《机器学习 2》课程报告 8-9 讲

小组成员： 赵敬业、苏煜家、杨鸿谦、熊子骁

2022 年 6 月 13 日

目录

1 第八讲字典学习

1.1 回归分析

回归分析：是一种统计学上分析数据的方法，目的在于了解两个或多个变量间是否相关、相关方向与强度，并建立数学模型以便观察特定变量来预测研究者感兴趣的变量。

线性方法解的形式：$f(x) = w_0 + w_1x^{(1)} + w_2x^{(2)} + \ddots + w_px^{(p)}$

线性方法优缺点：

- 优点
  - 易建模
  - 易解释
  - 易计算
- 缺点
  - 精度一般
  - 抗干扰能力差
  - 先验依赖强

线性方法的三要素:

- 假设空间: 线性关系假设。也是线性模型的一个关键问题。这里可以嵌入线性模型的很多先验知识。比如音乐数据 MIT 团队嵌入了一些变量重要、一些不重要的先验信息取得了较好的结果；但是实际上很多自变量对因变量的影响是非线性的，如销量和价格的关系。
- 优化策略: 最小化残差
- 学习算法: 这多了去了：OLS 方法、所有能想到的算法

1.2 广义可加模型 (GAM)

公式:

$$f(x) = \mathrm{w}_0 + \mathrm{w}_1 g_1\left(x^{(1)}\right) + \mathrm{w}_2 g_2\left(x^{(2)}\right) + \ldots + w_p g_p\left(x^{(p)}\right)$$

其中，用 g 刻画各个变量的影响。

至于实施步骤，广义可加模型具有以下步骤

a) 选择假设空间。假设空间可以选择任何函数，但是一般选择光滑函数，并且可以由此嵌入先验信息，比如
   - 样条基函数
   - 回归树
   - 多项式

b) 选择优化策略。优化策略选择范围比较广泛。如

- OLS
- RLS
- LASSO
- 其他正则化模型

c) 选择算法。可以选择包括

- 解析表达
- 梯度下降

自变量无关性假设

1.3 字典学习

核心思想：无论人类的知识有多么浩繁，也无论人类的科技有多么发达，一本新华字典或牛津字典足以表达人类从古至今乃至未来的所有知识，那些知识只不过是字典中字的排列组合罢了！

广义可加模型只是对一个变量进行了操作。而字典学习是对所有的变量进行某种组合，如公式

$$f(x) = \mathrm{w}_1 h_1(x) + \mathrm{w}_2 h_2(x) + ... + w_N h_N(x)$$

优缺点：

- 优点。字典学习实质上是对于庞大数据集的一种降维表示，或者说是信息的压缩。正如同字是句子最质朴的特征一样，字典学习总是尝试学习蕴藏在样本背后最质朴的特征。
- 缺点。解释性一般，计算量大，需要一定的数学基础

步骤

a) 假设空间选择。选择一簇函数近似学习目标。可编码。

   (a) 基的选择：多项式；回归树；核函数；神经网络。

   (b) N 的选择：理论上越大越好，算法上适当大即可。

b) 优化策略选择。可求解是优化策略必须的能力，可解释是应该争取的，解释性比较强的模型对管理者更为友好。

   (a) 目标函数。目标函数依然包括经验风险最小化。

   (b) 约束条件。即为结构风险极小化，具体可以包括二范数约束、一范数约束、核范数约束等。

c) 算法选择。求解所建立的数学模型。有效性、快速性。其中正则化参数的作用是使方程有惟一解，控制解的结构、提高泛化能力等。

(a) 二范数。可以使用解析表达式 (其实也可以用下降方法，反正是凸的)。问题定义：

$$\operatorname{argmin} \frac{1}{m} \sum_{i=1}^{m} \left(f\left(x_i\right)-y_i\right)^2$$
$$\text{s.t.} \sum_{j=1}^{N} \left|w_j\right|^2 \leq t$$

