第六章、机器学习项目构建与评价

赵涵

2022 年 5 月 4 日

1 机器学习项目流程概述

人类在成长、生活过程中积累了很多的历史与经验。人类定期地对这些经验进行“归纳”，获得了生活的“规律”。当人类遇到未知的问题或者需要对未来进行“推测”的时候，人类使用这些“规律”，对未知问题与未来进行“推测”，从而指导自己的生活和工作。

机器学习中的“训练”与“预测”过程可以对应到人类的“归纳”和“推测”过程。通过这样的对应，我们可以发现，机器学习的思想并不复杂，仅仅是对人类在生活中学习成长的一个模拟。由于机器学习不是基于编程形成的结果，因此它的处理过程不是因果的逻辑，而是通过归纳思想得出的相关性结论。

对于一个机器学习项目的构建，一般分为四个部分：

1. 数据收集。
2. 数据清洗。
3. 特征工程。
4. 数据建模与模型评价。

下面我们一一介绍相关的内容。

2 数据清洗

在介绍数据清洗之前，还需要简单说一下，数据的收集，所谓的数据收集，方式其中多种多样，最常见的网络下载，网络爬虫，数据库读取，还有开放数据，调查问卷...根据项目的问题，选择合适的方法。由于当下的时代是一个数据爆炸的时代，一般情况下，不会存在数据量过少的问题，在目前比较主流的领域，计算机视觉，自然语言处理，语音识别，网络上都有开源的数据集供给研究者和相关领域的老师和学生下载。

在收集数据时，会出现一种问题，就是数据不平衡，数据不平衡是指数据集中各类样本数量不均衡的情况。常用不平衡处理方法有采样和代价敏感学习采样欠采样、过采样和综合采样的方法。所谓的代价敏感学习，就是指为不同类别的样本提供不同的权重，从而让机器学习模型进行学习的一种方法。具体的方法，可以参看教科书对应的章节。

数据分为两大类，一类是结构化数据，包括报表，表格，财务记账，二手房数据库，二手车信息等，一般情况下，只要能在表格里录入的数据，都可以称为结构化数据；另一类称为非结构化数据，这一类数据主要是为了与第一类数据进行区分，只要不能存储在表格里的数据，都可以泛称非结构化数据，例如图片，文本，音频，流程图等。

在数据收集完成后，下一步就是数据清洗，数据清洗是指发现并纠正数据文件中可识别的错误的最后一道程序，包括检查数据一致性，处理无效值和缺失值等。与问卷审核不同，录入后的数据清理一般是由计算机而不是人工完成。我们用文本数据来做一个例子，比如一段文字，正常情况下，应该具备完整的语义，正确的文字，正确的标点等，但是在收集和人工录入的过程中，难免会有拼写错误，标点缺失，错别字等，这类数据，都统一称为脏数据，对于整个数据集，首先要建立一个异常检测的模型或者流程，把这些脏数据尽可能地甄别出来，满足建立模型的数据要求。这一部分，我们在此不在过多涉及，异常检测相关内容，我们放在无监督学习的章节中进行介绍。

在数据清洗完成后，一般情况下，在建立模型的预备阶段，会通过数据可视化，进行**探索性数据分析**（Exploratory data analysis），主要的工作就是数据可视化，先讨论一下结构化数据，对于结构化数据，做一些可视图，比如各个量之间均值，方差，整个数据集的中位数，众数，标签的分布等一些最常见的表示，此举的目的就是为了了解数据的性质，为后面建立模型提供一个参考。

3 特征工程

获取原始数据并提取或创建新特征的过程称为**特征工程**（Feature Engineering）。这可能意味着需要对变量进行变换，例如自然对数和平方根，或者对分类变量进行one-hot编码，以便它们可以在模型中使用。特征工程在数据挖掘中有举足轻重的位置，数据领域一致认为：数据和特征决定了机器学习的上限，而模型和算法只能逼近这个上限而已。

