第四章 机器学习基础

4.1 学习算法

机器学习算法是一种能够从数据中学习的算法。Mitchell(1997)提供了一个简洁的定义："对于某类任务 T 和性能度量 P，一个计算机程序被认为可以从经验 E 中学习是指，通过经验 E 改进后，它在任务 T 上由性能度量 P 衡量的性能有所提升。"

4.1.1 任务 T

通常机器学习任务定义为机器学习系统应该如何处理样本（example）。样本是指我们从某些希望机器学习系统处理的对象或事件中收集到的已经量化的特征（feature）的集合。我们通常会将样本表示成一个向量 $\boldsymbol{x} \in \mathbb{R}^n$，其中向量的每一个元素 x_i 是一个特征。例如，一张图片的特征通常是指这张图片的像素值。

- **分类**：在这类任务中，计算机程序需要指定某些输入属于 k 类中的哪一类。为了完成这个任务，学习算法通常会返回一个函数 $f: \mathbb{R}^n \to \{1, ..., k\}$。
- **输入缺失分类**：当输入向量的每个度量不被保证的时候，分类问题将会变得更有挑战性。为了解决分类任务，学习算法只需要定义一个从输入向量映射到输出类别的函数。当一些输入可能丢失时，学习算法必须学习一组函数，而不是单个分类函数。每个函数对应着分类具有不同缺失输入子集的 x。
- **回归**：在这类任务中，计算机程序需要对给定输入预测数值。为了解决这个任务，学习算法需要输出函数 $f: \mathbb{R}^n \to \mathbb{R}$。
- **转录**：这类任务中，机器学习系统观测一些相对**非结构化**表示的数据，并转录信息为离散的文本形式。
- **机器翻译**：在机器翻译任务中，输入是一种语言的符号序列，计算机程序必须将其转化成另一种语言的符号序列。
- **结构化输出**：结构化输出任务的输出是向量或者其他包含多个值的数据结构，并且构成输出的这些不同元素间具有重要关系。
- **异常检测**：在这类任务中，计算机程序在一组事件或对象中筛选，并标记不正常或非典型的个体。
- **合成和采样**：在这类任务中，机器学习程序生成一些和训练数据相似的新样本。通过机器学习，合成和采样可能在媒体应用中非常有用，可以避免艺术家大量

昂贵或者乏味费时的手动工作。

- **缺失值填补**：在这类任务中，机器学习算法给定一个新样本 $\boldsymbol{x} \in \mathbb{R}^n$， $\boldsymbol{x}$ 中某些元素 x_i 缺失。算法必须填补这些缺失值。
- **去噪**：在这类任务中，机器学习算法的输入是，干净样本 $\boldsymbol{x} \in \mathbb{R}^n$ 经过未知损坏过程后得到的损坏样本 $\tilde{\boldsymbol{x}} \in \mathbb{R}^n$。算法根据损坏后的样本 $\tilde{\boldsymbol{x}}$ 预测干净的样本 $\boldsymbol{x}$，或者更一般地预测条件概率分布 $p(\boldsymbol{x} \mid \tilde{\boldsymbol{x}})$。
- **密度估计或概率质量函数估计**：在密度估计问题中，机器学习算法学习函数 $p_{\text{model}} : \mathbb{R}^n \rightarrow \mathbb{R}$，其中 $p_{\text{model}}(\boldsymbol{x})$ 可以解释成样本采样空间的概率密度函数（如果 $\boldsymbol{x}$ 是连续的）或者概率质量函数（如果 $\boldsymbol{x}$ 是离散的）。

4.1.2 性能度量 P

对于诸如分类、缺失输入分类和转录任务，我们通常度量模型的准确率（accuracy）。准确率是指该模型输出正确结果的样本比率。我们也可以通过错误率（error rate）得到相同的信息。错误率是指该模型输出错误结果的样本比率。

通常，我们会更加关注机器学习算法在未观测数据上的性能如何，因为这将决定其在实际应用中的性能。因此，我们使用测试集（test set）数据来评估系统性能，将其与训练机器学习系统的训练集数据分开。

4.1.3 经验 E

根据学习过程中的不同经验，机器学习算法可以大致分类为无监督（unsupervised）算法和监督（supervised）算法。

无监督学习算法（unsupervised learning algorithm）训练含有很多特征的数据集，然后学习出这个数据集上有用的结构性质。在深度学习中，我们通常要学习生成数据集的整个概率分布，显式地，比如密度估计，或是隐式地，比如合成或去噪。还有一些其他类型的无监督学习任务，例如聚类，将数据集分成相似样本的集合。

监督学习算法（supervised learning algorithm）训练含有很多特征的数据集，不过数据集中的样本都有一个标签（label）或目标（target）。

4.1.4 示例：线性回归

我们将机器学习算法定义为，通过经验以提高计算机程序在某些任务上性能的算法。这个定义有点抽象。为了使这个定义更具体点，我们展示一个简单的机器学习示例：线性回归（linear regression）。

