种无监督学习模式

12.1 无监督学习的挑战

无监督学习相比于监督学习面临更多挑战：

- 目标变得主观，因为我们没有具体的预测变量Y
- 无法检测我们的结果，这与监督学习可以用交叉检验或者测试集不同

无监督学习通常作为**探索性数据分析 (exploratory data analysis)** 的一部分。它的应用包括：

1. 在若干名癌症患者中找到一些子组，以便更好地对这种疾病进行分类
2. 根据用户的购物习惯，为他推荐喜欢的商品
3. 将相似浏览记录的人识别出来，进行搜索推广

12.2 主成分分析

我们之前讨论过主成分回归。现在我们讨论的**主成分分析 (Principal components analysis)** 是一种无监督学习方法，它从一组特征$X_1, X_2, \ldots, X_p$中找到衍生的特征作为主成分。它的应

用包括：

- 进行数据可视化
- 数据插值，为有空缺的矩阵填充值
- 探寻潜在的特征

12.2.1 什么是主成分

对于一组特征$X_1, X_2, \ldots, X_p$，我们可能希望查看它们的数据分布。两两特征之间的散点图可以实现这一点，但是当p过多时将会很不方便。现在，我们希望用更少的特征尽可能多地反应原来的信息，以可视化这些数据

PCA 找到包含尽可能多的数据信息的低维表示，它通过观测值沿每个维度的变化量来衡量更低的有意义的维度。所有的降维都是原来p个特征的线性组合。一组特征的**第一主成分 (first principal component)** 是标准化的 (normalized)$X_1, X_2, \ldots, X_p$的线性组合：

$$X_1 = \Phi_{11}X_1 + \Phi_{21}X_2 + \cdots + \Phi_{p1}X_p$$

这个线性组合拥有最大的方差。此处标准化是指$\sum_{i=1}^{p} \Phi_{j1}^2 = 1$，我们称$\Phi_{i1}, i = 1, 2, \ldots, p$为主成分的**载荷 (loading)**，载荷组成了组成分的载荷向量$\Phi_1 = (\Phi_{11}, \Phi_{21}, \ldots \Phi_{p1})^T$。对载荷平方和的约束限制了任意的方差变化

由于我们只关心方差，我们假设设计矩阵X的每一列均值为 0，然后对于每个样本：

$$z_{i1} = \Phi_{11}x_{i1} + \Phi_{21}x_{i2} + \cdots + \Phi_{p1}x_{ip}$$

我们寻找这样的一组载荷使它们的样本方差最大。换句话说，第一组成分向量是下面这个优化式子的解：

$$\max_{\phi_{11}, \ldots, \phi_{p1}} \left\{ \frac{1}{n} \sum_{i=1}^{n} \left(\sum_{j=1}^{p} \phi_{j1} x_{ij} \right)^2 \right\} \quad \text{subject to} \quad \sum_{j=1}^{p} \phi_{j1}^2 = 1$$

对于上式，由于$\frac{1}{n} \sum_{i=1}^{n} x_{ij} = 0$，$z_{11}, \ldots z_{n1}$的均值也是 0，且我们要优化的式子即是最大化$n$对样本方差$z_{i1}$，我们称$z_{11}, \ldots, z_{n1}$为第一主成分的**得分 (score)**。最大化上式可用**特征分解 (eigen decomposition)** 的方法解决，但超出本书范围

关于主成分的几何解释如下：载荷向量Φ_1的元素$\Phi_{11}, \Phi_{21}, \ldots, \Phi_{p1}$定义了特征空间中的一个方向，在此方向上的数据变化最大。如果我们将n对数据投影到这个方向，投影值$z_{11}, \ldots, z_{n1}$就是它们的主成分得分，在几何上即是在这条主成分轴Z_1上的坐标位置。载荷向量中的元素大小表示特征在投影方向上的权重大小

找到第一个主成分Z_1后，我们可以找到第二个主成分Z_2，它也是$X_1, X_2, \ldots, X_p$的线性组合，但是在所有与Z_1不相关的线性组合中，这个组合的方差最大。第二个主成分的分数形式如下：

$$z_{i2} = \phi_{12}x_{i1} + \phi_{22}x_{i2} + \cdots + \phi_{p2}x_{ip}$$

这里Φ_2是第二主成分的载荷向量。事实上，为了约束Z_2与Z_1不相关，Z_2必须是与Z_1正交也即垂直的。为了找到Φ_2，我们求解一个类似的优化问题，但是添加条件是Φ_2与Φ_1正交

计算完主成分后，我们可以将他们相互绘制，以获得低维度的可视化图。例如，绘制分数向量Z_1, Z_2, Z_3的两两组合，在几何上相当于将原始数据投影到Φ_1, Φ_2, Φ_3的子空间上并绘制投影点

12.2.2 主成分的另一种解释

上一节我们认为主成分就是原始数据变动最大的方向。下面提供另一种解释：主成分提供了一个能最接近观测值的低维表面

> 这里说是“表面”实际上是比原始维度低的图形的面。例如三维空间的低维表面是一个二维平面

第一个主成分的载荷向量的一个重要属性是：它是p维空间中最接近n个观测值的直线。将此概念推广，则前两个主成分是最接近n个观测值的平面；前三个主成分是三维超平面

使用这个解释，则前M个主成分得分向量Z和前M个主成分载荷向量Φ提供了一个最优的M维估计，它在欧氏距离上最优。对第i个观测值x_{ij}：

$$x_{ij} \approx \sum_{m=1}^{m} z_{im} \Phi jm$$

正式来看，给定原始数据矩阵$\mathbf{X}$，为了找到对它们的估计，使得$x_{ij} \approx \sum_{m=1}^{M} a_{im} b_{jm}$，我们最小化残差平方和：

$$\underset{\mathbf{A} \in \mathbb{R}^{n \times M}, \mathbf{B} \in \mathbb{R}^{p \times M}}{\text{minimize}} \left\{ \sum_{j=1}^{p} \sum_{i=1}^{n} \left(x_{ij} - \sum_{m=1}^{M} a_{im} b_{jm} \right)^2 \right\}$$

其中$\mathbf{A}$是一个$n \times M$的矩阵，位置(i, m)的元素就是a_{im}，$\mathbf{B}$是一个$p \times m$的矩阵，位置(j, M)的元素就是b_{jm}。可以证明对于任意的M，$\mathbf{A}$和$\mathbf{B}$实际上列分别对应主成分得分$Z_1, Z_2, \ldots, Z_M$和主成分载荷$\Phi_1, \Phi_2, \ldots, \Phi_M$。优化问题的最优解得到的估计矩阵就是最终答案。这意味这我们得到的矩阵要能最小化：

$$\sum_{i,j} \left(x_{ij} - \sum_{m=1}^{M} z_{im} \Phi_{jm}^2 \right)$$

总的来说，M个主成分得分和载荷向量在M足够大时能通过对原始数据的很好近似，当$M = \min(n-1, p)$时，$x_{ij} = \sum_{m=1}^{m} z_{im} \Phi_{jm}$