3.1 特征选择

对于一个数据点来说，它的属性或者特征可能会非常多，其中有一些特征，可能对目标任务的影响微乎其微，举个例子来说，对于房价预测问题，房子有一个特征是所在社区的绿化，这个特征来说，相对于房子面积，户型，地段这些特征相比，对房价的影响来说，可以忽略（可能对于追求更高生活品质的人来说，这个特征相对重要）。如何选择特征构造合适的模型，称为**特征选择**（feature selection）。假设一个数据具有d个特征，那么理论上来说，模型特征具有2^d种组合，多以可以看到，随着维度d增大，选出最优的特征子集，是一个NP-hard问题，即不存在多项式时间复杂度的算法得到全局最优解。退而求其次，目前我们很多启发式的算法，在之前我们介绍了一种正则化方法，称为ℓ_1正则化，这种正则化方法不仅可以使得模型去过拟合，而且还能得到参数的稀疏解，即某些参数为0。如果说参数是一个列向量的话，那些为0的参数对应的特征，就是可以去掉的特征，我们把ℓ_1正则化方法称为**嵌入式方法**（Embedding method），更新规则采用近端梯度下降，详细内容参看教科书第十一章的内容。还有一类算法是直接对特征子集进行选择，例如**前向搜索法**（forward search）：

1. 首先初始化集合$\mathcal{F} = \emptyset$
2. 做如下的循环：
   (a) 对$i = 1, \ldots, d$做循环，如果$i \notin \mathcal{F}$，那么$\mathcal{F}_i = \mathcal{F} \cup \{i\}$，然后使用交叉验证方法，评估特征子集$\mathcal{F}_i$，即得到该子集对应的泛化误差。
   (b) 把之前特征子集对应着的最小泛化误差对应的$\mathcal{F}$，设置为最优特征子集。
3. 选择并输出泛化误差最小的特征集合作为最优子集。

这种特征选择方法称为**包裹式特征选择方法**（wrapper model feature selection）。与前向搜索类似的是后向搜索，即从整个特征集合里，一个一个的去掉，然后评估特征子集对应的泛化误差，这里我们就不在详细介绍了，参考前向搜索方法。

最后还有一类特征选择的方法，称为**过滤式方法**（filtering method）。所谓过滤式方法，不在依赖所选择的模型，直接从数据入手，观察数据特征与数据标签之前的关系（这里单指监督学习，因为无监督学习不存在标签的概念），最常用的一个指标，是**互信息**（mutual information）：

$$MI(x_i, y) = \sum_{x_i \in \{0,1\}} \sum_{y \in \{0,1\}} p(x_i, y) \log \frac{p(x_i, y)}{p(x_i)p(y)} \tag{1}$$

可以看到，这里给出的是针对特征是离散变量情况而言的，对于连续变量，同样也有互信息的计算公式，只需把求和改成积分即可，对于$p(x_i, y), p(x_i), p(y)$都可以通过训练集计算得到。在之前我们已经提到过信息熵的概念，其实互信息也可以看成类似的信息熵，历史上称为**Kullback-Leibler（KL）Divergence**：

$$MI(x_i, y) = KL(p(x_i, y)||p(x_i)p(y)) \tag{2}$$

KL散度用来衡量两种分布的相似程度，取值范围为$[0, +\infty)$，取0时的含义是对数里面的比值是1，即两者分布完全相同。如果两个分布之间差异巨大，那么KL散度的值会趋近无穷。KL散度在我们后面讨论变分自动编码器时，还会遇到，届时会详细讨论信息熵，相对熵，KL散度等相关概念。

最后我们需要指出，当你得到了每个特征与标签之间的KL 散度得分，那么应该选择多少特征呢？一种标准的方法，还是通过交叉验证方法，在对应的训练模型下，挑选出泛化误差最低的最优子集模型。

4 数据建模

这里所谓的数据建模，就是对取得的数据，根据任务的不同，选择合适的模型。除非你对数据非常的熟悉，知道什么模型最适合数据，一般而言，我们会从最简单的模型出发，比如分类任务，可以先选择Logistic回归，如果分类效果不好，要对不好的结果进行分析，是数据线性不可分造成的，还是数据不平衡造成的（其中一类数据非常少），还是数据特征太少造成的等等。下面我们会继续围绕这些问题，给出相应的解决措施。下面我们主要从特征与模型之间的结合，介绍模型选择相关的内容。

5 数据集划分与模型选择

数据建模中，供给我们选择的模型有很多，那么如何选择出一个效果好的模型是关键。除此之外，还存在一个问题，就是数据集的不同，对同一个模型来说，也会影响它最终得到的结果，所以数据集的划分是很自然的。由于数据集的划分是随机的，在保证划分比例确定的情况下，不同的顺序，会得到不同的模型。一般情况下，我们固定顺序，然后把数据集分为k份，把$k-1$份数据集作为训练集，留下一份数据集作为验证集，这样就会训练出k个模型，这k个模型对应着不同的精度（这里可以理解为准确性），这种训练模式，称为k-折交叉验证。把k 个模型对应的精度求平均，得到交叉验证的精度，但是最后在使用时，一般选择精度最高的模型。我们这里简单介绍一下这种交叉验证的由来与细节，因为在一些竞赛或者模型提升时，是很重要的手段。