线性回归解决回归问题，目标是建立一个系统，将向量 $\boldsymbol{x} \in \mathbb{R}^n$ 作为输入，预测标量

$y \in \mathbb{R}$ 作为输出。线性回归的输出是其输入的线性函数。令 $\hat{y}$（预测值）表示模型预测 y 应该取的值。我们定义输出为

$$\hat{y} = \boldsymbol{w}^T \boldsymbol{x}$$

其中 $\boldsymbol{w} \in \mathbb{R}^n$ 为参数（parameter）向量。

因此，我们可以定义任务 T：通过输出 $\hat{y} = \boldsymbol{w}^T \boldsymbol{x}$ 从 $\boldsymbol{x}$ 预测 y 。

假设我们有 m 个输入样本组成的设计矩阵，我们不用它来训练模型，而是评估模型性能如何。我们也有每个样本对应的正确值 y 组成的回归目标向量。因为这个数据集只是用来评估性能，我们称之为测试集（test set）。我们将输入的设计矩阵记作 $\mathbf{X}^{(test)}$，回归目标向量记作 $y^{(test)}$ 。

度量模型性能的一种方法是计算模型在测试集上的均方误差（mean squared error）。如果 $\hat{y}^{(test)}$ 表示模型在测试集上的预测值，那么均方误差表示为：

$$\mathrm{MSE}_{test} = \frac{1}{m}\sum_i (\hat{\boldsymbol{y}}^{(test)} - \boldsymbol{y}^{(test)})_i^2$$

为了构建一个机器学习算法，我们需要设计一个算法，通过观察训练集获得经验（$\boldsymbol{X}^{(train)}, \boldsymbol{y}^{(train)}$），减少 MSE_{test} 以改进权重 w。一种直观方式是最小化训练集上的均方误差，即 MSE_{train} 。

最小化 MSE_{train}，我们可以简单地求解其导数为 0 的情况：

$$\begin{aligned}
&\nabla_\omega \mathrm{MSE}_{train} = 0 \\
&\Rightarrow \nabla_\omega \frac{1}{m}\left\|\hat{\boldsymbol{y}}^{(train)} - \boldsymbol{y}^{(train)}\right\|_2^2 = 0 \\
&\Rightarrow \frac{1}{m}\nabla_\omega \left\|\boldsymbol{X}^{(train)}\boldsymbol{w} - \boldsymbol{y}^{(train)}\right\|_2^2 = 0 \\
&\Rightarrow \nabla_\omega \left(\boldsymbol{X}^{(train)}\boldsymbol{w} - \boldsymbol{y}^{(train)}\right)^{\mathrm{T}} \left(\boldsymbol{X}^{(train)}\boldsymbol{w} - \boldsymbol{y}^{(train)}\right) = 0 \\
&\Rightarrow \nabla_\omega \left(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{X}^{(train)\mathrm{T}}\boldsymbol{X}^{(train)}\boldsymbol{w} - 2\boldsymbol{w}^{\mathrm{T}}\boldsymbol{X}^{(train)\mathrm{T}}\boldsymbol{y}^{(train)} + \boldsymbol{y}^{(train)\mathrm{T}}\boldsymbol{y}^{(train)}\right) = 0 \\
&\Rightarrow 2\boldsymbol{X}^{(train)\mathrm{T}}\boldsymbol{X}^{(train)}\boldsymbol{w} - 2\boldsymbol{X}^{(train)\mathrm{T}}\boldsymbol{y}^{(train)} = 0 \\
&\Rightarrow \boldsymbol{w} = \left(\boldsymbol{X}^{(train)\mathrm{T}}\boldsymbol{X}^{(train)}\right)^{-1}\boldsymbol{X}^{(train)\mathrm{T}}\boldsymbol{y}^{(train)} \rightarrow \text{正规方程（normal equation）}
\end{aligned}$$

4.2 容量、过拟合和欠拟合

在我们的线性回归示例中，我们通过最小化训练误差来训练模型，

$$\frac{1}{m^{(train)}}\nabla_{\omega}\left\|\boldsymbol{X}^{(train)}\boldsymbol{w}-\boldsymbol{y}^{(train)}\right\|_2^2$$

但是我们真正关注的是测试误差 $\frac{1}{m^{(test)}}\nabla_{\omega}\left\|\boldsymbol{X}^{(test)}\boldsymbol{w}-\boldsymbol{y}^{(test)}\right\|_2^2$。

以下是决定机器学习算法效果是否好的因素：

1、降低训练误差。

2、缩小训练误差和测试误差的差距。

这两个因素对应机器学习的两个主要挑战：欠拟合（underfitting）和过拟合（overfitting）。欠拟合是指模型不能在训练集上获得足够低的误差。而过拟合是指训练误差和和测试误差之间的差距太大。

通过调整模型的容量（capacity），我们可以控制模型是否偏向于过拟合或者欠拟合。通俗地，模型的容量是指其拟合各种函数的能力（也可以理解为参数个数，容量越大需要学习的参数越多）。容量低的模型可能很难拟合训练集。容量高的模型可能会过拟合，因为记住了不适用于测试集的训练集性质。