12.2.3 解释方程的比例

在对原始数据投影后，我们感兴趣的问题是对于每个组成分，它**解释的方差比例 (proportion of variance explained, PVE)** 由多少。对于已经标准化的数据，总体方差：

$$\sum_{j=1}^{p} \text{Var}(X_j) = \sum_{j=1}^{p} \frac{1}{n} \sum_{i=1}^{n} x_{ij}^2$$

第m个主成分解释的方差：

$$\frac{1}{n}\sum_{i=1}^{n} z_{im}^2 = \frac{1}{n}\sum_{i=1}^{n}\left(\sum_{j=1}^{p} \Phi_{jm} x_{ij}\right)^2$$

因此，PVE 的计算如下：

$$\frac{\sum_{i=1}^{n} z_{im}^2}{\sum_{j=1}^{p}\sum_{i=1}^{n} x_{ij}^2} = \frac{\sum_{i=1}^{n}\left(\sum_{j=1}^{p} \phi_{jm} x_{ij}\right)^2}{\sum_{j=1}^{p}\sum_{i=1}^{n} x_{ij}^2}.$$

每个主成分的 PVE 都是正数，计算累计 PVE 可对前M个主成分的 PVE 进行求和，它们的 PVE 之和为 1

总体方差可以分解为：

$$\underbrace{\sum_{j=1}^{p}\frac{1}{n}\sum_{i=1}^{n} x_{ij}^2}_{\text{Var. of data}} = \underbrace{\sum_{m=1}^{M}\frac{1}{n}\sum_{i=1}^{n} z_{im}^2}_{\text{Var. of first } M \text{ PCs}} + \underbrace{\frac{1}{n}\sum_{j=1}^{p}\sum_{i=1}^{n}\left(x_{ij} - \sum_{m=1}^{M} z_{im}\phi_{jm}\right)^2}_{\text{MSE of } M\text{-dimensional approximation}}$$

通过最大化前面M个主成分的方差，我们就能最小化M维近似的均方误差，反之亦然。

此外，我们通过上式可以发现 PVE 的定义等价于：

$$1 - \frac{\sum_{j=1}^{p}\sum_{i=1}^{n}\left(x_{ij} - \sum_{m=1}^{M} z_{im}\phi_{jm}\right)^2}{\sum_{j=1}^{p}\sum_{i=1}^{n} x_{ij}^2} = 1 - \frac{\text{RSS}}{\text{TSS}},$$

其中TSS表示X的总平方和，RSS表示M个主成分维度重构给出的残差平方和，则上式表示保留的方差比例。这与R^2的定义非常类似，我们可以将 PVE 解释为前M个主成分给出的R^2

通过绘制累积 PVE 的图像，我们得到了**碎石图 (scree plot)**，它能逐步反应各主成分解释的方差比例

12.2.4 关于 PCA 的更多信息

变量尺度问题：
我们之前执行 PCA 时都将变量标准化了。与线性回归不同，对变量乘以系数将会影响 PCA 的结果，量纲也会影响 PCA 结果，这是因为载荷向量需要赋予方差大的特征更高的系数。然而，对于同量纲的问题，我们不应该对其进行标准化

> 对于同量纲且方差差异有意义的数据，我们不需要标准化；但若它们数值范围差异过大，或者仅需要分析相关性结构，我们可以标准化
> 标准化会消除原始方差差异，仅衡量相关性

主成分的唯一性：
主成分的方向通常唯一 (除非矩阵的特征值相同，实际中很少出现)。载荷向量的符号正负不影响其方向性，因此我们分析时应当只考虑载荷的绝对值大小

决定主成分数量：
决定主成分数量是一个主观的过程，取决于我们希望探索多少感兴趣的因子。如果前面几个因子我们感兴趣，则我们很可能继续选择更多的因子来分析，反之我们不会继续关注后面的因子。这是 PCA 被称为探索性分析的原因

一种决定主成分数量的方法是观察碎石图，当图像出现拐弯时，后续的主成分解释的方差将相当小，我们不会选择；然而，这也是主观的过程

> 用于监督学习的 PCA 可以通过交叉检验等方式来决定主成分数量

12.2.5 主成分的其他用处

我们之前学过的主成分回归实际上用到了主成分分数向量作为特征来执行回归，这是因为$M << p$个主成分分数向量通常能减少噪声影响

12.3 缺失值和矩阵补全

数据集通常不是完整的，包含一些随机的缺失值，一些基础的处理方式包括：

- 删除包含缺失值的行。这可能导致一些浪费，尤其是损失比例很大时
- 若x_{ij}缺失，用第j列的均值替换它

本节介绍如何用主成分来**估算 (impute)** 缺失值，这种方案叫做**矩阵补全 (matrix completion)**。补全后的矩阵可以用于统计学，例如线性回归或者 LDA。估算缺失值基于**随机缺失 (missing at random)** 的数据集，如果数据的缺失本身提供信息 (如删失数据中的超过刻度等)，则不适用这个方法

事实上，数据会不可避免地丢失。例如考察n个客户对 Netfilx 的p部电影的评分，大多数用户不可能看过全部电影。如果我们能对矩阵进行补全，则可以支持我们的**推荐系统 (recommender system)**

用主成分补充缺失值
补全矩阵缺失值和获得主成分可以同时进行。考察下面的优化问题：

$$\min_{\mathbf{A}\in\mathbb{R}^{n\times M},\mathbf{B}\in\mathbb{R}^{p\times M}}\left\{\sum_{(i,j)\in\mathcal{O}}\left(x_{ij}-\sum_{m=1}^{M}a_{im}b_{jm}\right)^2\right\}$$

其中$\mathcal{O}$是是所有观察到的观测值(i,j)的集合，是原始矩阵$n\times p$对的一个子集。求解出估计的矩阵$\mathbf{A}$和$\mathbf{B}$后，我们有：

- 可以估计缺失值x_{ij}，即$\hat{x}_{ij}=\sum_{m=1}^{M}\hat{a}_{im}\hat{b}_{jm}$，其中$\hat{a}_{im},\hat{b}_{jm}$分别是$\mathbf{A},\mathbf{B}$的第$(i,m)$和第$(j,m)$个值
- 可以估算前M个主成分得分和载荷矩阵，即用估算的缺失值来获取

事实证明与完整的情况不同，求解上面的矩阵不适用特征分解的方法，但是下面的迭代算法提供了一个估算方案：

1. 首先创建一个$n\times p$的完整数据矩阵$\widetilde{X}$，它的元素(i,j)满足：

$$\tilde{x}_{ij}=\begin{cases}x_{ij} & \text{if } (i,j)\in\mathcal{O}\\ \bar{x}_j & \text{if } (i,j)\notin\mathcal{O},\end{cases}$$

