机器学习：监督学习(Supervised Learning)之回归(Linear, NonLinear, Ridge, Lasso)

~~Copyright: Jingmin Wei, Automation - Pattern Recognition and Intelligent System, School of Artificial Intelligence and Automation, Huazhong University of Science and Technology~~

Copyright: Jingmin Wei, Computer Science - Artificial Intelligence, Department of Computer Science, Viterbi School of Engineering, University of Southern California

---

---

1. 线性回归($Linear\ Regression$)的求解方法

概述：考虑监督学习问题，定义数据集 $D=\{(x_i,y_i)|\ i=1,\cdots,m\}$ 。其中 $x_i\in R^{1\times n}$ 为样本，y_i 为标签。

假设模型的权重为 w ，偏置为 ε ，线性回归的前向模型为：

$$y_i=w^Tx_i+\varepsilon_i,\quad \varepsilon_i\sim N(0,\sigma^2)$$

根据监督学习的思想，我们想让预测值 w^Tx_i 和标签值 y_i 之间的差距尽可能小，因此该优化问题为：

$$J(w) \triangleq \sum_{i=1}^{m}(y_i - w^T x_i)^2$$

$$\arg\min_w J(w)$$

那么我们需要解决两个数学问题，$J(w)$ 从何而来？如何运用优化算法求解使 $J(w)$ 最小时 w 的取值？

1.1. 均方误差的推导过程(最大似然估计)

根据最大似然估计，推导上面的均方误差损失 $J(w)$ 。

首先讲述分布变换中的一个重要结论：假设随机变量 z 满足标准正态分布 $N(0,1)$ ，则随机变量 $x = \mu + \sigma z$ 满足正态分布 $N(\mu, \sigma^2)$ 。

ε_i 是先验知识，通过高斯分布得到的。

$$y = w^T x_i + \varepsilon_i, \quad \varepsilon_i \sim N(0, \sigma^2)$$

$$P(y_i|x_i, w) \sim N(y_i|w^T x_i, \sigma^2), \quad \text{即服从高斯分布}$$

同时假设样本是独立同分布的 (i,i,d) 。

根据Lesson 3.5 参数估计(MLE, MAP, Bayes, KNN, Parzen, GMM, EM算法)的内容，可以定义似然函数：

$$\begin{aligned} L(w) &\triangleq \ln P(D|w) \\ &= \ln \prod_{i=1}^{m} P(D|w) \\ &= \sum_{i=1}^{m} \ln P(y_i|x_i, w) \\ &= \sum_{i=1}^{m} \ln\Big[(\frac{1}{2\pi\sigma^2})^{\frac{1}{2}} \exp\big(-\frac{1}{2\sigma^2}(y_i - w^T x_i)^2\big)\Big] \\ &= \sum_{i=1}^{m} \ln\Big(\sqrt{\frac{1}{2\pi\sigma^2}}\Big) - \frac{1}{2\sigma^2}\sum_{i=1}^{m}(y_i - w^T x_i)^2 \\ &= -\frac{1}{2\sigma^2}||y - Xw||_2^2 + C \end{aligned}$$

根据第Lesson 3.5 参数估计(MLE, MAP, Bayes, KNN, Parzen, GMM, EM算法)的最大似然估计(MLE)，目标等价为：

$$\begin{aligned} w^* &= \max L(w) \Leftrightarrow \min||y - Xw||_2^2 \\ &= \arg\max_w \log P(D|w) \end{aligned}$$

因此根据最大似然估计法：即在假设样本的分布满足高斯概率密度函数，且样本满足独立同分布的情况下，线性回归的目标函数等价于最小化拟合误差的平方和。

$$J(w) = ||y - Xw||_2^2$$

这就是线性回归中，均方误差的来源，之后可以通过使用广义逆直接求解导数为 0 的点，或者使用随机梯度下降，牛顿法等求解该优化问题。

1.2. 均方误差的随机梯度下降优化

求解当 $J(w)$ 取最小时，w 的取值。

第一个思路：转化为**一阶优化问题**求解 $\min J(w)$ (梯度下降)：

$$J(w) \triangleq \sum_{i=1}^{m}(y_i - w^T x_i)^2$$

$$w_{t+1} = w_t + \eta \cdot \nabla\{J(w_t)\} = w_t + \eta \cdot 2\sum_{i=1}^{m}(w_t^T x_i - y_i)x_i$$