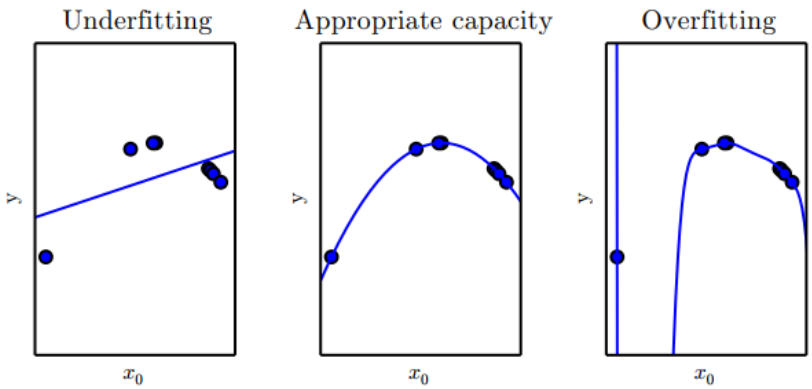


图 25 我们用三个模型拟合了这个训练集的样本。训练数据是通过随机抽取 x 然后用二次函数确定性地生成 y 来合成的。(左)用一个线性函数拟合数据会导致欠拟合——它无法捕捉数据中的曲率信息。(中)用二次函数拟合数据在未观察到的点上泛化得很好。这并不会导致明显的欠拟合或者过拟合。(右)一个 9 阶的多项式拟合数据会导致过拟合。在这里我们使用 Moore-Penrose 伪逆来解这个欠定的正规方程。得出的解能够精确地穿过所有的训练点，但可惜我们无法提取有效的结构信息。在两个数据

点之间它有一个真实的函数所不包含的深谷。在数据的左侧，它也会急剧增长，而在这一区域真实的函数却是下降的。

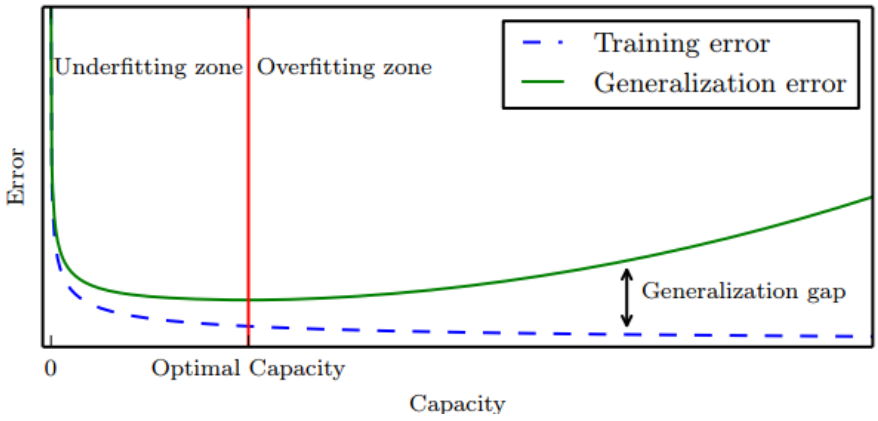


图 26 容量和误差之间的典型关系。训练误差和测试误差表现得非常不同。在图的左端，训练误差和泛化误差都非常高。这是欠拟合机制（underfitting regime）。当我们增加容量时，训练误差减小，但是训练误差和泛化误差之间的间距却不断扩大。最终，这个间距的大小超过了训练误差的下降，我们进入到了过拟合机制（overfitting regime），其中容量过大，超过了最佳容量（optimal capacity）。

奥卡姆剃刀原则：同样能够解释已知观测现象的假设中，我们应该挑选“最简单”的那一个。

没有免费午餐定理：不存在能够在所有可能的分类问题中性能均为最优的算法。

正则化：修改学习算法，使其降低泛化误差而非训练误差。（L1 正则化，L2 正则化）

$$J(\boldsymbol{w}) = \text{MSE}_{\text{train}} + \lambda \boldsymbol{w}^{\top} \boldsymbol{w}$$

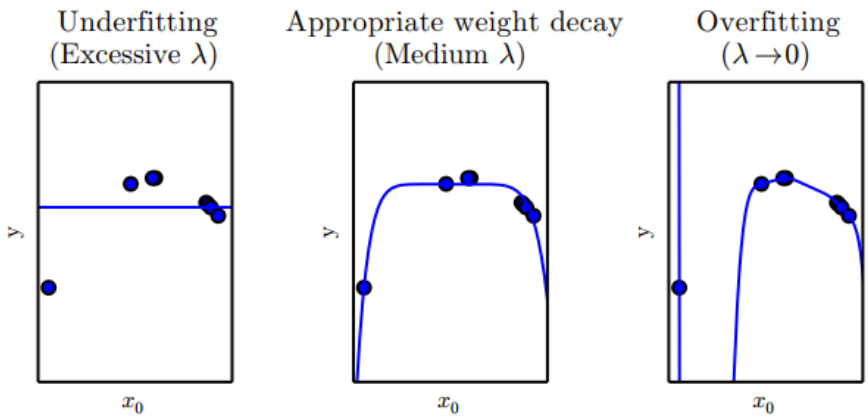


图 27 我们通过改变权重衰减的量来避免高阶模型的过拟合问题。(左)当 λ 非常大时，我们可以强迫模型学习到了一个没有斜率的函数。由于它只能表示一个常数函数，所以会导致欠拟合。(中)取一个适当的 λ 时，学习算法能够用一个正常的形状来恢复曲率。即使模型能够用更复杂的形状来来表示