   其中$\overline{x}_j$是第j列观测值的平均值。此处我们得到$\mathcal{O}$就是未缺失的观测值的集合。接着，当优化目标式子 (上式) 未满足时，执行：
2. 用补全的完整数据矩阵$\widetilde{X}$求解分解的矩阵$\mathbf{A}$和$\mathbf{B}$，即优化下面的式子：

$$\min_{\mathbf{A}\in\mathbb{R}^{n\times M},\mathbf{B}\in\mathbb{R}^{p\times M}}\left\{\sum_{j=1}^{p}\sum_{i=1}^{n}\left(\tilde{x}_{ij}-\sum_{m=1}^{M}a_{im}b_{jm}\right)^2\right\}$$

3. 对于不在$\mathcal{O}$的元素(i,j)，使$\widetilde{x}_{ij}\leftarrow\sum_{m=1}^{M}\hat{a}_{im}\hat{b}_{jm}$
4. 计算目标

$$\sum_{(i,j)\in\mathcal{O}}\left(x_{ij}-\sum_{m=1}^{M}\hat{a}_{im}\hat{b}_{jm}\right)^2$$

   执行循环完毕后，返回缺失值$\widetilde{x}_{ij},(i,j)\notin\mathcal{O}$

> 上面算法在每次循环都会减小优化目标，但是不能保证该算法能够实现全局最优

人为删去随机数据的实验表明，使用插补法得到真实值和插补值之间存在一定的相关性。虽然插补矩阵计算的模型结果不如真是矩阵，但是它依然提供了一个相当好的近似，考虑到实际中的数据大多存在缺失，插补法依然有其存在价值

> 此实验中的`USErrests`数据集特征$p=4$相对较少，可能影响估算算法的稳健性

在使用算法估算时，我们常常需要考虑合适的主成分数量M，这可以通过类似验证集的方法实现：我们筛选出一个验证集，人为删去其中的一些变量，然后挑选M来估计并评价不同的M的效果，最后再泛化到整个数据集中

推荐系统

流媒体播放网站的推荐系统依赖于各种算法。以 Netfilx 为例，为了推荐用户喜欢的电影，它要求用户对p部电影中看过的进行评分，产生了一个非常大的矩阵$n\times p$，一个早期的例子是$n=480,189$和$p=17770$，其中99%的元素为空。为了补全这些矩阵，Netflix 基于这样一个思想：第i位用户观看的电影集合将与其他用户观看的电影有重叠；此外，其他用户与第i位用户有相似偏好。因此可用第i位用户未看过电影的类似用户的评分来预测第i位用户是否喜欢这个电影

在数学上，这相当于用上面的算法估算$\hat{x}_{ij}=\sum_{m=1}^{M}\hat{a}_{im}\hat{b}_{jm}$，且$M$主成分的具体意义是某个团体分类：

- $\hat{a}_{im}$表示第i个用户属于第m个团体的强度 (喜欢某类电影用户的集合)
- $\hat{b}_{jm}$表示第j个电影属于第m类别的强度

类似于上面算法的主成分模型是许多推荐系统的核心

12.4 聚类方法

聚类 (clustering) 是一类包含广阔的算法，它用于寻找数据集的子组或**簇 (clusters)**。聚类分割各种簇，使得相似的数据靠的越近，不同的数据离得越远。如何衡量数据的相似与不同取决于我们分析的数据集的背景知识，例如对于n名癌症病人的基因表达数据，我们希望找到癌症的部分亚群

> 市场细分需要用到聚类方法。例如我们希望识别哪些人群容易接受哪种类型的广告

PCA 与聚类的区别如下：

- PCA 寻找原始观测值的低维表示，以尽可能地解释数据的方差
- 聚类在观测值中寻找同质的子组

本章介绍两种最著名的聚类算法：**K-means 聚类 (K-means clustering)** 和 **层次聚类 (hierarchical clustering)**。前者在给定聚类数目的情况下聚类，后者则是将从 1 到n的所有可能聚类以**聚类树图 (dendrogram)** 的形式展现。二者各有优劣

一般而言，我们可以根据特征对观测进行聚类，以识别观测之间的子群，也可以根据观测对特征进行聚类，以发现特征之间的子群。在接下来的内容中，为了简单起见，我们将根据特征讨论聚类观测

> 转置矩阵即可改变聚类模式

12.4.1 K-means 聚类

K-means 聚类在指定聚类数目K之后，将观测数据完全分配到这些簇中。这个过程非常直观。首先定义一些记号：

1. $C_1 \cup C_2 \cup \cdots \cup C_K = \{1, \ldots, n\}$ 。即所有观测值都在K个簇中
2. $C_k \cap C_{k'} = \emptyset, \forall k \neq k'$。即所有簇互斥，没有属于两个簇的元素

K-means 聚类的核心思想是类内差异尽可能小的聚类是好的聚类。记第i个观测值属于第k个簇为$i \in C_k$，则 K-means 实际上是一个优化问题：

$$\underset{C_1,\ldots,C_k}{\text{minimize}} \left\{ \sum_{k=1}^{K} W(C_k) \right\}$$

其中$W(C_k)$衡量了类内差异。类内差异有许多定义方式，一种最常见的是**欧式平方距离 (squared Euclidean distance)**：

$$W(C_k) = \frac{1}{|C_k|} \sum_{i,i' \in C_k} \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2$$

其中$|C_k|$表示第k个簇内观测值数量。上式即类内差异表示为第k个簇钟所有观测值成对之间的欧氏距离平方和处于观测数。由上面两个式子得到：

$$\min_{C_1,\ldots,C_K}\left\{\sum_{k=1}^{K}\frac{1}{|C_k|}\sum_{i,i'\in C_k}\sum_{j=1}^{p}(x_{ij}-x_{i'j})^2\right\}$$