其中

$$Y=\begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{pmatrix} \quad H=\begin{pmatrix} h_1\left(x_1\right) & h_2\left(x_1\right) & h_3\left(x_1\right) & \cdots & h_N\left(x_1\right) \\ h_1\left(x_2\right) & h_2\left(x_2\right) & h_3\left(x_2\right) & \cdots & h_N\left(x_2\right) \\ \vdots & \vdots & \vdots & & \vdots \\ h_1\left(x_m\right) & h_2\left(x_m\right) & h_3\left(x_m\right) & \cdots & h_N\left(x_m\right) \end{pmatrix} \quad W=\begin{pmatrix} \mathscr{W}_0 \\ \mathscr{W}_1 \\ \mathscr{W}_2 \\ \vdots \\ \mathscr{W}_N \end{pmatrix}$$

解析解：

$$W_{2,\lambda}=\left(\mathrm{H}'H+\lambda I\right)^{-1} X^{\mathrm{T}} Y$$

(b) 一范数。可以使用梯度下降等方法。具体算法如下：

$$\rho_j=\frac{1}{m} \sum_{i=1}^{m} h_j(x)\left(y_j-\sum_{k!=j} w_k h_k\left(x_i\right)\right)$$

$$\hat{w_j}=\begin{cases} \left(\rho_j+\lambda/2\right)/z_j, & \rho_j>\lambda/2 \\ 0, & \rho_j \in(-\lambda/2, \lambda/2) \\ \left(\rho_j-\lambda/2\right)/z_j, & \rho_j<-\lambda/2 \end{cases}$$

(c) 零范数。可以使用贪婪算法，计算复杂度 $O((k^*)^3)$。步骤如下：

i. 初始化：$f_0(x)=0, r_0(x)=\sum_{i=1}^{m}(f_0(x_i)-y_i)^2$
ii. 贪婪选基 $h_k^*=\arg\min|\sum_{i=1}^{m} h(x_i) r_{k-1}(x_i)|$
iii. 贪婪选权重 $w_k^*=\arg\min\sum_{i=1}^{m}(w_1 h_1^*(x_1)+\ddots+w_k h_k^*(x_i)-y_i)$
iv. 更新残差 $f_k(x)=(w_1 h_1^*(x_1)+\ddots+w_k h_k^*(x_i), r_0(x)=\sum_{i=1}^{m}(f_k(x_i)-y_i)^2$
v. 更新 k, 回到第二步或停机

其中需要说明的是稀疏性选择具有比较大的意义。比如对于一个新的知识，生疏者会有更多的神经元被激活，熟练者神经元会有更少被激活；雷达只有在稀疏时才能有意义。

2 第九讲神经网络

2.1 神经网络的原理

核方法是以再生核希尔伯特空间为假设空间，以结构风险极小化作为优化策略，以梯度型方法作为学习算法的一种方法；字典学习以冗余字典作为假设空间。这两种方法都是在足够大 (核方法是恰好足够大) 的空间中进行学习，获得一个用来逼近数据的低维空间。

从假设空间上来看，神经网络的假设空间与前两种提到的方法不一样，神经网络的假设空间是无限维的，基底具体长什么样子由训练产生；并且神经网络非线性非凸，而前两种方

法最后都是线性模型。

神经网络可以从仿生的角度进行解释。生物神经网络包括细胞体、树突、突触和轴突等部分，分别对应神经元、内权、阈值和外权。

$$\sum_{k=1}^{n} a_k \sigma(w_k x + \theta_k)$$

如上面公式所述，其中 a_k 代表外权；函数 σ 代表激活函数，模拟了生物的兴奋和抑制状态；w_k 即为内权；θ 即为阈值，模拟了突触延时和不应期。

2.2 神经网络的分类

按照连接方式分，可以将神经网络分成前馈网络、反馈网络和图神经网络。前馈网络一般用来做预测，反馈网络一般用来做自然语言处理，图神经网络最近比较火热，但具体能做什么还不清楚，笔者了解到的有使用这种方法预测推特用户是否是水军的。

按照隐层数量分，可以将神经网络分为浅层和深度神经网络。这里将一层的神经网络定义为浅层，多层的定义为多层，~~以满足发表论文、对外忽悠人的需要~~, 本节中只讨论浅层神经网络，作为对照，深度神经网络有很多优点以及不同之处，留在之后讨论。