根据上式不断循环，直到两次 w 之间的差 $w_{t+1} - w_t$可以忽略不计，或者这次 $J(w_t) = 0$，或者达到了迭代最大次数。这是梯度下降的基本优化思路，我们这里将其转为随机梯度下降算法。

对数据加载器 $dataloader$ 做定义，假设 $dataloder$ 里有 n 个数据，每次循环都会 $shuffle$ (数据打乱)。数据以 $batch$ 的格式存在，每一个 $batch$ 中有 $batch\_size$ 个数据，总共有 $batch\_number$ 个 $batch$，那么可以得到：

$$batch\_size \times batch\_number = n$$

根据Lesson 3 优化方法基础的内容，算法是批处理的 $mini - batch\ SGD$，在第二层循环中，每次计算一个随机 $batch$ 的局部梯度均值，则随机梯度下降算法过程表示如下：

$$\begin{aligned}
&while(until\ \nabla J(w) = 0 \vee (w_{t+1} - w_t < 1e-3) \vee \text{迭代次数到 } n):\\
&\quad for\ batch\ in\ dataloader(shuffle):\\
&\qquad \nabla J(w) = \frac{2}{batch_size}\sum_{i=1}^{batch_size}(w_t^T x_i - y_i)x_i\\
&\qquad w_{t+1} = w_t - \eta \cdot \nabla J(w)
\end{aligned}$$

理解这个梯度下降的过程很重要！之后所有算法中的随机梯度下降法将沿用这一框架。

1.3. 广义逆(最小二乘法推导)

第二个思路：广义逆。

目标是求向量二范数最小，即每个元素的平方和最小：

$$\min J(w) = ||y - Xw||_2^2$$

首先令：

$$y \triangleq \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix} \quad X \triangleq \begin{bmatrix} --x_1^T-- \\ \vdots \\ --x_m^T-- \end{bmatrix}$$

1.3.1 一些有用的矩阵引理

$$||A||_2^2 = A^T A$$
$$(AB)^T = B^T A^T$$
$$a^T b = b^T a$$
$$\frac{\partial\ x^T A x}{\partial\ x} = (A + A^T)x$$
$$\frac{\partial\ w^T x}{\partial\ x} = w$$

根据上述引理，则：($\because \frac{\partial w^T (X^T y)}{\partial w} = \frac{\partial y^T X w}{\partial w} = X^T y$)

$$\begin{aligned} \therefore\ \frac{\partial J(w)}{\partial w} &= \frac{\partial (y - Xw)^T (y - Xw)}{\partial w} \\ &= \frac{\partial w^T X^T X w - w^T X^T y - y^T X w}{\partial w} \\ &= 2X^T X w - X^T y - X^T y \\ &= 2X^T (Xw - y) \end{aligned}$$

目标是 $\nabla J(w) = 0 \Leftrightarrow X^T X w^* = X^T y$ 。

所以可求得 w 的计算公式：

$$w^* = (X^T X)^{-1} X^T y$$

拓展： 如何判断 c 是否可逆？(**可通过 L_2 正则化项解决不可逆情况**)

$$X \in R^{m \times n} \quad m < n,\ m \geqslant n$$
$$X^T X \in R^{n \times n}$$

> n 阶矩阵 A 可逆，则 $|A| \neq 0$ ；$r(A) = n$(满秩)；列向量 $a_1, \cdots a_n$ 线性无关；对应的齐次方程组只有 0 解；A 的 n 个特征值非 0 。
>
> n 阶矩阵 A 不可逆，则 $|A| = 0$ ；$r(A) < n$；列向量 $a_1, \cdots a_n$ 线性相关；对应的齐次方程组有非 0 解。

1.4. 最大后验估计，岭回归($Ridge\ Regression - L_2$)