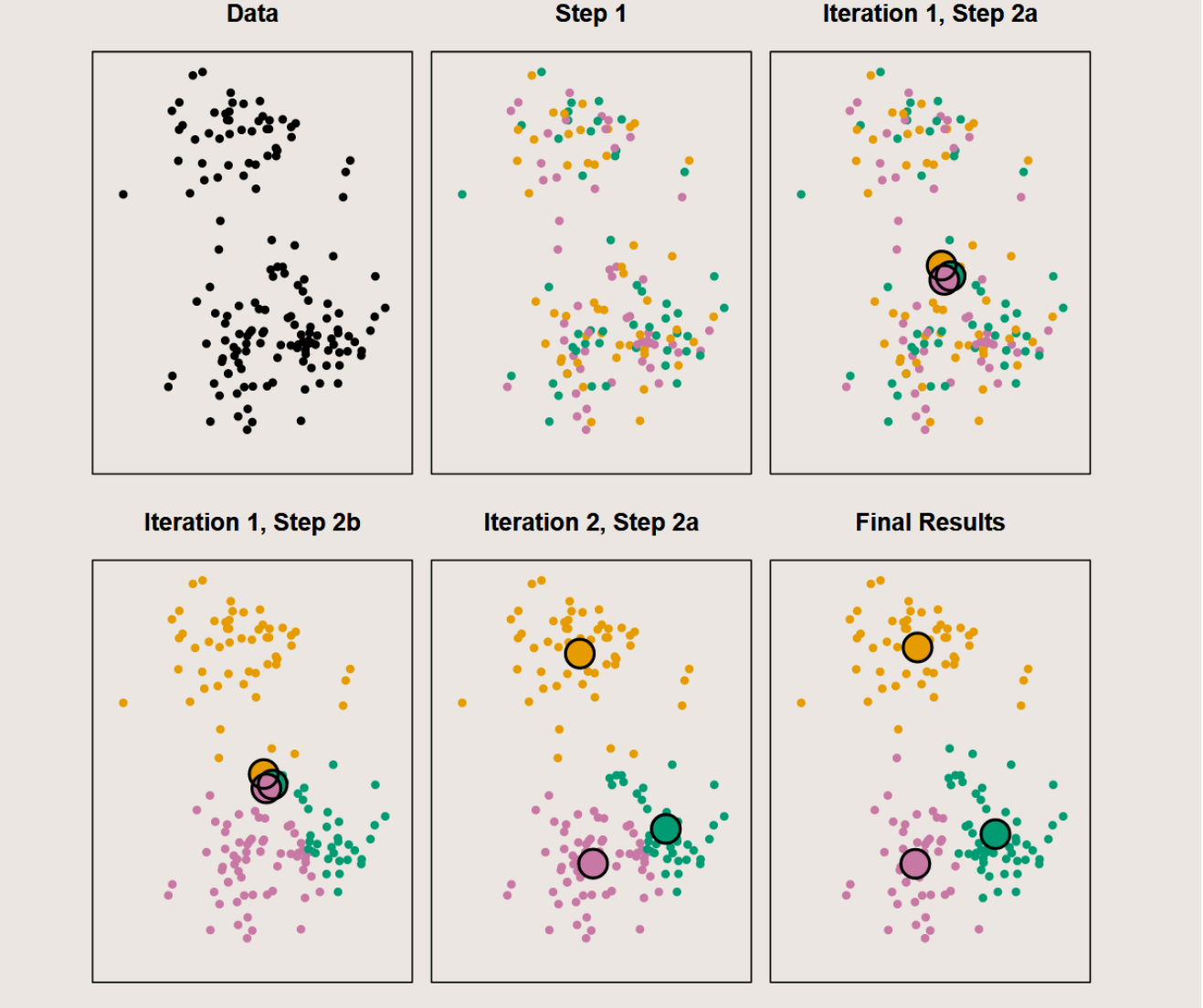
这个优化目标的精确求解很不容易，因为有K^n种可能的解法。下面这个算法可以给出局部最优解：

1. 对∀观测值，分配一个从1到k的数字作为初始聚类分配
2. 不断迭代，直到停止分配簇
   2.1 对于K个聚类中的每个簇，计算簇的**质心 (centroid)**。第k个簇的质心是第k个簇所有观测值特征均值的向量
   2.2 将每个观测值分配给距离质心最近的簇，其距离用欧氏距离衡量

下面的式子有助于我们理解上面算法能优化目标的原因：

$$\frac{1}{|C_k|}\sum_{i,i'\in C_k}\sum_{j=1}^{p}(x_{ij}-x_{i'j})^2=2\sum_{i\in C_k}\sum_{j=1}^{p}(x_{ij}-\bar{x}_{kj})^2$$

其中$\bar{x}_{kj}=\frac{1}{|C_k|}\sum_{i\in C_k}x_{ij}$表示特征$j$在簇$C_k$中的均值。等式右边实际上是簇内观测值到均值的平方距离和的两倍，即我们只需要优化簇内观测值到其质心的距离。当结果不再发生变化时，达到最优



K-means 聚类的最终结果依赖于初始的簇分类质心，因此需要多运行几次取最优的结果。这是因为 K-means 寻找的是局部最优解

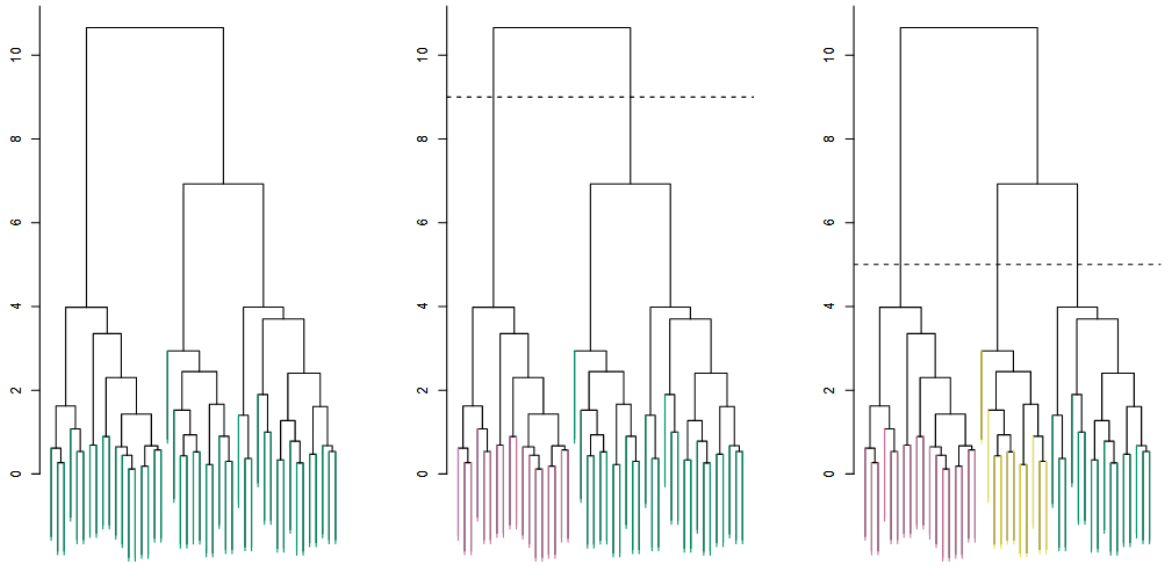
> 簇的数量选取很重要，将在后面讲解

12.4.2 层次聚类

K-means 聚类的一个劣势是需要指定K的大小，因此层次聚类提供了一个更吸引人的方案，而且它还能生成直观的聚类树图。本节介绍**自下而上 (bottom-up)** 和 **凝聚式 (agglomerative)** 两种聚类方案

解释聚类树图
聚类树图是一颗倒着的树 (就像决策树)，每个叶子节点表示观测值之一，它们向树根逐渐合并成枝干。不难看出，越早融合的观测值相似程度越高，接近树根融合的枝干则可能完全不相同

值得注意的是，我们不能仅仅根据叶子节点的相近程度来判断它们的相似性，而应该结合它们位于的层次 (即从下往上融合的时机)。通过对树状图水平切割，我们能得到对应的聚类；不同的切割高度对应不同的聚类，相当于 K-means 中的K



> n个观测值可以产生2^{n-1}中不同的排列顺序，因此不能仅仅根据距离判断相似度

肉眼观察树状图可以直接让我们选择需要划分的聚类数量，然而，有时候这种划分并不容易抉择。层次聚类看上去是“嵌套”的，这也就是说更高处的切割将会包含更低处的切割 (子集的关系)；但是对于没有嵌套关系的数据集，这种划分往往不那么合适，这时候层次聚类的效果不如 K-means