按照隐层数和连接方式分类，可以分为全连接神经网络和卷积神经网络、梳妆神经网络等。不同的神经网络结构可以用来嵌入不同的先验知识。

2.3 神经网络的三要素

2.3.1 假设空间

$$H_n = \left\{ f(x) = \sum_{k=1}^{n} a_k \sigma(w_k x + \theta_k) \right\}$$

在假设空间中，需要确定神经元个数和激活函数。其中激活函数主要有

- Sigmoid 函数。所有从 0 突变为 1 的光滑函数都是 Sigmoid 函数，著名的是逻辑函数 $f(x) = \frac{1}{1+e^{-x}}$，这种激活函数主要问题是在求梯度的过程中容易出现问题，当函数值离远点较远，则梯度非常小，容易导致梯度消失，以及出现饱和问题。
- tanh 函数。$tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$, 这种函数和 Sigmoid 没有本质区别，只是函数值域变为 (-1,1)
- Relu 函数。$Relu = max(0, x)$ 克服了前面梯度消失的问题，因为当存在梯度的时候，梯度一定是 1，但是不光滑导致求导困难，而且一旦进入了负数区域导致神经元被抑制就永远被抑制。
- ELU 函数。有人尝试寻找能好处占尽的激活函数而创造了 ELU，正数部分和 Relu 一样，保证了梯度不消失，在非正数区域换成了 $\alpha(e^x - 1)$, 尽量避免了梯度消失、神经元永不激活的问题，但因为计算量大导致应用较少~~和笔者建立的复杂的模型一样拉跨~~。

2.3.2 优化策略

优化策略包括损失函数选择和罚函数选择。值得一提的是十年前有一位高人非常擅长调神经网络，就是巧妙利用了罚函数，笔者也听说有人试图用 *Fisher* 信息熵作为罚函数，不知道效果如何。

2.3.3 学习算法

神经网络假设空间和优化策略都是可圈可点的地方，而学习算法是硬伤 (好像所有稍微复杂一点的、非凸非光滑的模型都这样????)。学习算法和优化策略不适配，是神经网络没有解释性的根本原因。学习算法选择一般选择梯度型的算法，比如误转逆传播算法。

2.4 浅层神经网络的优缺点

2.4.1 优点

维数无关 [1]。神经网络的逼近性质和维数没有关系，可避免维数祸根问题。

线性方法能做的，神经网络一定能做。[2](我看不懂，老师让我引用的)。

2.4.2 缺点

饱和性问题 [3]。浅层神经网络面临者饱和性的问题。

稀疏性问题 [4]。浅层神经网络面临着稀疏性问题，即无法模拟稀疏问题。

旋转不变性。浅层神经网络在旋转不变性的模拟方面不能有很好的效率，导致在地震预测等方面效果不好。

在其他一些结构上，浅层神经网络表现也不好，如一些特殊的结构、流形问题等。

2.5 训练方法

神经网络的训练包括两个层面的参数，即显参数和隐参数。按照这种分类方法，可以同时训练显参数和因参数，即为单场景训练；分别训练显参数和因参数，即为双场景训练。

2.5.1 双场景训练

内权是较为典型的隐参数，这里记为 T。可以通过 ADMM 算法进行训练得到，也可以通过随机赋值、还可以直接强行嵌入领域知识。

外权直接通过线性的方法进行训练即可。

3 第八讲勘误

考虑到老师平时工作繁忙，本人 (赵敬业) 也因考研等原因没有时间亲自叩门拜访老师，特在此处将整理过程中 PPT 第八讲可能出现的错误罗列至此，第九讲因为笔者水平原因没有找到问题。出现错误还望老师指正。

如图1，删除线划掉的” 假设空间” 应为” 优化策略”，下方删除线的”LASSO” 应为” 坐标下降法等梯度算法”