假定我们试图选择不同的模型，去学习一问题。简单起见，我们有一个有限的模型集合$\mathcal{M} = \{M_1, \ldots, M_d\}$[1]，这$d$ 个模型，可能包含SVM，神经网络，logistic 回归等等。给定一个训练集S，让这些模型去进行训练，最后得到训练误差，我们选择最小的训练误差对应的模型即可。流程分为两步：

1. 在训练集S上训练每个模型M_i，得到每个模型对应的假设h_i；
2. 选择泛化误差最小的假设作为最优模型。

[1]对于无限个模型，也并不困难。

其实上面的算法并不可靠。试想模型集合里面有多项式回归，那么高阶的多项式回归一定会更小的训练误差，所以上面的算法一定会选出高方差的模型[2]。

所以要对数据集进行划分，提出所谓的**交叉验证**（cross validation）。一般情况下，分为三部分：**训练集**（Training Set），**验证集**（Validation Set），**测试集**（Test Set）。训练集为帮助我们训练模型，简单的说就是通过训练集的数据让我们确定拟合曲线的参数。验证集也叫做**开发集**（Dev Set），用来做**模型选择**（model selection），即做模型的最终优化及确定的，用来辅助我们的模型的构建；测试集是为了测试已经训练好的模型的精确度。三者根据数据量的大小，具有不同的划分规则，在小数据集当中，一般三者比例为：$60\%, 20\%, 20\%$，或者$70\%, 10\%, 20\%$ （顺序为训练集，验证集，测试集）。在数据集的量级在十万，百万以上级别时，例如人脸识别这类任务中，一般划分为$98\%, 1\%, 1\%$。总结一下，我们采用如下的方法去选择模型：

1. 随机把集合S进行划分，根据数据集的量和任务本身，划分合适的比例，分为S_{train}, S_{cv}（这里我们不在划分S_{test}，因为此时交叉验证集只使用一次）；
2. 在训练集上S_{train}上训练每个模型M_i，得到每个模型的假设函数h_i；
3. 把每个模型在验证集上测试，得到每个模型对应的$\hat{\epsilon}_{cv}(h_i)$，选择最小的误差对应的模型作为最优模型。

当选择了最优模型后，其实还有一分部数据没有使用，就是交叉验证集，所以最后一步，会把所有的数据集放在一起，在此进行训练，我们会获得一个比之前更好的模型。这样的做法已经比之前的不划分数据集，效果好多了，但是再往下思考一步，我们会发现，其实整个数据集还是没有完全利用上，假设划分比例为$7:3$，我们还是只在70%的数据集上训练，剩下的30%虽然在最用添加到最优的模型训练上，但是不排除其他的模型因为添加剩下的这部分数据，效果反超当前的最优模型。所以为了充分利用所有的数据，我们提出k折交叉验证，方法如下：

1. 随机划分数据集为k份，假设数据集包含样本数m个，那么每份数据集的个数为m/k，如果除不尽，把多出的均匀的分给部分数据集即可。
2. 对每一个模型M_i做循环，具体如下：For $j = 1, ..., k$

   在数据集$S_1 \cup \cdots S_{j-1} \cup \S_{j+1} \cup \cdots S_k$训练模型$M_i$ （仅去掉一份数据集），得到每个模型对应的假设h_{ij}，然后把得到的假设h_{ij}在S_j上进行测试，得到$\hat{\epsilon_{S_j}}(h_{ij})$。对每个模型来说，会有$k$个假设对应的误差，然后对这$k$个误差求平均，得到$\hat{\epsilon}(h_i)$。
3. 选择具有最低的平均误差作为最优模型，很明显，这个最优模型是基于整个数据集得到的。

一般说来，这里的k一般取10，如果太大了，计算量太大，成本太高，太小的话，体现不出来这种方法的优势。有一种极端的取法，就是有多少样本，就分成多少份，即$k = m$，这种方法称为**留一交叉验证法**（leave-one-out cross validation）。

