Learning Theory续[1]

简介

在本章中，我们将了解如何选择模型、如何筛选特征以及如何避免过拟合。

1 交叉验证

假定一个包含有限模型的集合$\mathcal{M} = \{M_1, \cdots, M_d\}$（这里面可能有SVM、神经网络、逻辑回归……），我们需要从中挑选出一个最优模型。给定数据S（大小等于m），最朴素的选择方法可能是

1. 在S上训练每个模型M_i，得到假设h_i
2. 输出训练误差最小的假设

我们在数据集上训练一个模型，通过ERM总能够找到该模型最优假设的估计。虽在note4中已证明，当数据足够大时，这种估计的误差确是有上限。不过，这在假设类VC维不等于无穷的情况下才满足，若不限制VC维（随着数据的增加任由其增长），那么选择出来的假设必定维数很高，泛化能力很弱，并非最优。（注：这里的假设类是指无穷假设类，无穷假设类的VC维不一定无穷。）因此，提出新的模型验证方法，并输出一个泛化误差小的假设是必须的。

1.1 简单交叉验证

1. 将S随机分为训练集S_{train}和验证集$S_{\text{validation}}$（通常按照7:3的比例分配）
2. 在S_{train}上训练每个模型M_i，并得到假设h_i
3. 输出在$S_{\text{validation}}$上泛化误差最小的假设

简单交叉验证的缺点在于浪费了一部分数据，即使我们选择泛化误差最小的模型在S重新训练假设，我们仍然只是在训练集上挑选模型。尤其在数据很少时，这个缺点会被放大。

1.2 K折交叉验证

1. 将S随机均分成k（通常取10）个子集，它们依次为$S_1, \cdots, S_k$
2. 对于每个模型M_i，我们都以以下流程评价之
    - For $j = 1, \cdots, k$：在$M_1 \cup \cdots \cup S_{j-1} \cup S_{j+1} \cup \cdots S_k$上训练模型$M_i$，得到假设$h_{ij}$，并在$S_j$测试假设$h_{ij}$，得到误差$\hat{\epsilon}_{S_j}(h_{ij})$
    - 取k个误差$\hat{\epsilon}_{S_j}(h_{ij})$的平均值作为$M_i$泛化误差的估计
3. 对泛化误差估计最小的模型，输出其在S上训练所得的假设

在k折交叉验证中，我们对数据的利用是很充分的，不过计算成本显然比简单交叉验证昂贵得多。当我们取k=m时，即每次只留下一条数据作为验证集，我们称这样特殊的k折交叉验证为**留一交叉验证**。

2 特征选择

假定在一个监督学习问题中，特征的数目n(n远大于m)很大，现在如果我们采用简单的线性分类器模型，显然我们的假设类的VC维是$O(n)$，存在过拟合的隐患（不满足notes4中定理4.1的推论）。因此，从特征集中筛选出与学习任务有关联的子集是必须的。

2.1 向前搜索

1. 初始化特征集$\mathcal{F}=\phi$（空集合）
2. 重复以下流程，直至$|\mathcal{F}|$达到某一阀值k
    - For $i=1,\cdots,n$ if $i \notin \mathcal{F}$, let $\mathcal{F}_i=\mathcal{F} \cup \{i\}$，利用交叉验证评价$\mathcal{F}_i$（即只在特征集$\mathcal{F}_i$上训练算法，并估计其泛化误差）
    - 设置$\mathcal{F}$为以上最佳特征集
3. 输出$\mathcal{F}$

向前搜索的流程是：从原始特征集中选择一个最佳的特征放入$\mathcal{F}$，然后在剩余的特征集上重复此过程直至选出top k特征。向前搜索是warppper model feature selection的一个例子，与之类似的有**向后搜索**，这种搜索的过程是首先初始化特征集$\mathcal{F}=\{1,\cdots,m\}$，然后不断的从中删除特征直至得到想要的特征子集。warpper model feature selection的筛选效果很好，但是计算昂贵，对于大小为n的原始特征集，计算复杂度为$O(n^2)$。

2.2 过滤特征选择（filter feature selection）

过滤特征选择的基本思路是：首先在训练数据上计算特征x_i对类标签y的分数$S(i)$（表达了特征x_i在标签y判断上提供有用信息的多少），然后简单的选择分数最高的k个特征即可。我们用来计算分数$S(i)$的一种方法是衡量特征与标签之间的关联程度，称为互信息（mutual information），公式如下