如图2贪婪算法部分，应该把图中三处划红线的”n” 改为”m”, 因为这里是对于样本进行的加和，不是对维度，本页中也已经展示出来样本是最多 m 个。

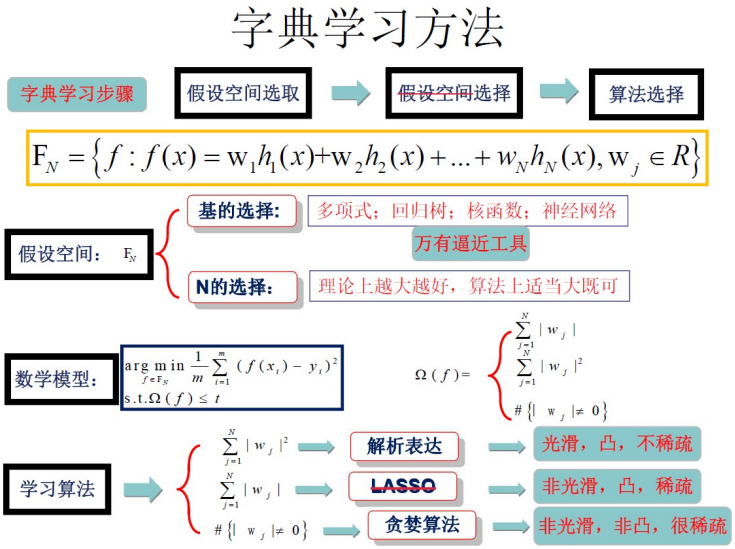
字典学习方法
字典学习步骤
假设空间选取
假设空间选择
算法选择
$F_N=\{f:f(x)=w_1h_1(x)+w_2h_2(x)+\ldots+w_Nh_N(x),w_j\in R\}$
假设空间：
基的选择：
多项式；回归树；核函数；神经网络
万有逼近工具
N的选择：
理论上越大越好，算法上适当大既可
数学模型：
$\arg\min_{f\in F_N}\frac{1}{m}\sum_{i=1}^{m}(f(x_i)-y_i)^2$
$s.t.\Omega(f)\le t$
$\Omega(f)=\sum_{j=1}^{N}|w_j|$
$\sum_{j=1}^{N}|w_j|^2$
$\#\{|w_j|\neq 0\}$
学习算法
$\sum_{j=1}^{N}|w_j|^2$
解析表达
光滑，凸，不稀疏
$\sum_{j=1}^{N}|w_j|$
LASSO
非光滑，凸，稀疏
$\#\{|w_j|\neq 0\}$
贪婪算法
非光滑，非凸，很稀疏

图 1:

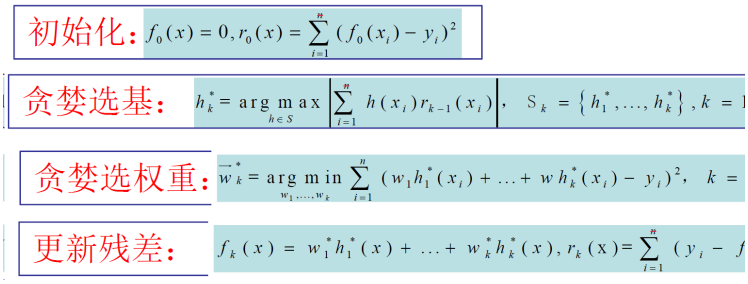
初始化：$f_0(x)=0,r_0(x)=\sum_{i=1}^{n}(f_0(x_i)-y_i)^2$
贪婪选基：$h_k^*=\arg\max_{h\in S}\left|\sum_{i=1}^{n}h(x_i)r_{k-1}(x_i)\right|,\ S_k=\{h_1^*,\ldots,h_k^*\},k=$
贪婪选权重：$w_k^*=\arg\min_{w_1,\ldots,w_k}\sum_{i=1}^{n}(w_1h_1^*(x_i)+\ldots+wh_k^*(x_i)-y_i)^2,\ k=$
更新残差：$f_k(x)=w_1^*h_1^*(x)+\ldots+w_k^*h_k^*(x),r_k(x)=\sum_{i=1}^{n}(y_i-f$

图 2:

参考文献

[1] Barron, A. R. Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Transactions on Information theory* **39**, 930–945 (1993).

[2] Mhaskar, H. N. Neural networks for optimal approximation of smooth and analytic functions. *Neural computation* **8**, 164–177 (1996).

[3] Debao, C. Degree of approximation by superpositions of a sigmoidal function. *Approximation Theory and its Applications* **9**, 17–28 (1993).

[4] Chui, C. K., Li, X. & Mhaskar, H. N. Neural networks for localized approximation. *mathematics of computation* **63**, 607–623 (1994).