> 合适的划分：切割成两个簇划分为“动物”和“植物“，切割为三个簇在植物中细分出”花草“和”树木“
> 不合适的划分：不同国籍人群的数据集，切割成两个簇划分为“男人”和“女人”，切割成三个簇则在女人中划分出三个国家，无层次嵌套关系

层次聚类算法
层次聚类的算法如下：

1. 从n个观测值和某个不相似度测量值 (例如欧氏距离) 开始，将观测值划分为C_n^2个不想似对，每个观测值为一个聚类
2. 从$i = n, n-1, \ldots, 2$:
   2.1 检查所有不相似对的不相似度，选择不相似度最小的 (即最相似的) 进行合并
   2.2 对于新的对，重新计算不相似度

我们的不相似度仅仅度量了两个观测值之间的不相似情况，现在拓展到度量两组之间，这里引入了一个新的概念**连接 (linkage)**。连接包括完全 (complete)、平均（average)、单个 (single) 和质心 (centroid) 四种，前两者往往优于单个类型的连接，因为能产生更均平衡的树状图；质心连接常用在几何中，它可能产生一种称为**反转 (inversion)** 的现象，即后合并的节点高度可能低于先合并的，这是因为质心 (几何中心) 的计算是动态的

> 最终得到的树状图取决于连接方式

链接（Linkage）	描述（Description）
完全（Complete）	最大的聚类间不相似度。计算聚类 A 中的观测值与聚类 B 中的观测值之间的所有成对不相似度，并记录这些不相似度中的最大值。
单一（Single）	最小的聚类间不相似度。计算聚类 A 中的观测值与聚类 B 中的观测值之间的所有成对不相似度，并记录这些不相似度中的最小值。单一链接可能导致延伸的、拖尾的聚类，其中单个观测值会逐一融合。
平均（Average）	平均的聚类间不相似度。计算聚类 A 中的观测值与聚类 B 中的观测值之间的所有成对不相似度，并记录这些不相似度的平均值。
质心（Centroid）	聚类 A 的质心（长度为 p 的均值向量）与聚类 B 的质心之间的不相似度。质心链接可能导致不良的反转（inversions）。

不相似度度量的选择
通常，我们的不相似度度量量是欧氏距离。有时候，我们又会使用相关性来度量特征的相似性，即如果两个观测值的特征高度相关，则它们被认为相似，即时它们在空间上相隔很远

> 基于相关性的度量将注重观测值的轮廓 (profile) 和变化趋势而非绝对距离

选择欧氏距离或者相关性取决于我们研究的数据。对于购买商品的用户，如果用欧氏距离我们可能将购买数量很少 (即很少上网) 的用户聚类到一起，而不是他们对产品的偏好

用相关性度量相似性时，应当对数据标准化。这种标准化也适用于 K-means。若不标准化，则可能因为数值大小 (量纲大小) 错误聚类

12.4.3 聚类中的实际问题

小决定和大后果
执行聚类前我们需要的决定：

- 是否标准化数据

- 对于层次聚类
  - 不相似性的度量
  - 连接的选取
  - 剪切聚类树图的位置
- 对于 K-means 聚类，选择合适的K

实践中，常常进行多种不同选择，选取最容易解决或者可解释的方案

> 聚类是主观性很强的

验证获得的簇
聚类后，我们获得若干数量的簇。然而，这些簇的划分是基于实际情况还是受到噪声影响的，我们无法知道。有一些用p值来决定是否接受这些簇的方法，但是本书不涉及，可参阅*The Elements of Statistical Learning, ESL*

其他在聚类中应当考虑的问题
不论是 K-means 还是层次聚类都会将n个观测值分配到所有可能的簇中。然而，这里面可能有些噪声，它们并不属于任何簇 (甚至属于多个簇)，但是强制被分类到各个簇中了。**混合模型 (mixture model)** 是一种软分类方案，它允许一个观测值被分配到多个簇中，参见 ESL

此外，聚类方法对数据的扰动很敏感。例如，我们对n个数据进行聚类，若删除一些观测值再聚类，很可能得到不同的聚类模型

对聚类结果的解读采取审慎的态度
前文提到聚类的一些小决定可能导致差异很大的结果，聚类的稳健性不高。因此，有必要使用多种参数聚类以及对数据集的子集进行聚类，查看哪些模式反复出现。最后，谨慎地分析结果，它们容易被误解或者过度解释

第十三章多重检验

本章介绍经典的 p 值假设检验。假设检验通常设置零假设，并试图证伪零假设以获得我们想要证明的目的 (就像数学中的反证法)。在现代统计学中，我们可能希望同时检验大量的零假设，并且小心地避免错误地拒绝过多的零假设

13.1 快速回顾假设检验

假设检验提供了一个严格的框架以回答统计学中“是或否”的问题

13.1.1 检验一个假设

实施假设检验共有 4 步：

1. 定义零假设和备择假设
2. 构建检验统计量
3. 计算 p 值
4. 基于 p 值，决定是否拒绝零假设

第一步, 定义零假设和备择假设
假设检验将可能的情况分为两类，即**原假设 (null hypothesis)** 和 **备择假设 (alternative hypothesis)**。原假设记为 H_0，它一般是我们希望证伪的事实陈述，虽然它很可能是真的；备择假设记为 H_a，代表我们希望证明的事实，通常与原假设相反。一个经典的零假设可以是：

There is no difference between the expected blood pressure of mice in the control and treatment grou

H_0 与 H_a 的处理并不对称。若我们的数据支持否定 H_0，则支持 H_a，且认为 H_0 不成立；若不支持拒绝 H_0，我们就不清楚是否是样本量太小还是 H_0 确实成立

第二步，构造检验统计量
检验统计量 (test statistic) 提供了反对原假设的证据，符号为 T，表示我们的数据与 H_0 的一致程度。以上面的小鼠实验为例，记 $x_1^t, \ldots, x_{n_t}^t$ 表示 n_t 个实验组的样本，$x_1^c, \ldots, x_{n_c}^c$ 表示 n_c 个对照组的样本，且 $\mu_t = E(X^t), \mu_c = E(X^c)$ 分别表示两组均值，为了检验原假设 $H_0 : \mu_t = \mu_c$，我们需要构造**两个样本下的 t 统计量 (two-sample t-statistic)**：

$$T = \frac{\hat{\mu}_t - \hat{\mu}_c}{s\sqrt{\frac{1}{n_t} + \frac{1}{n_c}}}$$

其中：

$$\hat{\mu}_t = \frac{1}{n_t}\sum_{i=1}^{n_t} x_i^t, \quad \hat{\mu}_c = \frac{1}{n_c}\sum_{i=1}^{n_c} x_i^c$$

