1 线性回归

1.1 单变量线性拟合

1.1.1 简单多项式拟合与误差函数

实际的数据集通常没有明显的规律, 或者内含的规律被随机分布的噪声所隐藏. 对于回归问题, 最简单的方式就是通过多项式进行拟合.

假定训练集为包含 N 组数据的集合 $\{(x_1, y_1), \cdots, (x_N, y_N)\}$, 其中各 x_i 为自变量, 各 y_i 为真实值. 我们考虑形式如下的多项式函数以预测自变量 x 对应的值 (其中 $\hat{y}$ 表示 y 的预测值):

$$\hat{y} = f(x; \boldsymbol{w}) = w_0 + w_1 x + \cdots + w_M x^M = \sum_{j=0}^{M} w_j x^j$$

定义 1.1 拟合多项式的阶数 上述拟合多项式中最高次项的次数 M 被称作拟合多项式的**阶数**.

定义 1.2 权重矢量 权重矢量 $\boldsymbol{w}$ 是拟合多项式的参数构成的矢量. 例如, 上述多项式的权重矢量即为

$$\boldsymbol{w} = (w_0, w_1, \cdots, w_M)$$

给定多项式的阶数, 我们能写出很多可能的拟合多项式. 为了得出与数据集相差最小的多项式, 我们需要定义**误差函数**以衡量预测值与实际值之间的差距.

定义 1.3 误差函数 **误差函数 (Error Function)**(或称**代价函数 (Cost Function)**) 是衡量模型预测值 $\hat{y}$ 与实际值 y 之间的差距的函数.

机器学习中常用的误差函数为**平方和误差函数 (Sum-of-Square Error Function)**, 即**二乘**[1]**误差函数**:

$$E(\boldsymbol{w}) = \frac{1}{2} \sum_{i=1}^{N} (\hat{y}_i - y_i)^2 = \frac{1}{2} \sum_{i=1}^{N} (f(x_i; \boldsymbol{w}) - y_i)^2$$

定义 1.4 最小二乘多项式拟合 使用二乘误差函数的多项式拟合称作最小二乘多项式拟合.

容易看出这与统计学中最小二乘法的关系. 显然, 误差函数越小, 模型给出的预测值 $\hat{y}$ 整体而言与目标值 y 越接近.

1.1.2 拟合多项式的求法: 以线性回归为例

线性拟合是最简单的多项式拟合的方法. 使用一次函数模型

$$f(x) = ax + b$$

[1]这里的二乘即平方. 另外, 函数前的系数 1/2 是为了方便计算而采取的取法.

进行拟合, 那么权重矢量 $\boldsymbol{w}=(a,b)$. 我们有如下结论:

定理 1.5 简单线性回归的参数 假定训练集为包含 N 组数据的集合 $\{(x_1,y_1),\cdots,(x_N,y_N)\}$, 采取模型 $f(x)=ax+b$ 进行拟合, 误差函数为二乘误差函数, 那么得到的结果应为

$$a=\frac{\overline{xy}-\bar{x}\bar{y}}{\overline{x^2}-\bar{x}^2} \qquad b=\frac{\overline{x^2}\bar{y}-\bar{x}\cdot\overline{xy}}{\overline{x^2}-\bar{x}^2}$$

其中 $\bar{*}$ 代表基于训练集数据得到的平均值.

证明. 误差函数 $E(\boldsymbol{w})$ 的自变量为 $\boldsymbol{w}$, 具有两个独立的参数 a,b. 为了让 $E(\boldsymbol{w})$ 最小, 我们不妨对 a,b 分别求偏导, 并令偏导数为 0.

$$\begin{cases}\dfrac{\partial E}{\partial a}=\displaystyle\sum_{i=1}^{N}(ax_i+b-y_i)\,x_i=0\\ \dfrac{\partial E}{\partial b}=\displaystyle\sum_{i=1}^{N}(ax_i+b-y_i)=0\end{cases}$$

整理可得

$$\begin{cases}a\displaystyle\sum_{i=1}^{N}x_i^2+b\sum_{i=1}^{N}x_i-\sum_{i=1}^{N}x_iy_i=0\\ a\displaystyle\sum_{i=1}^{N}x_i+bN-\sum_{i=1}^{N}y_i=0\end{cases}$$

解这个方程组即可得到定理中的结论. □

对于一般的简单多项式拟合问题, 都可以用求偏导数的方法进行求解.