函数，权重衰减鼓励用一个带有更小参数的更简单的模型来描述它。(右)当权重衰减趋近于 0（即使用 Moore-Penrose 伪逆来解这个带有最小正则化的欠定问题）时，这个 9 阶多项式会导致严重的过拟合

4.3 超参数和验证集

用于学习参数的数据子集通常仍被称为训练集，尽管这会和整个训练过程用到的更大的数据集相混。用于挑选超参数的数据子集被称为验证集（validation set）。通常，80%的训练数据用于训练，20%用于验证。由于验证集是用来“训练”超参数的，尽管验证集的误差通常会比训练集误差小，验证集会低估泛化误差。所有超参数优化完成之后，泛化误差可能会通过测试集来估计。

4.3.1 交叉验证

将数据集分成固定的训练集和固定的测试集后，若测试集的误差很小，这将是有问题的。一个小规模的测试集意味着平均测试误差估计的统计不确定性，使得很难判断算法 A 是否比算法 B 在给定的任务上做得更好。

最常见的是 k-折交叉验证过程，将数据集分成 k 个不重合的子集。测试误差可以估计为 k 次计算后的平均测试误差，在第 i 次测试时，数据的第 i 个子集用于测试集，其他的数据用于训练集，下面分别给出了 10-折交叉验证示意图和 k-折交叉验证算法。

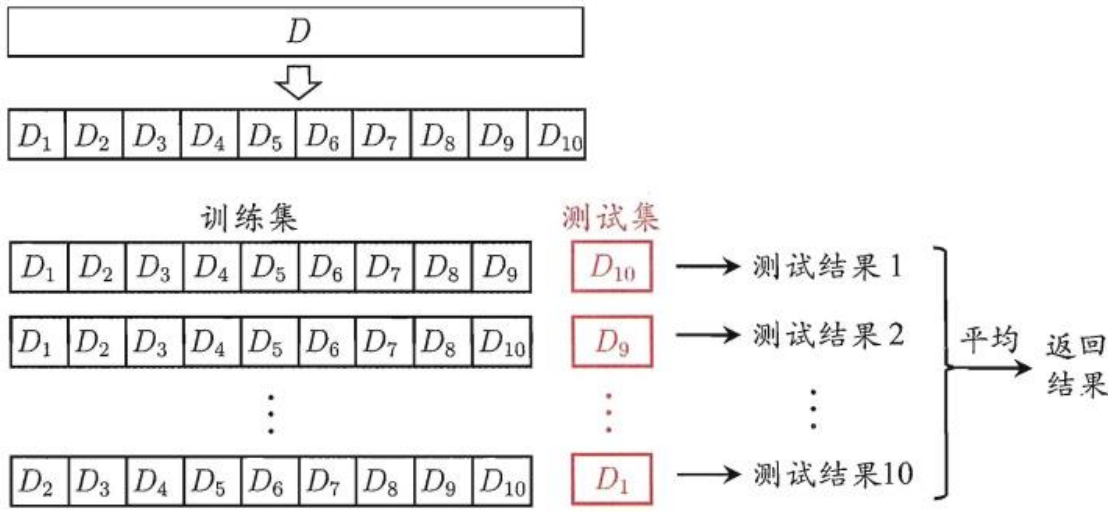


图 28 10-折交叉验证示意图

算法 5.1 k-折交叉验证算法。当给定数据集 $\mathbb{D}$ 对于简单的训练/测试或训练/验证分割而言太小难以产生泛化误差的准确估计时（因为在小的测试集上，L 可能具有过高的方差），k-折交叉验证算法可以用于估计学习算法 A 的泛化误差。数据集 $\mathbb{D}$ 包含的元素是抽象的样本 $\boldsymbol{z}^{(i)}$（对于第 i 个样本），在监督学习的情况代表（输入，目标）对 $\boldsymbol{z}^{(i)} = (\boldsymbol{x}^{(i)}, y^{(i)})$，或者无监督学习的情况下仅用于输入 $\boldsymbol{z}^{(i)} = \boldsymbol{x}^{(i)}$。该算法返回 $\mathbb{D}$ 中每个示例的误差向量 $\boldsymbol{e}$，其均值是估计的泛化误差。单个样本上的误差可用于计算平均值周围的置信区间（式 (5.47)）。虽然这些置信区间在使用交叉验证之后不能很好地证明，但是通常的做法是只有当算法 A 误差的置信区间低于并且不与算法 B 的置信区间相交时，我们才声明算法 A 比算法 B 更好。