且：

$$s = \sqrt{\frac{(n_t - 1)s_t^2 + (n_c - 1)s_c^2}{n_t + n_c - 2}}$$

它表示两个样本合并标准差的估计。这里 s_t^2 和 s_c^2 是两样本方差的无偏估计。一个较大的 T 值提供了更强的拒绝 H_0 的证据

第三步，计算 p 值
在上一步中，我们发现双样本 t 统计量的绝对数值反映了拒绝 H_0 的强度。那么，多大才算大？**p 值 (p-value)** 定义为 H_0 为真的情况下，观察到等于或更极端的检验统计量的概率，因此较小的 p 值提供了反对 H_0 的证据

具体来说，我们设上一步的统计量值为 $T = 2.33$，在 H_0 成立条件下，T 服从标准正态分布，则观察到不小于 T 的统计量的概率为 $1 - 0.98 = 2\%$。因此，我们得到的 p 值就是 0.02

在 H_0 假设下的统计量分布就是**零分布 (null distribution)**，常用的检验统计量在满足足够多样本和其他一些假设下，都是遵循已知分布的 (如正态分布、χ^2 分布等)

p 值可能是统计学中最常用和滥用的概念。请注意，p 值不是 H_0 成立的概率，其唯一解释是：重复多次实验，预期看到如此极端检验统计量的比例。它衡量的是数据在 H_0 条件下成立的可能性，而不是 H_0 本身成立的可能性

第四步，决定是否拒绝原假设
一旦计算出 p 值，我们就可以决定是否拒绝原假设。事实上，拒绝原假设的阈值取决于观测者，因为 p 值实际上反映的是若 H_0 成立，我们看到如此小的 p 值的概率。若设定阈值不足以支持我们拒绝原假设，我们只好认为原假设成立

13.1.2 两类错误

若原假设真实成立，则它是一个**真原假设 (true null hypothesis)**；否则称为**假原假设 (false null hypothesis)**。我们无法先验地知道原假设是否成立，因此需要假设检验

执行假设检验后，我们的行为如下表所示：

Decision	**Truth** (H_0)	**Truth** (H_a)
Reject (H_0)	**Type I Error**	Correct
Do Not Reject (H_0)	Correct	**Type II Error**

显然，当我们在 H_0 为假时拒绝 H_0，或在 H_0 为真时不拒接 H_0，则我们得到了正确结果；然而，若我们在 H_0 正确时错误地拒绝了 H_0，则我们犯了**第一类错误 (Type I error)**。**第一类错误率 (Type I error rate)** 定义为当 H_0 成立时，错误拒绝 H_0 的概率。类似地，若我们在 H_0 错误时接受了 H_0，则称为**第二类错误 (Type II error)**。假设检验的**功效 (power)** 定义为在 H_a 成立的条件下，正确拒绝 H_0 的概率

我们通常不能同时降低两类错误率。实践中，常认为第一类错误比第二类错误严重，因为前者通常涉及不正确的科学发现。所以我们希望较低的第一类错误率，如 $\alpha = 0.05$，同时假设检验的功效较大。可以证明，p 值和第一类错误率之间存在对应关系。通过在 p 值小于 α 时拒绝 H_0 就可以确保第一类错误率 $\leq \alpha$

13.2 多重检验面临的问题

当我们希望进行**多重检验 (multiple testing)**，我们很容易在大量的实验下错误地拒绝部分原假设，即犯第一类错误。例如，对于 m 个原假设 $H_{01}, \ldots, H_{0m}$，我们设定 $\alpha = 0.01$，则预期会拒绝 $\alpha \cdot m$ 个假设检验，当 m 很大时将会是大量的第一类错误。后续的章节将会讨论如何降低这种风险

13.3 族系错误率

族系错误率 (falmily-wise error rate, FWER) 定义为对 m 个原假设 $H_{01}, \ldots H_{0m}$ 进行假设检验时，至少发生一次第一类错误的概率

	H_0 **is True**	H_0 **is False**	**Total**
Reject H_0	V	S	R
Do Not Reject H_0	U	W	$m - R$

	H_0 is True	H_0 is False	Total
Total	m_0	$m-m_0$	m

根据上表，族系错误率即为：

$$\mathrm{FWER} = \Pr(V \geq 1)$$

拒绝 p 值低于 α 的任何原假设的策略（即在 α 级别控制每个原假设的 I 类误差）会导致 FWER为：

$$\begin{aligned}\mathrm{FWER}(\alpha) &= 1 - \Pr(V = 0) \\ &= 1 - \Pr(\text{do not falsely reject any null hypotheses}) \\ &= 1 - \Pr\left(\bigcap_{j=1}^{m}\{\text{do not falsely reject } H_{0j}\}\right)\end{aligned}$$

当我们作一个极强的假设，即 m 个检验都是独立的，且 m 个原假设都正确时，有：

$$\mathrm{FWER}(\alpha) = 1 - \prod_{j=1}^{m}(1-\alpha) = 1 - (1-\alpha)^m$$

根据上式，当我们只检验一个原假设时，族系错误率等同于第一类错误率；若执行 $m = 100$ 个独立测试，则族系错误率为 $1-(1-\alpha)^{100}$，当 $\alpha = 0.05$ 时，这个错误率将达到 0.994，几乎相当于必然出现一个第一类错误。因此，将族系错误率控制在 α 水平，比将第一类错误率控制在 α 水平的要求要高得多

13.3.2 控制族系错误率的方法

Bonferroni 方法

设 A_j 表示事件：我们对第 j 个原假设犯第一类错误，其中 $j = 1, \ldots, m$，则：

$$\begin{aligned}\mathrm{FWER} &= \Pr(\text{falsely reject at least one null hypothesis}) \\ &= \Pr(\bigcup_{j=1}^{m} A_j) \\ &\leq \sum_{j=1}^{m} \Pr(A_j).\end{aligned}$$

在上式中，我们基于不等式 $P(A \cup B) \leq P(A) + P(B)$ 得出不论 A 与 B 是否独立，**Bonferroni 方法 (Bonferronni method)** 或**Bonferroni 校验 (Bonferronni correciton)** 将拒绝每个假设检验的阈值设置为 α/m，因此 $P(A_j) \leq \alpha/m$，也即：

$$\mathrm{FWER}(\alpha/m) \leq m \times \frac{\alpha}{m} = \alpha$$

这也就相当于把每个零假设的阈值除以总数 m 来避免族系错误

Bonferroni 方法让我们不会错误地拒绝过多原假设，但是它相当保守，使我们拒绝了很少的原假设，提升了犯第二类错误的概率。Bonferroni 校正是迄今为止所有统计中最著名和最常用的

多重性校正。它的普遍性在很大程度上是由于它非常易于理解和实施，还因为它成功地控制了 I 类错误，而不管 m 假设检验是否独立

> 根据不等式，我们得到的 FWER 比目标 FWER 往往低很多，这表明 Bonferroni 方法过于保守

Holm 逐步递减法
Holm 逐步递减法 (Holm's step-down procedure) 比 Bonferroni 方法稍微复杂，在这个方法中，我们否定每个原假设的阈值取决于所有 m 个 p 值的值，与 Bonferroni 方法要求每个原假设的阈值都低于 α/m 不同。Holm 方法也不对 m 个原假设作独立假设，因此它的功效始终更大