1.1.3 过拟合与正则化

我们用一组简单的数据说明过拟合这一现象. 考虑由 $\sin(2\pi x)$ 附加正态分布的噪声所构成的数据, 由下面的程序给出:

```
1 import numpy as np
2 x = np.arange(-0.5, 0.5, 0.1)  # 生成 x 值, 步长 0.1
3 y = np.sin(2 * np.pi * x)
4 y_noisy = y + np.random.normal(loc=0, scale=np.sqrt(0.3), size=x.shape)  # 添加方差为 0.3 的正态分布噪声
```

现在我们把拟合的结果呈现如下:

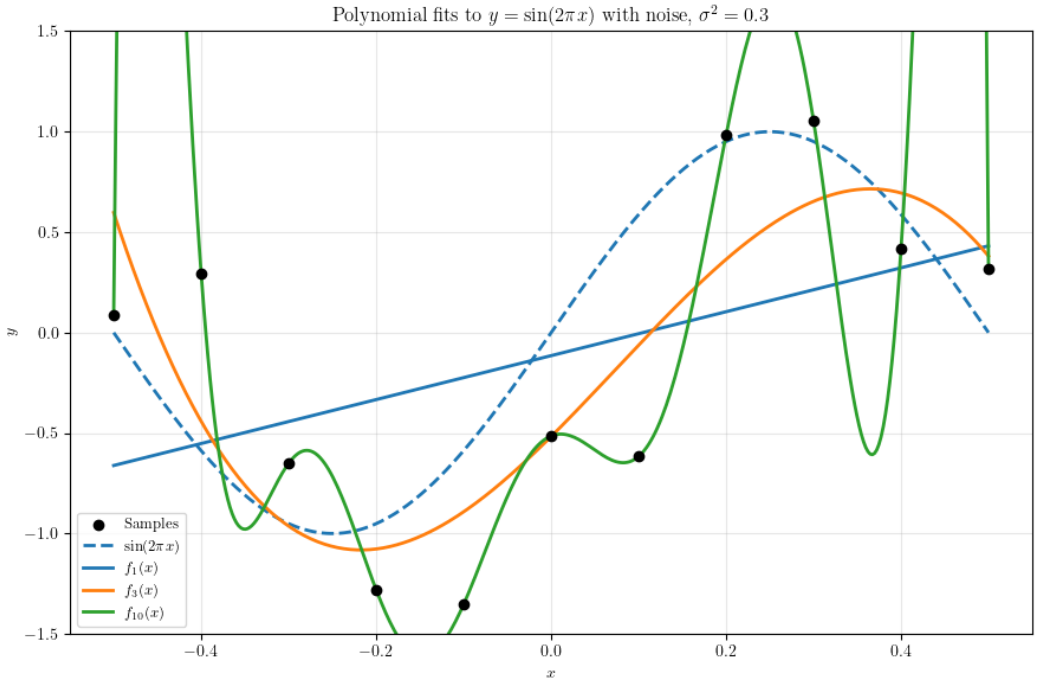

图 1: 不同阶数的多项式拟合

如图所示, 我们用不同阶数的多项式对数据进行拟合. 可以看到, 线性拟合的效果并不好, 而阶数为 3 的多项式拟合则相对而言比较接近实际值. 随着阶数的增加, 拟合效果变得越来越好. 然而, 当阶数过高时 (例如这里采取的阶数为 10), 拟合曲线开始出现剧烈的振荡, 并且在数据点之间的区域偏离了真实函数. 可以想见, 在一般的测试集上, 高阶多项式的拟合效果会变得很差.

定义 1.6 过拟合 当模型在训练数据上表现良好, 但在未见过的数据上表现较差时, 我们称模型发生了**过拟合 (Overfitting)** 现象.

过拟合的原因在于, 复杂的函数具有更强的表达能力, 能够更好地拟合训练数据 (上面的 f_{10} 对每个数据点都没有偏差), 但同时也会受到数据中的噪声影响, 导致模型在训练数据之外的表现变差.

除了在测试集中较差的表现之外, 过拟合的另一个特征是模型的参数值变得非常大. 这和曲线的明显振荡是对应的.

解决过拟合问题的一个简单办法是增加测试集的规模, 或减小模型的复杂度. 另一种常用的方法是**正则化 (Regularization)**.

定义 1.7 正则化 **正则化 (Regularization)** 是通过在误差函数中加入对模型复杂度的惩罚项, 以防止模型过拟合的方法.

例如, 我们可以使用如下的正则化误差函数:

$$\tilde{E}(\boldsymbol{w}) = \frac{1}{2}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2 + \frac{\lambda}{2}||\boldsymbol{w}||^2$$

代替原来的误差函数进行拟合. 其中 λ 为正则化参数, 控制正则化项的权重. 如果 $\lambda = 0$, 那么没有惩罚;λ 越大, 惩罚越强, 此时模型倾向于用更小的参数进行拟合.λ 极大时,$\boldsymbol{w}$ 的各参数接近 0, 是一条接近横轴的平坦的线. 通过调整 λ, 我们可以在拟合训练数据和符合测试数据之间找到一个平衡点.