Define KFoldXV($\mathbb{D}, A, L, k$):
Require: $\mathbb{D}$ 为给定数据集，其中元素为 $\boldsymbol{z}^{(i)}$
Require: A 为学习算法，可视为一个函数（使用数据集作为输入，输出一个学好的函数）
Require: L 为损失函数，可视为来自学好的函数 f，将样本 $\boldsymbol{z}^{(i)} \in \mathbb{D}$ 映射到 $\mathbb{R}$ 中标量的函数
Require: k 为折数
  将 $\mathbb{D}$ 分为 k 个互斥子集 $\mathbb{D}_i$，它们的并集为 $\mathbb{D}$
  for i from 1 to k **do**
    $f_i = A(\mathbb{D} \backslash \mathbb{D}_i)$
    for $\boldsymbol{z}^{(j)}$ in $\mathbb{D}_i$ **do**
      $e_j = L(f_i, \boldsymbol{z}^{(j)})$
    end for
  end for
  Return $\boldsymbol{e}$

图 29 k-折交叉验证算法

所以在实际工作中一般会有训练集、交叉验证集和测试集，如图 30 所示。

训练集：用于训练数据或样本。

交叉验证集：判断学习率是否要调整，何时结束训练。一般来讲，训练数据每过一轮(one epoch)，都要在交叉验证集上看一下性能(如损失函数)，由此做一些来判断。

测试集：判断模型的性能好坏，而不再对参数有什么优化。

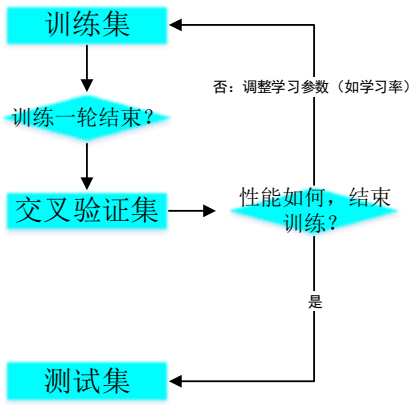


图 30 交叉验证过程

4.4 估计、偏差和方差

4.4.1 点估计

点估计试图为一些感兴趣的量提供单个“最优”预测。一般地，感兴趣的量可以是单个参数（比如估计高斯分布的均值 u 为 0.1），或是某些参数模型中的一个向量参数（如线性回归中的权重向量），也有可能是整个函数（如线性回归中的函数映射）。

将参数 θ 的点估计表示为 $\hat{\theta}$。令 $\{\boldsymbol{x}^{(1)},\cdots\boldsymbol{x}^{(m)}\}$ 是 m 个独立同分布的数据点。点估计（point estimator）或统计量（statistics）是这些数据的任意函数：

$$\hat{\theta}_m = g\left(\{\boldsymbol{x}^{(1)},\cdots\boldsymbol{x}^{(m)}\}\right)$$

我们假设真实参数 θ 是固定但未知的，而点估计 $\hat{\theta}$ 是数据的函数。由于数据是随机过程采样出来的，数据的任何函数都是随机的。因此 $\hat{\theta}$ 是一个随机变量。

4.4.2 偏差

估计（一般就是指点估计）的偏差被定义为：

$$\text{bias}(\hat{\boldsymbol{\theta}}_m) = \mathbb{E}(\hat{\boldsymbol{\theta}}_m) - \boldsymbol{\theta}$$

其中期望作用在所有数据（看作是从随机变量采样得到的）上，$\boldsymbol{\theta}$ 是用于定义数据生成分布的 $\boldsymbol{\theta}$ 的真实值。如果 $\text{bias}(\hat{\boldsymbol{\theta}}_m) = 0$，那么估计量 $\hat{\boldsymbol{\theta}}_m$ 被称为是无偏（unbiased），这意味着 $\mathbb{E}(\hat{\boldsymbol{\theta}}_m) = \boldsymbol{\theta}$。如果 $\lim_{m\to\infty} \text{bias}(\hat{\boldsymbol{\theta}}_m) = 0$，那么估计量 $\hat{\boldsymbol{\theta}}_m$ 被称为是渐近无偏（asymptotically unbiased），这意味着 $\lim_{m\to\infty} \mathbb{E}(\hat{\boldsymbol{\theta}}_m) = \boldsymbol{\theta}$。

4.4.2.1 示例 1：伯努利分布

考虑一组服从均值为 θ 的伯努利分布的独立同分布的样本 $\{x^{(1)},\cdots x^{(m)}\}$：

$$P(x^{(i)};\theta) = \theta^{x^{(i)}}(1-\theta)^{(1-x^{(i)})}$$

这个分布中参数θ的常用估计量是训练样本的均值：

$$\hat{\theta}_m = \frac{1}{m}\sum_{i=1}^{m} x^{(i)}$$

则有

$$\begin{aligned}
\text{bias}(\hat{\theta}_m) &= \mathbb{E}(\hat{\theta}_m) - \theta \\
&= \mathbb{E}\left[\frac{1}{m}\sum_{i=1}^{m} x^{(i)}\right] - \theta \\
&= \frac{1}{m}\sum_{i=1}^{m} \mathbb{E}\left[x^{(i)}\right] - \theta \\
&= \frac{1}{m}\sum_{i=1}^{m} \sum_{x^{(i)}=0}^{1} \left(x^{(i)}\theta^{x^{(i)}}(1-\theta)^{(1-x^{(i)})}\right) - \theta \\
&= \frac{1}{m}\sum_{i=1}^{m} (\theta) - \theta \\
&= \theta - \theta = 0
\end{aligned}$$