针对广义逆不可逆的问题，可以采用 L_2 正则化的思想来解决。

根据[Lesson 3.5 参数估计(KNN, Parzen窗, GMM, EM算法)]的内容，即在最大似然的基础上，引入先验(约束条件) $P(w) = \prod_{i=1}^{n} P(w_i)$ ，化为最大后验估计(MAP)问题。

把 w_i 看成是随机变量，服从某一分布。$\forall i, j$ ，假设 w_i 和 w_j 是独立的。

根据贝叶斯公式：

$$P(D)P(w|D) = P(w)P(D|w)$$

根据最大后验概率的目标，可忽略贝叶斯公式中的分母，即定义：

$$\begin{aligned} w^* &= \arg\max_w \log P(w|D) \\ &\propto \arg\max_w \log P(D|w) + \log P(w) \\ P(w|D) &\sim P(w)P(D|w) \end{aligned}$$

针对岭回归，可以引入先验：

$$P(w) = \prod_{i=1}^{n} N(w_i|0,\ \tau^2)$$

则目标函数变为：

$$\begin{aligned} J(w) &= \frac{1}{2\sigma^2}\sum_{i=1}^{n}(y_i - w^T x_i)^2 + \tau^2 \sum_{i=1}^{m} w_i^2 \\ &= ||y - Xw||_2^2 + \tau^2 ||w||_2^2, \quad (\text{二范数}) \end{aligned}$$

$$\begin{aligned} w^*_{LR} &= \arg\max_w ||y - Xw||_2^2 \\ w^*_{ridge} &= \arg\max_w J(w) \end{aligned}$$

为了求最大的 $J(w)$ ，依旧对 w 求导，且类比 w^*_{LR} 线性回归：

$$w^*_{LS} = (X^T X)^{-1} X^T y$$

$$\begin{aligned} \because ||X||_2^2 &= X^T X = X^T I X \\ w^*_{ridge} &= (X^T X + \tau I)^{-1} X^T y \end{aligned}$$

这样对于岭回归，广义逆算法就一定可逆。使用梯度下降求解该优化问题也是可行的。

根据画图可得，实际上是一系列过原点同心圆(等值线) $||w||_2^2 = C$，即$x_1^2 + x_2^2 = C$ ，与一系列同心椭圆(等值线) $||w||_1 = C$ 的切线。$Ridge$ 方法对应的约束域是圆，其切点只会存在于圆周上，不会与坐标轴相切，则在任一维度上的取值都不为 0 ，因此没有稀疏。

1.5. 最大后验估计，拉索回归($Lasso\ Regression - L_1$)

拉索回归和岭回归都是引入了最大后验估计的思想，二者的区别在于，引入的先验不同。

拉索回归引入的先验 $P(w)$ 如下：

$$P(w) = \prod_{i=1}^{m} Lap(w_j|0,\ \frac{1}{\lambda})$$

$$\text{其中，} Lap(w_j|\mu,\ b) = \frac{1}{2b}\exp(-\frac{|x-\mu|}{b})$$

目标函数变为：

$$J(w)=\frac{1}{2\sigma^2}\sum_{i=1}^{n}(y_i-w^Tx_i)^2+\tau\sum_{i=1}^{m}|w_i|$$

无法通过求导来求解优化问题，因为目标函数带绝对值，但是可以通过梯度下降来求解。

根据画图也可得，实际上是一系列过原点 $|x_1|+|x_2|=C$ 的菱形, 即 $|x_1|+|x_2|=C$ ，与一系列同心椭圆(等值线) $||w||_1=C$ 的切线，且切点刚好在坐标轴上。