定义 1.8 岭回归 上述采用二乘误差函数与参数范数的平方作为正则化项的回归方法, 称为**岭回归 (Ridge Regression)**.

同样地, 采取不同的正则化项也可以得到不同的正则化方法.

定义 1.9 Lasso 回归 如果正则化项采用参数范数的绝对值, 即

$$\tilde{E}(\boldsymbol{w}) = \frac{1}{2}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2 + \lambda||\boldsymbol{w}||_1 = \frac{1}{2}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2 + \lambda\sum_{j=0}^{M}|w_j|$$

这种回归方法称为 **Lasso 回归 (Least Absolute Shrinkage and Selection Operator)**.

1.1.4 维度灾难

当输入数据 $\boldsymbol{x}$ 的分量较多时, 我们称其为高维数据. 例如, 一张 28×28 的灰度图像可以看作是一个 784 维的向量.

高维数据的一个重要问题是**维度灾难 (Curse of Dimensionality)**.

定义 1.10 维度灾难 维度灾难是指随着数据维度的增加, 数据量需求呈指数级增长, 从而导致计算和存储的巨大开销, 以及模型难以捕捉数据的真实分布等问题.

前面所举的例子中的 $\boldsymbol{x}$ 是一维的, 这是很容易用多项式拟合的. 但是, 如果类似的方法用到高维数据上就有很大的问题了. 对于一张分辨率为 640×480 的图像, 每个像素采用 RGB 表示, 那么一张图像就可以看作是一个 $640 \times 480 \times 3 = 921600$ 维的向量. 如果仍然采用多项式拟合, 那么需要的各阶参数数量为

$$N_0 = 1 \qquad N_1 = 921600 \qquad N_2 = C_{921600}^2 = 424354112000 \qquad N_3 = C_{921600}^3 = 1.303 \times 10^{17} \cdots$$

很难找到足够的训练数据来拟合这些参数.

简单而言, 维度的增加对数据量的要求是指数增长的, 而数据量本身却是有限的. 这就导致了维度灾难. 此外, 高维空间中的点往往是稀疏分布的; 高维空间的球绝大部分分布于球面附近, 这些反直觉的性质使得模型难以捕捉数据的真实分布.

例如, 采用监督学习的方式让程序分辨猫和狗, 很有可能程序采用与猫和狗无关的特征来进行分类, 例如图像的亮度, 特定像素点的数目等, 从而导致模型在测试集上表现不佳.

1.2 从概率论的角度看待线性回归

1.2.1 正态分布与最小二乘法

通常, 我们假定观测值 y 可以写成下面的形式:

$$y = g(x; \boldsymbol{w}) + \epsilon$$

其中 $g(x; \boldsymbol{w})$ 是一个确定的函数, 显示了数据的内在规律; ϵ 为噪声. 大部分时候, 我们都假设噪声满足均值为 0, 方差为 σ 的正态分布.

定义 1.11 正态分布 随机变量 X 服从均值为 μ, 方差为 σ^2 的**正态分布 (Normal Distribution)**(或称**高斯分布 (Gaussian Distribution)**), 如果它的概率密度函数为

$$p(X = x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \equiv \mathcal{N}\left(x|\mu, \sigma^2\right)$$

这样, 对于给定的自变量 x_i 和观测值 t_i, 其概率为

$$P\left(t_i|x_i\right) = P\left(\varepsilon = t_i - g\left(x_i; \boldsymbol{w}\right)\right) = \mathcal{N}\left(t_i - g\left(x_i; \boldsymbol{w}\right)|0, \sigma^2\right) = \mathcal{N}\left(t_i|g\left(x_i; \boldsymbol{w}\right), \sigma^2\right)$$

假定各组数据是相互独立的, 并且各处的噪声满足同一分布 (即方差 σ^2 在各处相同) 那么整个数据集的概率函数为

$$P\left(\{t_i\}|\{x_i\}\right) = \prod_{i=1}^{N} P\left(t_i|x_i\right) = \prod_{i=1}^{N} \mathcal{N}\left(t_i|g\left(x_i; \boldsymbol{w}\right), \sigma^2\right)$$

在不引起混淆的情况下, 数据集也可以用粗体字母表示, 例如 $\mathbf{t} = \{t_i\}, \mathbf{x} = \{x_i\}$. 另外按机器学习领域的惯例, 记 $\beta = \dfrac{1}{\sigma^2}$, 于是上述概率函数可以写成

$$P\left(\mathbf{t}|\mathbf{x}, \boldsymbol{w}; \beta\right) = \prod_{i=1}^{N} \mathcal{N}\left(t_i|g\left(x_i, \boldsymbol{w}\right), \beta^{-1}\right)$$

在训练过程中, 我们只知道训练集 $\mathbf{x}$ 与 $\mathbf{t}$, 不知道参数 $\boldsymbol{w}$ 与 β. 直观而言[2], 我们希望选择一组参数 $\boldsymbol{w}, \beta$, 使得在这组参数下, 观测到训练集的概率最大. 这就是**极大似然估计 (Maximum Likelihood Estimation)** 的思想.