最后我们需要强调一下，虽然我们介绍了完善的方法，但是对于刚开始测试一个新的模型时，我们可以选择简单的方法取进行评估模型，比如说使用数据集划分后，分为3 份，训练集，验证集和测试集，使用训练集训练，使用验证作为泛化误差的结果，然后通过这个结果，分析误差原因，进而改进模型，直到泛化误差不在降低，在使用测试集给出最后的泛化误差，作为最后的结果。

[2]下一节，我们会解释什么是方差与偏差

6 模型评价

目前我们学习了机器学习处理的两类任务，一类是回归，一类是分类。当一个模型训练完成，在训练集上得到的结果，称为**训练误差**（Training Error），在测试集上会得到一个结果，称为**测试误差**（Testing error）。我们希望我们最后训练得到的模型具有泛化能力，即在新的数据上，也能得到不错的效果，所以经验误差对我们来说，是一个非常关键的评价指标。

对于回归任务的评价，是直接从损失函数中体现出来的，损失函数越小，说明模型最后的结果越好，对于分类任务而言，有多种多样的评价指标。

6.1 分类评价指标

我们这里以二分类为例，对于多分类问题，同样可以把二分类对应的指标加以推广，我们这里不在进行讨论。我们看如下的表格：

预测值 / 实际值	正例	反例
正例	TP	FN
反例	FP	TN

我们对上面四个英文的简写进行一个解释，其中TP 为真阳例（True Positive）：预测为真，实际为真；TN 为真阴例（True Negative）：预测为假，实际为假；FP 为假阳例（False Positive）：预测为真，实际为假；FN 为假阴例（False Negative）：预测为假，实际为真。

根据表格里面的四个值，可以组合得到一些综合评价指标，第一个是精确率（Precision），也叫查准率：

$$\text{Precision} = \frac{TP}{TP + FP} \tag{3}$$

分子是阳例的准确预测，分母是预测的全是阳例的个数，整体比值越大，说明预测阳例的准确率越高，比值越小，说明很多实际是阴例，但是预测错误，当成阳例了，意味着查准率太低。第二个是还有**查全率**，也称**召回率**（Recall）：

$$\text{Recall} = \frac{TP}{TP + FN} \tag{4}$$

这里我们解释一下，分子是预测正确的阳例，分母是所有实际是正例的值，比值越大说明查到阳例的准确性高，比值越小，说明很多是阳例的样本漏查了，所以顾名思义，叫查全率。在分类任务中，查准率与查全率是相互制约的，很明显，对于一个固定的模型，要想查的准，要求就得高，所以大概率会把一些正确的也分到错误里面，就降低了查全率，反之亦然。所以提出了两者的调和平均数，称为**F得分**（F-score）：

$$\text{F-score} = \frac{2Precision \cdot Recall}{Precision + Recall} \tag{5}$$

这个分数对应着两者之间的权衡。一般在很多竞赛当中，常把F得分当作模型好坏的依据。最后一个，也是最常用的，称为准确率（Accuracy），就是预测正确的样本数，除以总的样本数，即：

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{6}$$

这就是我们常说的精度，即准确率越高，精度越高，模型的效果越好。

我们在这里举一个例子，有100张照片，其中，猫的照片有60张，狗的照片是40张。输入这100张照片进行二分类识别，找出这100张照片中的所有的猫，混淆矩阵的结果如下表：

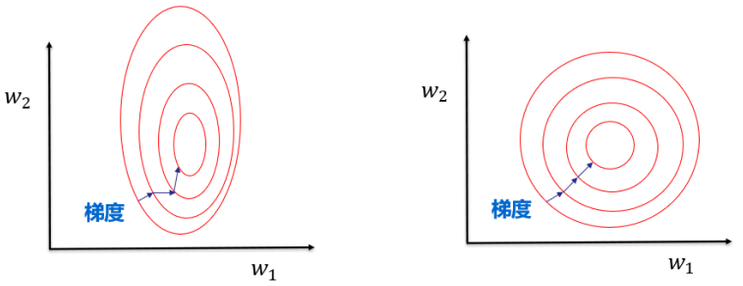

Figure 1: 标准化前后对应的梯度下降示意图。

预测值 / 实际值	正例	反例
正例	TP=40	FN=20
反例	FP=10	TN=30

正确率为：$70/100 = 0.7$，查准率为：$40/50 = 0.8$，查全率为：$40/60 = 0.67$。在一些领域，还有ROC曲线和PR 曲线用来衡量二分类的好坏，这里我们就不再详细介绍了，作为课后阅读，看教科书上相关的章节。