因为其约束域是正方形，会存在与坐标轴的切点，使得部分维度特征权重为 0 ，因此很容易产生稀疏的结果。

2. 回归问题

输出为连续的值，而不是离散的类别。

2.1. 一元线性回归

$$y=\theta_1x+\theta_2$$

令 $w=\theta_1(weight)$ ， $b=\theta_2(bias)$ 。

模型：$h(x)=wx+b$

优化参数：w,b

代价函数(损失函数)：$J(w,b)=\frac{1}{2m}\sum\limits_{i=1}^{m}(h(x_i)-y_i)^2$

目标：$minimize\ J(w,b)$

优化方式之一：梯度下降

$$w_i=w_i-\alpha\frac{\partial}{\partial w}J(w,b)$$

$$b_i=b_i-\alpha\frac{\partial}{\partial b}J(w,b)$$

因此，$w*,b*$ 通过优化算法结出。给定 x_{m+1} ，预测 $\hat{y}_{m+1}=h(x_{m+1},w*,b*)$ 。

2.2. 多元线性回归

增加了特征数量，总共有 n 个。

模型：$y:h(x)=\sum_{i=1}^{n}w_ix_i+b$

优化方式之一：梯度下降：

$$w_i=w_i-\alpha\frac{\partial}{\partial w_i}J(w,b)$$

$$b=b-\alpha\frac{\partial}{\partial b}J(w,b)$$

也可以简化成：

$$W = W - \alpha \frac{\partial}{\partial W} J(w, b)$$

2.3. 多项式回归

模型：$y : h(x) = w_1 x^1 + w_2 x^2 + b = w_1 x^1 + w_2 (x^1)^2 + b$

也可以选择构造其他的特征：

$$h(x) = w_1 x^1 + w_2 \sqrt{x^1} + w_3 (x^1)^2 + w_4 \sin(x^1) + b$$

$$\begin{array}{ccc} 1 & 2 & y \\ x_1^{(1)} & x_1^{(2)} & y_1 \\ \vdots & \vdots & \vdots \\ x_m^{(1)} & x_m^{(2)} & y_m \end{array} h(w_1, w_2, b) = w_1 x^{(1)}$$

可通过核函数升维：

$$\begin{array}{cccccl} 1 & 2 & 3 & 4 & 5 & h(w_1, w_2, \ldots, w_5, b) \\ x_i^{(1)} & x_i^{(2)} & x_i^{(1)} x_i^{(2)} & \left(x_i^{(1)}\right)^2 & \left(x_i^{(2)}\right)^2 & = w_1 x^{(1)} + w_2 x^{(2)} + w_3 x^{(3)} + w_4 x^{(4)} + w_5 x^{(5)} + b \end{array}$$

2.4. 特征缩放(最大最小标准化和归一标准化)

比例调节，将数据的特征缩放到 $[0, 1]$ 或 $[-1, 1]$ 之间。

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$

x' 是缩放后的特征数据。

标准化，让每个特征的值都有零均值和单位方差。

$$x' = \frac{x - \overline{x}}{\sigma}$$

在分布变换，中心极限定理，以及神经网络的 Batch Normalization 等算法中也有类似应用。

3. 过拟合与欠拟合，模型评估

3.1. 过拟合与欠拟合

考虑散点的欠拟合和过拟合问题。

信号和噪声："信号"是数据中真正想要学习到的信息。"噪声"则是数据集中的不相关的信息和不确定性。好的机器学习模型应该提高信噪比。

拟合优度：模型预测值与真实值相匹配的程度。学习“噪声”的模型被称为是过拟合，在训练集上表现良好，但是与训练集的拟合优度差。欠拟合是对已有训练集的拟合程度就差，模型表现效果差，没有学习到数据中的信息。

欠拟合曲线偏差较大，方差较小，过拟合的曲线偏差较小，方差较大。

3.2. 模型评估(偏差 / 方差 / 残差)

期望预测为：$\overline{f}(x) = E[f(x;D)]$ 。

期望泛化误差：表示为三个不同误差总和：偏差(bias)、方差(variance)、残差(irreducible error)。

$$\begin{aligned} f(x) &= w^T x, \quad w\text{服从正态分布} \\ D &= [x_i, y_i]_{i=1}^n \end{aligned}$$

$$\begin{aligned} E(f;D) &= E[(y - f(x;D))^2] \\ &= E[(f(x;D) - \overline{f}(x) + \overline{f}(x) - y_D)^2] \\ &= E[(f(x;D) - \overline{f}(x))] + E[(\overline{f}(x) - y_D)^2] \\ &= E[(f(x;D) - \overline{f}(x))^2] + E[(\overline{f}(x) - y + y - y_D)] \\ &= E[(f(x;D) - \overline{f}(x))^2] + E[(\overline{f}(x) - y)^2] + E[(y - y_D)^2] \\ &= var(x) + bias^2(x) + \varepsilon^2(x) \end{aligned}$$