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

（注：这里假定特征和标签都是二取值的，在更一般的情况下，累加的是所有的取值。）其中，概率$p(x_i,y),p(x_i)$和$p(y)$可以由他们在训练集上的经验分布估计。互信息也会经常表达为Kullback-Leibler(KL)差

$$MI(x_i, y)=KL\left(p(x_i, y) \| p(x_i) p(y)\right)$$

可以看到当x_i与y相互独立时，特征不能为标签提供任何信息，$p(x_i,y)=p(x_i)p(y)$，互信息等于0；当x_i能够为y的判断提供信息越多，$p(x_i,y)$越大（$p(x_i)p(y)$不变），互信息也越大。最后一个问题，如何选择参数k？事实上，我们可以在所有可能的k值所对应的特征集上应用交叉验证，并从中找出最佳参数k。

3 贝叶斯统计正则化

我们在参数估计的时候常用的一个方法是最大似然估计（maximum likelihood, ML），过程为

$$\theta_{ML}=\operatorname{argmax}_\theta \prod_{i=1}^m p\left(y^{(i)} | x^{(i)} ; \theta\right)$$

最大似然估计实际上体现了频率学派的观点，即认为θ是一个未知的常量（因此概率没有写成$p\left(y^{(i)}|x^{(i)},\theta\right)$）。而贝叶斯学派持有不同的观点，即认为$\theta$是一个随机变量，它的取值是以先验概率$p(\theta)$存在的。给定训练集$S=\left\{\left(x^{(i)},y^{(i)}\right)\right\}_{i=1}^{m}$，我们可以得到后验概率

$$\begin{aligned} p(\theta|S) &= \frac{p(S|\theta)p(\theta)}{p(S)} \\ &= \frac{\left(\prod_{i=1}^{m} p\left(y^{(i)}|x^{(i)},\theta\right)\right)p(\theta)}{\int_{\theta}\left(\prod_{i=1}^{m} p\left(y^{(i)}|x^{(i)},\theta\right)\right)p(\theta)d\theta} \end{aligned}$$

在这里概率$p\left(y^{(i)}|x^{(i)},\theta\right)$与选择的模型有关，比如在贝叶斯逻辑回归模型中，$p\left(y^{(i)}|x^{(i)},\theta\right)=h_{\theta}\left(x^{(i)}\right)^{y^{(i)}}\left(1-h_{\theta}\left(x^{(i)}\right)\right)^{1-y^{(i)}}$，其中$h_{\theta}\left(x^{(i)}\right)=1/\left(1+\exp\left(-\theta^{T}x^{(i)}\right)\right)$。利用$\theta$的先验概率，我们可求出给定$S$和新样本$x$的后验概率

$$p(y|x,S)=\int_{\theta}p(y|x,\theta)p(\theta|S)d\theta$$

与前面先求出θ再进行预测完全不同，这里直接对θ积分，得到使上式最大的y即为分类类别。理想很美好，现实却很残酷——计算积分是一项很困难的事情，所以我们通常做法是找到使$p(\theta|S)$最大的θ，在后面做预测时就不需要积分了。这个过程也称为最大后验估计（maximum a posteriori, MAP），过程为

$$\theta_{MAP}=\operatorname{argmin}_{\theta}\prod_{i=1}^{m}p\left(y^{(i)}|x^{(i)},\theta\right)p(\theta)$$

与最大似然估计相比，最大后验估计在表达式末尾多乘了一项，通常有$\theta\sim\mathcal{N}(0,\tau^{2}I)$。事实上，最大后验估计比最大似然估计能够更好地克服过拟合，这是由于最大似然估计只是最大化$p(y|x;\theta)$，很容易造成θ过于复杂（即θ中0项很少），方差过大；然而最大后验估计考虑了$p(y|x,\theta)$和$p(\theta)$两者，相当于综合权衡了偏差和方差。（注：另一个看待这个问题的思路是，最大似然估计实际上是在进行$\min\,(y-h_{\theta}(x))^{2}$的过程，然而最大后验估计是在进行$\min\left((y-h_{\theta}(x))^{2}+\lambda\|\theta\|^{2}\right)$。）

---

1. Writen by Jimmy on 2016/03/08.↩