6.2 偏差与方差

在训练模型时，数据的特征，往往是具有不同量纲的，导致有的数量级非常大，比如10^5左右，而有的数量级很小，可能10^2左右，导致一个样本，不同的特征差异很大，所以在训练之前，要对数据进行标准化，标准化首先可以提升模型精度：不同维度之间的特征在数值上有一定比较性，可以大大提高分类器的准确性；其次加速模型收敛：最优解的寻优过程明显会变得平缓，更容易正确的收敛到最优解，如图（2）。 标准化有多种形式，我们在这简单介绍两种，一种为：

$$x^\star = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{7}$$

这种方法把特征对应的值全都映射到$[0, 1]$区间，这种做法使得各特征对目标变量的影响一致，会将特征数据进行伸缩变化，所以数据归一化是会改变特征数据分布的。另一种标准化称为Z-Score标准化，为：

$$x^\star = \frac{x - \mu}{\sigma} \tag{8}$$

这里的μ, σ分别为：

$$\mu = \frac{1}{m}\sum_{i=1}^{m} x^{(i)} \tag{9}$$

$$\sigma^2 = \frac{1}{m}\sum_{i=1}^{m} (x^{(i)})^2 \tag{10}$$

处理后的的数据均值为0，方差为1。经过这种标准化变换之后的特征数据分布没有发生改变。一般情况下，当数据特征取值范围或量纲数量级差异较大时，做一下标准化处理。

下面我们介绍机器学习当中非常重要的一项评价，方差与偏差，这对应着模型对数据的拟合程度，如图（2）： 第一个模型是一个线性模型，欠拟合，不能很好地适应我们的训练集；第三个模型过于强调拟合原始数据，而丢失了算法的本质：预测新数据。我们可以看出，若给出一个新的值使之预测，它将表现的很差，是过拟合，虽然能非常好地适应我们的训练集但在新输入变量进行预测时可能会效果不好；而第二个模型似乎最合适。下面我们针对欠拟合的情况，给出一些解决办法：

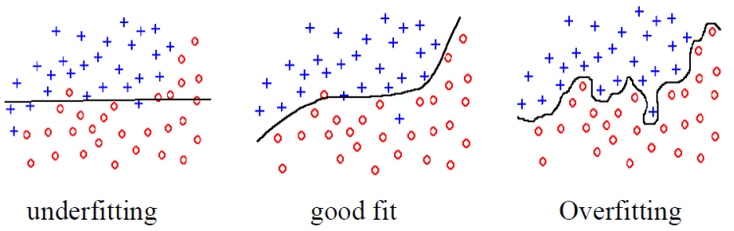

Figure 2: 过拟合与欠拟合。

- 添加新特征：当特征不足或者现有特征与样本标签的相关性不强时，模型容易出现欠拟合。通过挖掘组合特征等新的特征，往往能够取得更好的效果。
- 增加模型复杂度：简单模型的学习能力较差，通过增加模型的复杂度可以使模型拥有更强的拟合能力。例如，在线性模型中添加高次项，在神经网络模型中增加网络层数或神经元个数等。
- 减小正则化系数：正则化是用来防止过拟合的，但当模型出现欠拟合现象时，则需要有针对性地减小正则化系数。

面对过拟合的情况，有如下的一些解决方案：

- 获得更多的训练数据：使用更多的训练数据是解决过拟合问题最有效的手段，因为更多的样本能够让模型学习到更多更有效的特征，减小噪声的影响。
- 降维：即丢弃一些不能帮助我们正确预测的特征。可以是手工选择保留哪些特征，或者使用一些模型选择的算法来帮忙（例如PCA）。
- 正则化：正则化(regularization)的技术，保留所有的特征，但是减少参数的大小（magnitude），它可以改善或者减少过拟合问题。
- 集成学习方法：集成学习是把多个模型集成在一起，来降低单一模型的过拟合风险。
- 早停：即在模型正在训练时，还没有收敛，提前停止训练。
- Dropout：在神经网络训练时，如果神经网络里神经元过多，也会造成过拟合，Dropout方法是在训练时，随机扔掉一些神经元，每次迭代时，扔掉不同的神经元，这样增加了模型的稳定性。
- 数据增广：所谓增广，就是原始数据不够时，又不想直接去收集数据，此时利用人工或者一些生成模型，创造更多新的数据。比如图片，可以通过镜像，伸缩，调节对比度等手段，一张图片变成多张图片，但是仍旧表达的时一个意思。后面我们会介绍两个生成模型：变分自动编码器与生成对抗网络，用来生成数据。