偏差(bias)：期望输出与真实标记的差别，又错误的模型假设造成的，模型呈现欠拟合的状态。

方差(variance)：度量了同样大小的训练集变动所导致的学习性能的变化，即刻画了数据扰动造成的影响，模型呈现过拟合的状态。

噪声(残差 irreducible error)：数据本身存在的误差导致的学习困难。

$$\text{偏差}(bias) : bias = [\overline{f}(x) - y]^2$$
$$\text{方差}(variance) : var(x) = E_D[f(x \cdot D) - \overline{f}(x)^2]$$

理想情况下，应该在过拟合和欠拟合中做出权衡选择一个模型，使得模型在训练集上的表现量化同时也能准确对没有出现过的数据进行预测。

增加模型的复杂度会增加预测结果的方差同时减小偏差，相反减小模型复杂度会增加偏差、减小反差，这就是为什么被称为偏差和方差的权衡。

泛化能力(generalization)：对未出现的数据进行预测的能力被称为模型的泛化能力。

4. 预防

防止欠拟合：

> 选取或构造性的特征。
>
> 增加模型复杂度。
>
> 使用集成的方法。
>
> 增加模型训练时间。

监测过拟合：初始数据集分成单独的训练集和验证集，该方法可以近似我们的模型在新数据上的表现。

训练过程：训练集较小时训练误差远远小于验证误差，模型完全过拟合。训练集增大时，训练误差越来越接近验证误差，这时模型拟合效果较好。

防止过拟合：

1. 增加数据量。
2. 合理的数据切分。使用合理的比例切分训练集，验证集和测试集。
3. 正则化方法。即在损失函数上添加对训练参数的惩罚范数，对需要训练的参数进行约束。常用的参数有 l_1 和 l_2 范数(Ridge Regression / Lasso Regression)。
4. 神经网络中可以使用 Dropout 。引入 Dropout 层，随机丢掉一些神经元，即让某几个神经元，以一定的概率 p 停止工作，减轻网络的过拟合现象。
5. 提前停止。当损失不再减小，或者精度不再增加时可以停止训练，但可能会导致参数训练不充分。
6. k 折交叉验证选择训练参数。
7. 通过Lesson 7 信息论与决策树中基于互信息计算的特征选择，删除部分相关度高的特征。

4.1. k 折交叉验证

将数据划分为 k 个子集，称之为折叠，然后迭代的用其中的 $k-1$ 个子集用于模型训练，同时将剩余的子集用于作为验证集验证模型。

算法过程：

- 将训练集为 k 个大小相等的子集。
- 选择 $k-1$ 子集上训练模型。
- 用剩余的 1 个子集评估评估模型。
- 重复前面两步，每次选择不同的子集作为验证，总共 10 次训练验证。
- 综合模型在 10 次中的表现作为模型参数的评价指标。
- 评估参数。

4.2. 提前停止

模型会逐渐对训练集过拟合，泛化能力随着减弱，在模型泛化能力减弱的时候停止对模型的训练。

算法过程：

- 将数据及为训练集和验证集。
- 在训练集上进行训练，在验证集上获得验证结果(比如每 5 次迭代验证一下模型)。随着训练深入，如果在验证集上发现验证误差开始上升，则停止训练。
- 将停止后的权重作为网络的最终参数。