期望定义：取值乘以概率，然后再求和

因为$\text{bias}(\hat{\theta}_m)=0$，所以称估计$\hat{\theta}$是无偏的。

4.4.2.2 示例 2：均值的高斯分布估计

考虑一组独立同分布的样本$\{x^{(1)},\cdots x^{(m)}\}$服从高斯分布，其中$p(x^{(i)})=N(x^{(i)};\mu,\sigma^2)$。高斯概率密度函数如下：

$$p(x^{(i)};\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(-\frac{1}{2}\frac{\left(x^{(i)}-\mu\right)^2}{\sigma^2}\right)$$

高斯均值参数的常用估计量被称为样本均值（sample mean）：

$$\hat{\mu}_m = \frac{1}{m}\sum_{i=1}^{m} x^{(i)}$$

则有

$$\begin{aligned}\text{bias}(\hat{\mu}_m) &= \mathbb{E}(\hat{\mu}_m) - \mu \\ &= \mathbb{E}\left(\frac{1}{m}\sum_{i=1}^{m} x^{(i)}\right) - \mu \\ &= \left(\frac{1}{m}\sum_{i=1}^{m} \mathbb{E}\left[x^{(i)}\right]\right) - \mu \\ &= \left(\frac{1}{m}\sum_{i=1}^{m} \mu\right) - \mu \\ &= \mu - \mu = 0\end{aligned}$$

因此样本均值是高斯均值参数的无偏估计量。

4.4.2.3 示例 3：高斯分布方差估计

我们比较高斯分布方差参数 σ^2 的两个不同估计。

我们考虑的第一个方差估计被称为样本方差（sample variance）：

$$\hat{\sigma}_m^2 = \frac{1}{m}\sum_{i=1}^{m}(x^{(i)} - \hat{\mu}_m)^2$$

其中 $\hat{\mu}_m$ 是样本均值。

则有（下面会用到一些基础的概率知识）

$$
\begin{aligned}
\text{bias}(\hat{\sigma}_m^2) &= \mathbb{E}\left[\hat{\sigma}_m^2\right] - \sigma^2 \\
&= \mathbb{E}\left[\frac{1}{m}\sum_{i=1}^{m}(x^{(i)} - \hat{\mu}_m)^2\right] - \sigma^2 \\
&= \frac{1}{m}\mathbb{E}\left[\sum_{i=1}^{m}((x^{(i)})^2 - 2\hat{\mu}_m x^{(i)} + \hat{\mu}_m{}^2)\right] - \sigma^2 \\
&= \frac{1}{m}\mathbb{E}\left[\sum_{i=1}^{m}(x^{(i)})^2 - 2\hat{\mu}_m\sum_{i=1}^{m}x^{(i)} + \sum_{i=1}^{m}\hat{\mu}_m{}^2)\right] - \sigma^2 \\
&= \frac{1}{m}\mathbb{E}\left[\sum_{i=1}^{m}(x^{(i)})^2 - 2m\hat{\mu}_m{}^2 + m\hat{\mu}_m{}^2)\right] - \sigma^2 \\
&= \frac{1}{m}\mathbb{E}\left[\sum_{i=1}^{m}(x^{(i)})^2 - m\hat{\mu}_m{}^2)\right] - \sigma^2 \\
&= \frac{1}{m}\left(\sum_{i=1}^{m}\mathbb{E}\left((x^{(i)})^2\right) - m\mathbb{E}\left(\hat{\mu}_m{}^2\right)\right) - \sigma^2 \\
&= \frac{1}{m}\left(\sum_{i=1}^{m}\left(D(x^{(i)}) + \left(\mathbb{E}\left(x^{(i)}\right)\right)^2\right) - m\mathbb{E}\left(\hat{\mu}_m{}^2\right)\right) - \sigma^2 \\
&= \frac{1}{m}\left(\sum_{i=1}^{m}\left(\sigma^2 + \mu^2\right) - m\mathbb{E}\left(D(\hat{\mu}_m) + \left(\mathbb{E}(\hat{\mu}_m)\right)^2\right)\right) - \sigma^2 \\
&= \frac{1}{m}\left(m\sigma^2 + m\mu^2 - m\mathbb{E}\left(\frac{1}{m}\sigma^2 + \mu^2\right)\right) - \sigma^2 \\
&= \frac{1}{m}\left(m\sigma^2 + m\mu^2 - \sigma^2 - m\mu^2\right) - \sigma^2 \\
&= \frac{m-1}{m}\sigma^2 - \sigma^2 = -\frac{\sigma^2}{m}
\end{aligned}
$$

因此样本方差是有偏估计。

无偏样本方差（unbiased sample variance）估计为 $\tilde{\sigma}_m^2 = \frac{1}{m-1}\sum_{i=1}^{m}(x^{(i)} - \hat{\mu}_m)^2$。