但是我们强调，数据量越大，模型效果越好，数据量的多少，决定了模型的上限，所以诞生了机器学习届的名言：成功的机器学习应用不是拥有最好的算法，而是拥有最多的数据！

对于一个训练好的模型，过拟合和欠拟合用数学语言表达，就是方差与偏差之间的关系。这里指的方差：描述的是预测值的变化范围，离散程度，也就是离其期望值的距离。方差越大，数据的分布越分散；偏差为：描述的是预测值（估计值）的期望与真实值之间的差距。偏差越大，越偏离真实数据。如图（3）所示： 下面我们以回归问题为例，通过误差的期望，去观察偏差与方差相互之间的关系。假定训练集为D，测试集数据的每个数据点为x，要对每个数据预测对应的目标值$y \in \mathbb{R}$，是真实的标记，

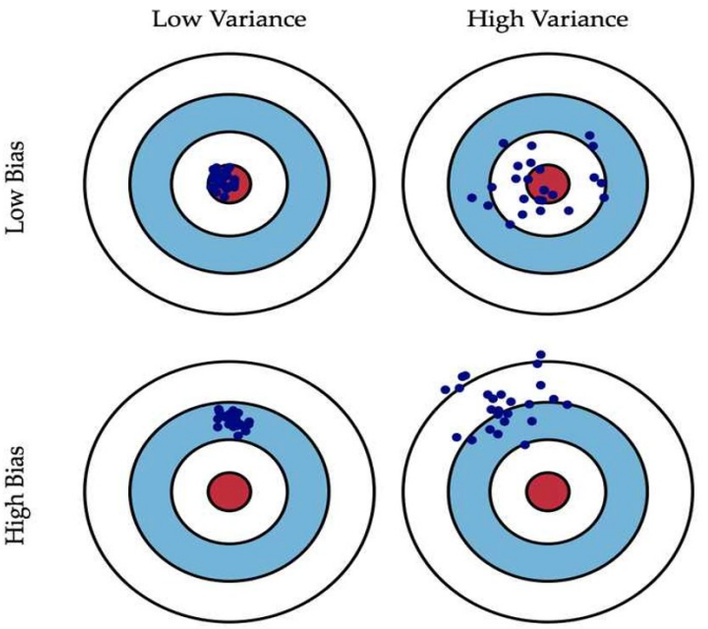

Figure 3: 方差与偏差。

在数据集中可能会有噪声，把在数据集中的标记记为y_D，把$f(x;D)$记为训练集D上学的模型f在x上的预测输出。那么均方误差（mean squared error）为：

$$E(f;D) = E_D[|f(x;D) - y_D|^2] \tag{11}$$

当你计算得到的MSE很大时，会有如下三种情况：

1. 过拟合：模型太接近训练样本集，以至于在测试集上并没有很好的效果，泛化能力不强。
2. 欠拟合：模型没有从训练集获得足够有用的信息，并没有抓住特征x与目标y之间的关系。
3. 数据噪音很大，模型既不欠拟合，也不过拟合。只是数据集中异常点很多，这些异常点可能对应着更重要的信息。