4.3. 正则化

前文有提到过这是线性回归算法的变种之一，此处只列写公式而不详细说明最大后验的推导过程。

$$
\begin{aligned}
J(w) &= \arg\min_w [L(w) + \lambda P(w)] \\
&= \arg\min_w [L(w) + |w|_1 + |w|_2] \text{目标函数} = \text{损失函数} + \text{正则化项} \\
&= \arg\min_w [L(w) + ||w||_2^2
\end{aligned}
$$

l_1 正则化，拉索回归； l_2 正则化，岭回归。

4.4. l_2 正则化(岭回归)

l_2 正则化需要正则化系数的平方作为惩罚项：

$$J(w,b) = \frac{1}{2m}\sum_{i=1}^{m}(h(x_i)-y_i)^2 + \lambda\sum_{i=1}^{n} w_i^2$$

目标函数：

$$J(w,b) = (w^T X^T - Y^T)(Xw - Y) + \lambda w^T w$$

MAP **角度推导**：参考前文岭回归。

梯度下降角度优化：

$$H(w,b) = \frac{1}{2m}\sum(h(x_i)-y)^2$$

多元线性回归的梯度下降：

$$w_i = w_i - \alpha[\frac{\partial}{\partial w_i}H(w,b)]$$

含 l_2 正则化的多元线性回归的梯度下降：

$$w_i = w_i - \alpha[\frac{\partial}{\partial w_i}H(w,b) + \frac{\lambda}{m}w_i]$$

一般不对 b 做正则化处理。

在 l_2 正则化中 λ 是正则化系数，控制对于权重的惩罚。λ 越大，权重越接近于 0，使得拟合曲线更平滑从而减少过拟合。

直接求优化问题：

$$\begin{aligned}
w &= \arg\min_w J(w) \\
J(w) &= w^T X^T X w - 2w^T X^T Y + Y^T Y + \lambda w^T w \\
&= w^T (X^T X + \lambda I) w - 2w^T X^T Y \\
\frac{\partial J(w)}{\partial w} &= 2(X^T X + \lambda I)w - 2X^T Y = 0
\end{aligned}$$

则：

$$(X^T X + \lambda I)w = X^T Y$$

加入 l_2 惩罚项的 $\hat{w}$：$\hat{w} = (X^T X + \lambda I)^{-1} X^T Y$，**这样使得 $X^T X$ 一定可逆**！

对比线性回归：$\hat{w} = (X^T X)^{-1} X^T Y$。

4.5. l_1 正则化($Lasso$ 回归)

l_1 惩罚项为系数的绝对值。

$$J(w,b) = \frac{1}{2m}\sum_{i=1}^{m}(h(x_i) - y_i)^2 + \lambda\sum_{i=1}^{n}|w_i|$$

$$\text{惩罚项} = \lambda\sum_{i=1}^{n}|w_i|$$

它是估计稀疏线性模型的方法，由于它倾向于具有少量参数值的情况，对于给定解决方案是相关情况下，有效减少了变量数量。因此， $Lasso$ 及其变种是压缩感知(压缩采样)的基础。

与 l_2 不同在于，l_1 倾向于使得最不重要的特征的权重接近 0 。换言之，$LASSO$ 回归可以自动实现特征选择，并且使得模型更加稀疏，即很少非零的特征权重。

$$l_1 : w_1 = w_i - \frac{\lambda}{n} sgn(w), \quad \text{后期梯度下降非常快}$$

$$l_2 : w_2 = (1 - \frac{\lambda}{n})w_i, \quad \text{后期缓慢下降，但不会让模型等于}0\text{，即让模型稀疏化}$$

4.6. 弹性网络

它是权衡 $LASSO$ 回归和岭回归后得出的方法，通过控制混合比率 r 我们可以实现想要的正则化效果。其中，当 $r = 0$ 时，弹性网络为岭回归；当 $r = 1$ 时，弹性网络为 $LASSO$ 回归。一般它比 $LASSO$ 回归更常用一些。

$$J(w,b) = \frac{1}{2m}\sum_{i=1}^{m}(h(x_i) - y_i)^2 + r\lambda\sum_{i=1}^{n}|w_i| + \frac{1-r}{2}\lambda\sum_{i=1}^{n}w_i^2$$

意义：多个特征和另一个特征相关的时候，弹性网络非常好用，拉索回归倾向于随机选择一个，而弹性网络倾向于随机选择两个。