Holm 方法的步骤如下：

1. 给定 FWER 的 α
2. 对 m 个原假设 $H_{01},\dots,H_{0m}$，计算对应 p 值 $p_1,\dots,p_m$
3. 按 p 值对原假设排序
4. 定义

$$L=\min\left\{j:p_{(j)}>\frac{\alpha}{m+1-j}\right\}$$

5. 拒绝首个 $p_j<p_{(L)}$ 及其之后的原假设，并接受之前的原假设

Tukey 方法
前两类方法不对零假设和检验统计量分布作任何假设。在某些特定条件下，我们可使用更适合任务的方法来控制 FWER，**Tukey 方法 (Tukey's method)** 是其中之一

以 5 个基金经理的表现为例，在观察五个基金经理的表现后，我们发现 1 号和二号的均值差异最大，因此可能希望进行假设检验 $H_0:\mu_1\neq\mu_2$，且得到了一个较小的 $p=0.0349$。然而，这并非是我们一开始就执行的假设检验，而是在比较了 5 个样本之后执行的，实际上相当于进行了 $5\times(5-1)/2=10$ 次的假设检验。因此 FWER 需要控制到 0.05，则基于 Bonferroni 方法每对原假设的 $\alpha=0.005$，我们反而不应该拒绝原假设

不过，Bonferroni 校验在这里有点严格，因为没有考虑 $m=10$ 对原假设的关系：实际上 m 个成对比较的 m 个 p 值不是独立的，因此我们的校验可以不那么严格。Tukey 方法允许我们对 $m=G(G-1)/2$ 对均值比较时，在 α 水平控制 FWER，同时拒绝 $p<\alpha_T$，这里 α_T 稍微 $>\alpha/m$

> 原书未提及如何计算 α_T，但代码已封装好

Scheffé 方法
假设我们希望对五个基金经理进行检验：

$$H_0:\frac{1}{2}(\mu_1+\mu_3)=\frac{1}{3}(\mu_2+\mu_4+\mu_5)$$

这个检验可用双样本 t 检验进行，也能得到一个很小的 $p = 0.004$。然而，这实际上也是我们查看了 5 个观测值得到的，据 Bonferroni 校正需要更小的 p 值才能拒绝原假设

Scheffé 方法 (Scheffé's method) 让我们通过计算几个 α_S 值来拒绝上面的原假设，同时确保族错误率低于 α。一旦计算了 α_S 值，我们就可以进行类似于：

$$H_0 : \frac{1}{3}(\mu_1 + \mu_2 + \mu_3) = \frac{1}{2}(\mu_4 + \mu_5)$$

这样的假设检验而不需要调整 α_S 值

> 这里没有提到怎么计算 α_S

Bonferroni 方法和 Holm 方法适用于任何情况的多重假设检验。但是，在一些特定条件下，使用 Scheffé' 方法或者 Tukey 方法可以在保证降低第一类错误率的同时提高假设检验的功效

13.3 FWER 和功效的权衡

回顾 FWER 和功效的定义：

- FWER 表示我们在多重检验中犯至少一次第一类错误的概率
- 功效表示在原假设为假时，成功拒绝原假设的概率，即拒绝零假设的数量除以假零假设的总数
  这里 FWER 保证了我们不发现谬误，而功效则体现我们假设检验获得新发现的作用

随着原假设数量 m 的增大，为了控制 FWER 在一个合适的范围 α 内，我们不得不降低每个假设检验的阈值，这将会导致功效的迅速降低。实践中，当 m 很大时，我们可能愿意容忍假阳性以获得更多的发现，这就是错误发现率背后的动机

13.4 错误发现率

13.4.1 错误发现率背后的直觉

	H_0 **is True**	H_0 **is False**	**Total**
Reject H_0	V	S	R
Do Not Reject H_0	U	W	$m - R$
Total	m_0	$m - m_0$	m

根据上表，我们定义假阳性 V 和总阳性 $V + S = R$ 的比率 V/R 为**错误发现比例(false discovery proportion, FDP)**。当 m 很大时，我们试图确保错误发现率足够低，以便大多数被拒绝的原假设不是假阳性

实践中，控制 FDP 很吸引人，但是几乎不可能完成。因为我们不能知道数据集上哪些假设是错误的，哪些是真实的，即使能控制 FWER 保证 $P(V \geq 1) \leq \alpha$，但是也不能保证 $V = 0$，除非不拒绝任何零假设

因此，我们转向控制 **错误发现率(false discovery rate, FDR)**。它定义为：

$$\mathrm{FDR} = \mathrm{E}(\mathrm{FDP}) = \mathrm{E}(V/R).$$

也就是错误发现比例 FDP 的期望。例如，当设定 FDR 为 20% 时，我们会尽可能多的拒绝原假设，同时保证假阳性的概率不超过 20%

> 期望表示多次重复试验下得到的假阳性率。一次特定实验下，这个值可能超过 FDR

与 p 值不同，p 值通常将低于阈值 (如 0.05) 的结果视为得出阳性结果的最低证据标准，而 FDR 不存在一个广泛接受的阈值，它的阈值选择取决于上下文甚至数据集，例如愿意付出调查的经费数量

13.4.2 Benjamini–Hochberg 程序

Benjamini–Hochberg 程序 (Benjamini–Hochberg Procedure) 是一种用于控制错误发现率 FDR 的方法，可将 m 个 p 值联系到一个设置的好的 FDR 值 q 上。它的步骤如下：

1. 确定 FDR 值 q
2. 计算 m 个原假设的 p 值 $p_1, \ldots, p_m$
3. 按从小到大的顺序排序，即令 $p_{(1)} \leq \cdots \leq p_{(m)}$
4. 定义：

$$L = \max\{j : p_{(j)} < qj/m\}$$

5. 拒绝所有 满足 $P_j \leq p_{(L)}$ 的 H_{0j} 原假设
   简单来说，就是排序后找到一个最大的序号 j，使得 $p_{(j)} \leq \frac{j}{m}q$，拒绝所有前面的原假设，同时后续的所有原假设都不拒绝以控制第一错误率

当 m 个 p 值独立或轻度依赖时， Benjamini–Hochberg 程序就能保证：

$$FDR \leq q$$

换言之，这个过程可以保证平均来说被拒绝的原假设不超过 q 的部分为假阳性，且无论多少个原假设为真，也无论原假设的 p 值是如何分布的

> Benjamini-Hochberg 程序类似于 Holm 方法，拒绝的阈值不是确定的，而是依赖全部数据的 p 值得出的函数，无法预先知道如 $p = 0.01$ 的原假设是否应被拒绝，这取决于其他的原假设 p 值情况

13.5 p 值和错误发现率的重采样方法

之前的讨论基于用某些统计量来检验特定零假设 H_0，这些统计量在 H_0 下具有特定分布，如正态分布、t 分布、χ^2 分布等，这被称为**理论零分布 (theoretical null distribution)**，我们用这些分布计算 p 值。对于绝大多数零假设，只要对数据作出严格假设，我们就可以使用理论零