我们简单起见，认为回归任务中，所有的数据集符合相似的分布，即便如此，模型在不同的训练集上学到的参数很可能不同。所以在不同训练集上的模型期望预测为：

$$\hat{f}(x) = E_D[f(x;D)] \tag{12}$$

使用样本数相同的不同训练集产生的方差为：

$$Var(x) = E_D[\big(f(x;D) - \hat{f}(x)\big)^2] \tag{13}$$

噪声为：

$$\epsilon^2 = E_D[(y_D - y)^2] \tag{14}$$

期望输出与真实标记的差别称为偏差，即：

$$bias^2(x) = (\hat{f}(x) - y)^2 \tag{15}$$

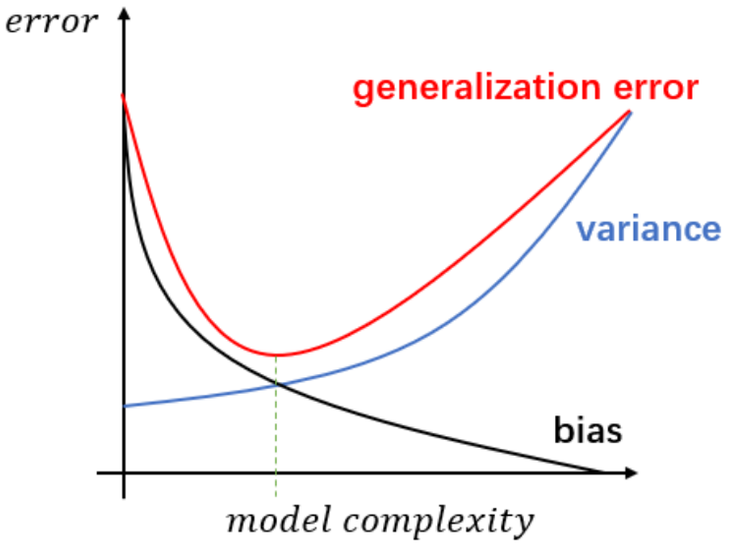

Figure 4: 方差与偏差之间的相互制衡。

噪声我们一般认为符合均值为0，方差为σ^2的高斯分布$N(0,\sigma^2)$，所以$E_D[y_D - y] = 0$。那么我们就可以对方程（11）进行拆解：

$$
\begin{aligned}
E(f;D) &= E_D[(f(x;D) - y_D)^2] && (16)\\
&= E_D[(f(x;D) - \hat{f}(x) + \hat{f}(x) - y_D)^2] && (17)\\
&= E_D[(f(x;D) - \hat{f}(x))^2] + E_D[(\hat{f}(x) - y_D)^2] + E_D[2(f(x;D) - \hat{f}(x))(\hat{f}(x) - y_D)] && (18)\\
&= E_D[(f(x;D) - \hat{f}(x))^2] + E_D[(\hat{f}(x) - y_D)^2] && (19)\\
&= E_D[(f(x;D) - \hat{f}(x))^2] + E_D[(\hat{f}(x) - y + y - y_D)^2] && (20)\\
&= E_D[(f(x;D) - \hat{f}(x))^2] + E_D[(\hat{f}(x) - y)^2] + E_D[(y - y_D)^2] + 2E_D[(\hat{f}(x) - y)(y - y_D)] && (21)\\
&= E_D[(f(x;D) - \hat{f}(x))^2] + (\hat{f}(x) - y)^2 + E_D[(y - y_D)^2] && (22)
\end{aligned}
$$

于是：

$$E(f;D) = bias^2(x) + var(x) + \epsilon^2 \quad (23)$$

换句话说，泛化误差可分解为偏差，方差与噪音之和。偏差度量了模型的期望预测与真是结果的偏离程度，刻画了学习算法本身的拟合能力；方差度量了同样大小的训练集的变动所导致的模型性能的不稳定，刻画了数据扰动所造成的影响；噪音表达了学习任务上任何模型所能达到的期望泛化误差的下届，客刻画了学习问题本身的难度。一般来说，偏差与方差是存在冲突的。如图（4）所示。给定学习任务，假定我们能控制学习模型的训练程度或者模型复杂度，在建模过于简单或者模型训练不足时，模型的拟合能力不够强，训练数据的扰动不足以时模型产生显著变化，此时偏差主导了泛化错误率；随着训练程度加深或者换更复杂的模型，模型的拟合能力逐渐增强，训练数据发生的扰动渐渐能被模型学到，方差逐渐主导了泛化错误率；在训练度充足后，模型的拟合能力非常强，训练数据的轻微扰动都会导致模型发生显著的变化，此时认为模型出现了过拟合现象。

那么通过训练集与交叉验证集时可以判断是否是过拟合或者欠拟合的，训练集误差和交叉验证集误差近似时：偏差/欠拟合；交叉验证集误差远大于训练集误差时：方差/过拟合。这里所对应的偏差与方差出现过高的问题，也与上面提到的方法类似，用来解决这些问题：

- 获得更多的训练实例——解决高方差。

- 尝试减少特征的数量——解决高方差。
- 尝试获得更多的特征——解决高偏差。
- 尝试增加多项式特征——解决高偏差。
- 尝试减少正则化程度 λ ——解决高偏差。
- 尝试增加正则化程度 λ ——解决高方差。

References