则有

$$
\begin{aligned}
\text{bias}(\tilde{\sigma}_m^2) &= \mathbb{E}\left[\tilde{\sigma}_m^2\right] - \sigma^2 \\
&= \mathbb{E}\left[\frac{1}{m-1}\sum_{i=1}^{m}(x^{(i)} - \hat{\mu}_m)^2\right] - \sigma^2 \\
&= \frac{1}{m-1}\mathbb{E}\left[\sum_{i=1}^{m}(x^{(i)} - \hat{\mu}_m)^2\right] - \sigma^2 \\
&= \frac{1}{m-1}m\mathbb{E}\left[\frac{1}{m}\sum_{i=1}^{m}(x^{(i)} - \hat{\mu}_m)^2\right] - \sigma^2 \\
&= \frac{m}{m-1}\mathbb{E}\left[\hat{\sigma}_m^2\right] - \sigma^2 \\
&= \frac{m}{m-1}\left(\frac{m-1}{m}\sigma^2\right) - \sigma^2 = 0
\end{aligned}
$$

4.4.3 方差和标准差

我们有时会考虑估计量的另一个性质是它作为数据样本的函数，期望的变化程度是多少。正如我们可以计算估计量的期望来决定它的偏差，我们也可以计算它的方差。估计量的方差（variance）就是一个方差 $\text{Var}(\hat{\theta})$。方差的平方根被称为标准差（standard error），记作 $\text{SE}(\hat{\theta})$。

估计量的方差或标准差告诉我们，当独立地从潜在的数据生成过程中重采样数据集时，如何期望估计的变化。正如我们希望估计的偏差较小，我们也希望其方差较小。

均值的标准差被记作

$$
\text{SE}(\hat{\mu}_m) = \sqrt{\text{Var}\left[\frac{1}{m}\sum_{i=1}^{m}x^{(i)}\right]} = \frac{\sigma}{\sqrt{m}}
$$

均值的标准差在机器学习实验中非常有用。我们通常用测试集样本的误差均值来估计泛化误差。测试集中样本的数量决定了这个估计的精确度。中心极限定理告诉我们均值会接近一个高斯分布，我们可以用标准差计算出真实期望落在选定区间的概率。例如，以均值 $\hat{\mu}_m$ 为中心的 95%置信区间是

$$
\left(\hat{\mu}_m - 1.96\text{SE}(\hat{\mu}_m), \hat{\mu}_m + 1.96\text{SE}(\hat{\mu}_m)\right)
$$

以上区间是基于均值 $\hat{\mu}_m$ 和方差 $\text{SE}(\hat{\mu}_m)$ 的高斯分布。在机器学习实验中，我们通常说算法 A 比算法 B 好，是指算法 A 的误差的 95%置信区间的上界小于算法 B 的误差的 95%置信区间的下界。

4.4.4 权衡偏差和方差以最小化均方误差

偏差和方差度量着估计量的两个不同误差来源。偏差度量着偏离真实函数或参数的误差期望。而方差度量着数据上任意特定采样可能导致的估计期望的偏差。

判断这种权衡最常用的方法是交叉验证。经验上，交叉验证在真实世界的许多任务中都非常成功。另外，我们也可以比较这些估计的均方误差（mean squared error,MSE）：

$$
\begin{aligned}
\mathrm{MSE} &= \mathbb{E}\left[\left(\hat{\theta}_m - \theta\right)^2\right] \\
&= \mathbb{E}\left[\left(\left(\hat{\theta}_m - \mathbb{E}\left(\hat{\theta}_m\right)\right) + \left(\mathbb{E}\left(\hat{\theta}_m\right) - \theta\right)\right)^2\right] \\
&= \mathbb{E}\left(\hat{\theta}_m - \mathbb{E}\left(\hat{\theta}_m\right)\right)^2 + \left(\mathbb{E}\left(\hat{\theta}_m - \theta\right)\right)^2 + 2\mathbb{E}\left(\left(\hat{\theta}_m - \mathbb{E}\left(\hat{\theta}_m\right)\right)\left(\mathbb{E}\left(\hat{\theta}_m\right) - \theta\right)\right) \\
&= \left(\mathbb{E}\left(\hat{\theta}_m - \theta\right)\right)^2 + \mathbb{E}\left(\hat{\theta}_m - \mathbb{E}\left(\hat{\theta}_m\right)\right)^2 + 2\mathbb{E}\left(\left(\hat{\theta}_m - \mathbb{E}\left(\hat{\theta}_m\right)\right)\left(\mathbb{E}\left(\hat{\theta}_m\right) - \theta\right)\right) \\
&= \mathrm{Bias}(\hat{\theta}_m)^2 + \mathrm{Var}(\hat{\theta}_m)
\end{aligned}
$$