分布计算 p 值。有时候，我们的原假设 H_0 和检验统计量 T 不再寻常，或者很难建立合适假设(如小样本)，我们则不能使用理论零分布

本章讨论利用计算机的快速计算近似 T 的零分布，我们需要通过对特定问题实例化来实现这个方法。本章考虑的示例是用双样本 t 检验检验两个随机变量的均值是否相等

13.5.1 关于 p 值的重采样方法

考虑零假设 $H_0 : E(X) = E(Y)$，备择假设 $H_a : E(X) \neq E(Y)$，记来自 X 和 Y 的样本数 n_X 和 n_Y，则双样本 t 检验统计量：

$$T = \frac{\hat{\mu}_X - \hat{\mu}_Y}{s\sqrt{\frac{1}{n_X} + \frac{1}{n_Y}}}$$

其中 $\hat{\mu}_X = \frac{1}{n_X}\sum_{i=1}^{n_X} x_i$，$\hat{\mu}_Y = \frac{1}{n_Y}\sum_{i=1}^{n_Y} y_i$，$\quad s = \sqrt{\frac{(n_X-1)s_X^2+(n_Y-1)s_Y^2}{n_X+n_Y-2}}$，且 s_X^2 和 s_Y^2 分别是两组样本的方差的无偏估计。显然，一个较大的 T 值提供了拒绝 H_0 的证据

当 n_X 和 n_Y 都很大时，上述 T 近似服从 $N(0,1)$ 分布；但是当二者很小时，在未知 X 和 Y 的分布情况下，我们就无法得到 T 的理论零分布。此时，我们需要用**重采样 (re-sampling)** 或者具体称为**排列 (permutation)** 的方法进行估计

我们先进行思想实验。设 H_0 成立，且 X 与 Y 同分布，则我们随机交换 X 和 Y 中的观测值，则交换统计值后的检验统计量 T 与原始统计量 T 应当具有相同分布，即仅当 H_0 成立且 X 和 Y 同分布时成立

上述想法提供了一个近似 T 的零分布的方案。取出所有的 $n_X + n_Y$ 的观测样本，并进行 B 次重新排列，这里 B 是一个较大的数，每次排列后重新分配观测样本 X' 和 Y'，并计算 $T^{*1}, \ldots, T^{*B}$ 表示新统计量 T。回顾 p 值是在 H_0 下至少在此极端值观测到检验统计量的概率，因此：

$$p\text{-value} = \frac{\sum_{b=1}^{B} l_{(|T^{*b}| \geq |T|)}}{B}$$

这就是检验统计量的值至少和原始数据的检验统计量一样极端的值的比例。其中 l 是一个指示函数，当排列 T 不小于原始 T 时值为 1

通过实验可以发现。当样本数据分布较为偏态或数据量较小的情况下 (此时理论零分布准确性较低)，重抽样得到的 p 与理论 p 之间差异明显；否则，重抽样 p 与理论 p 之间差异很小

> 建议在数据偏态或小样本下使用重抽样方法。其他时候，优势并不明显

13.5.2 错误发现率的重采样方法

假设我们希望控制 FDR，对于 m 个零假设 $H_{01}, \ldots, H_{0m}$，为了避免使用理论零分布，考虑使用 Benjamini–Hochberg 程序计算 m 个检验统计量 $T_1, \ldots, T_m$，然后获得临界的 $p_{(L)}$ 值并拒绝对应原假设。事实上，可用重采样方法避免计算 p 值

由 FDR 定义 $\mathrm{FDR} = E(V/R)$，则作近似估计：

$$\mathrm{FDR} = E\left(\frac{V}{R}\right) \approx \frac{E(V)}{R}$$

假设我们希望拒绝任何检验统计量超过 c 的任何原假设，然后计算：

$$R = \sum_{j=1}^{m} l_{(|T_j| \geq c)}$$

这里 R 是所有拒绝的原假设数量。然而，我们很难计算 V，因为我们不知道哪些被拒绝的 H_0 是假阳性。实际上我们可使用重采样方法模拟 $H_{01}, \ldots H_{0m}$ 下的数据，然后计算得到的检验统计量 T，以超过 c 的重采样检验统计量 T^* 的数量提供 V 的估计值

记 $x_1^{(j)}, \ldots, x_{n_X}^{(j)}$ 和 $y_1^{(j)}, \ldots, y_{n_Y}^{(j)}$ 表示和第 j 个零假设有关的统计样本，我们按照排列的方法再次随机排布 $n_X + n_Y$ 个样本，并重新分配到两组样本中以计算双样本统计量 T_i，并以超过设定统计量大小 c 的 T_i 数量为 $E(V)$，即错误拒绝 H_0 的数量。重复排列 一个大数 B 次，我们就能得到相应的 $E(V)$ 估计

计算步骤：

1. 选择一个阈值 c，其中 $c > 0$
2. 对于 $j = 1, \cdots, m$：
   - 计算 $T^{(j)}$，即第 j 个原假设 H_{0j} 的原始检验统计量
   - 从 $b = 1, \ldots, B$，其中 B 是一个大数
     - 计算 $n_X + n_Y$ 对观测值的随机排列，重新分配观测值到两个样本中
     - 对新的两对样本，计算 $T^{(j),*b}$
3. 计算 $R = \sum_{j=1}^{m} l_{(|T^{(j)} \geq c)}$
4. 计算 $\hat{V} = \frac{\sum_{b=1}^{B} \sum_{j=1}^{m} 1_{\left(|T^{(j),*b}| \geq c\right)}}{B}$
5. 估计 FDR，使用 $\hat{V}/R$

事实上，如果定义 p 值：

$$p_j = \frac{\sum_{j'=1}^{m} \sum_{b=1}^{B} 1_{(|T_{j'}^{*b}| \geq |T_j|)}}{Bm}$$

其中 $j = 1, \ldots, m$，以代替之前定义的 p 值：

$$p-\mathrm{value} = \frac{\sum_{b=1}^{B} 1_{(|T^{*b}| \geq |T|)}}{B}$$

并使用 Benjamini-Hochberg 程序计算这些重采样的 p 值，实际上与本节介绍的重采样方法等价

> 第一个计算方法一次性考虑了 m 对假设检验的重采样结果，适合多重检验；第二个计算方法适合单对的假设检验

13.5.3 什么时候应用重采样

下面两种情况，使用重采样比较有用：

1. 也许无理论零分布可用。例如，检验异常的原假设 H_0 或使用不常规的检验统计量 T
2. 存在理论零分布时，其有效性假设不成立。例如，仅当观测值正态分布时，双样本检验统计量 t 才服从 $t_{n_X+n_Y-2}$ 分布，且仅 n_X 和 n_Y 很大时，才服从 $N(0,1)$ 分布。当假设不成立时，理论 p 无效

此外，如果还有其他有效的方法重新采样或者重新排列数据，都可以用以重采样变体以计算估计 p 值或者估计 FDR