MSE 度量着估计和真实参数θ之间平方误差的总体期望偏差。MSE 估计包含了偏差和方差。理想的估计具有较小的 MSE 或是在检查中会稍微约束它们的偏差和方差。

偏差和方差的关系和机器学习容量、欠拟合和过拟合的概念紧密相联。用 MSE 度量泛化误差（偏差和方差对于泛化误差都是有意义的）时，增加容量会增加方差，降低偏差（过拟合）。

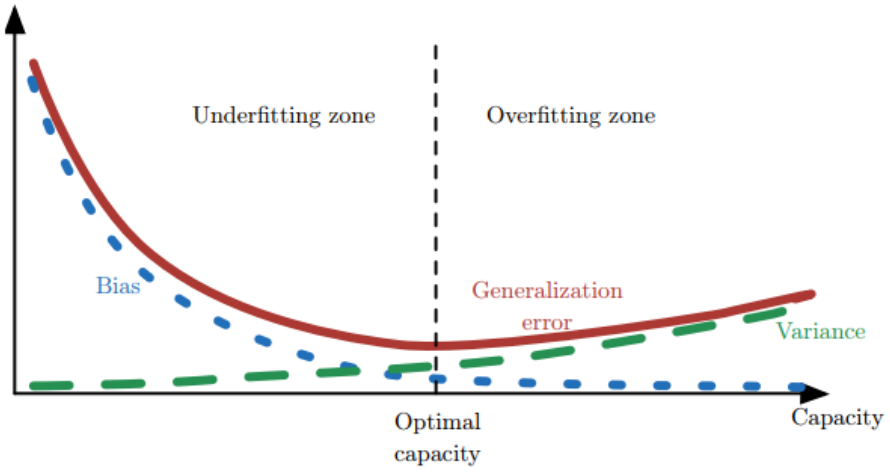


图 31 当容量增大（x 轴）时，偏差（用点表示）随之减小，而方差（虚线）随之增大，使得泛化误差（加粗曲线）产生了另一种 U 形。如果我们沿着轴改变容量，会发现最佳容量，当容量小于最佳容量会呈现欠拟合，大于时导致过拟合

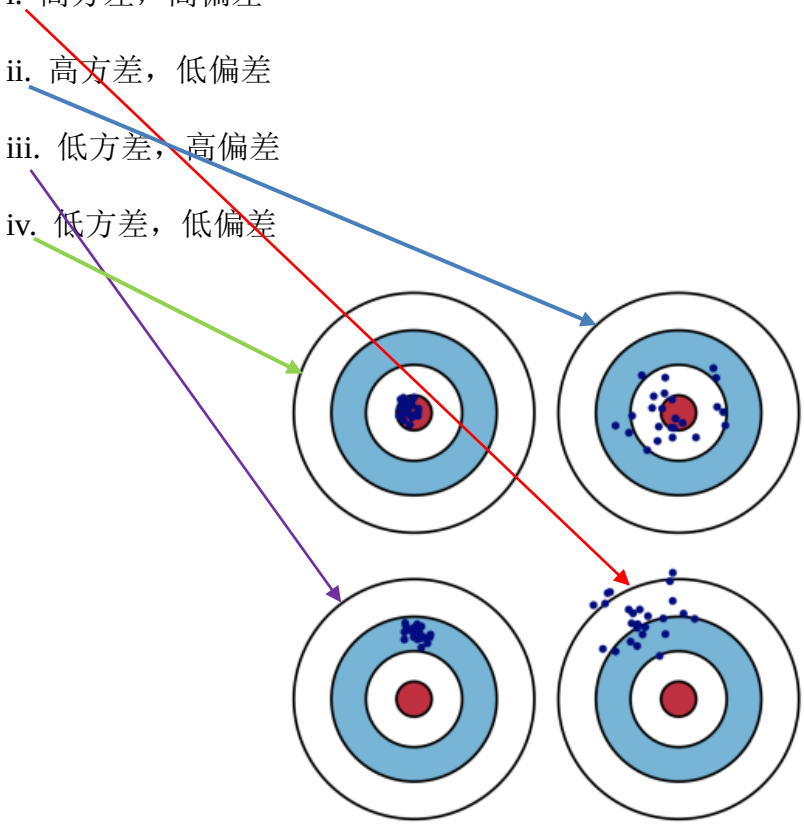
下图中靶心为我们要预测的真实值，命中的点离靶心越远，预测结果越差。假设有四种学习算法，对每种算法，我们多次重复整个建模过程，得到多个命中点。结果分别为如下四幅图所示。这四幅图分别对应以下哪种情况：

i. 高方差，高偏差

ii. 高方差，低偏差

iii. 低方差，高偏差

iv. 低方差，低偏差



4.4.5 一致性

目前我们已经探讨了固定大小训练集下不同估计量的性质。通常，我们也会关注训练数据增多后估计量的效果。特别地，我们希望当数据集中数据点的数量 m 增加时，点估计会收敛到对应参数的真实值。更形式地，我们想要

$$\text{plim}_{m\to\infty}\hat{\theta}_m = \theta$$

符号 plim 表示依概率收敛。上式表示的条件被称为一致性。

一致性保证了估计量的偏差会随数据样本数目的增多而减少。然而，反过来是不正确的——渐近无偏并不意味着一致性。