对于连乘函数的最值估计, 通常先取对数后再考虑. 于是上述概率函数取对数后得到

$$\ln P\left(\mathbf{t}|\mathbf{x}, \boldsymbol{w}, \beta\right) = \sum_{i=1}^{N} \ln \mathcal{N}\left(t_i|g\left(x_i; \boldsymbol{w}\right), \beta^{-1}\right) = -\frac{\beta}{2} \sum_{i=1}^{N} \left(g\left(x_i; \boldsymbol{w}\right) - t_i\right)^2 + \frac{N}{2} \ln \beta - \frac{N}{2} \ln(2\pi)$$

与训练集有关的项即前面的求和项. 回顾二乘误差函数的定义:

$$E(\boldsymbol{w}) = \frac{1}{2} \sum_{i=1}^{N} \left(g(x_i; \boldsymbol{w}) - t_i\right)^2$$

[2]我们将在后面详细讨论贝叶斯公式以完善对这种直观的严谨叙述.

仍然采用偏导等于 **0** 的办法, 即

$$\frac{\partial}{\partial \boldsymbol{w}} \ln P\left(\mathbf{t}|\mathbf{x}, \boldsymbol{w}, \beta\right) = -\beta \frac{\partial E(\boldsymbol{w})}{\partial \boldsymbol{w}} = \mathbf{0}$$

我们就可以得到与最小二乘法相同的结果, 即拟合函数为

$$f\left(x; \boldsymbol{w}_{\mathrm{ML}}\right) = g\left(x; \boldsymbol{w}_{\mathrm{ML}}\right)$$

这里的 $\boldsymbol{w}_{\mathrm{ML}}$ 表示极大似然估计得到的参数, 与前面推导中的一般权重矢量 $\boldsymbol{w}$ 不同.

这说明在噪声服从正态分布的假设下, 最小二乘法实际上是极大似然估计的一种实现方式. 另外, 我们还可以根据前面的推导求出噪声的表达形式:

$$\frac{1}{\beta} = \sigma^2 = \frac{1}{N} \sum_{i=1}^{N} \left(f\left(x_i; \boldsymbol{w}_{\mathrm{ML}}\right) - t_i\right)^2$$

1.2.2 Bayes 定理

定理 1.12 Bayes 定理 设 A, B 为两个事件, 且 $P(B) > 0$, 那么

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

其中 $P(A)$ 与 $P(B)$ 分别为事件 A 与事件 B 的概率, 又称为**先验概率**; $P(A|B)$ 与 $P(B|A)$ 分别为在事件 B 发生的条件下事件 A 发生的概率, 又称为**后验概率**与**似然函数**.

证明. 由条件概率的定义, 我们有

$$P(A|B) = \frac{P(A \cap B)}{P(B)} \qquad P(B|A) = \frac{P(A \cap B)}{P(A)}$$

整理可得 Bayes 定理. □

例题 1.1 如果你的邻居购买了 n 张彩票, 并且其中有 m 张是中奖的, 请估计你购买一张彩票中奖的概率.
注意: 在你没有购买彩票之前, 你对中奖概率一无所知, 因此只能假定中奖概率是均匀分布的, 即它是 $[0, 1]$ 上等概率取的一个数.

解. 由于本题涉及的中奖概率是连续变量, 因此下面的概率均为概率密度函数. 设中奖概率为 x, 那么买 n 张彩票而中 m 张的概率密度函数为

$$P(m, n|x) = C_n^m x^m (1-x)^{n-m}$$

根据 Bayes 定理, 我们有

$$P(x|m, n) = \frac{P(m, n|x)P(x)}{P(m, n)}$$

而

$$P(m, n) = \int_0^1 P(m, n|x)P(x)\mathrm{d}x$$

由于 x 在 $[0,1]$ 上均匀分布, 因此 $P(x)=1$. 于是

$$P(x|m,n)=\frac{P(m,n|x)}{\int_0^1 P(m,n|x)\mathrm{d}x}=\frac{C_n^m x^m(1-x)^{n-m}}{\int_0^1 C_n^m x^m(1-x)^{n-m}\mathrm{d}x}=\frac{x^m(1-x)^{n-m}}{\int_0^1 x^m(1-x)^{n-m}\mathrm{d}x}=\frac{x^m(1-x)^{n-m}}{\mathrm{B}(m+1,n-m+1)}$$

这里的 B 表示 Beta 函数. 对于连续变量 x 而言, 我们最好用期望来表示 x 的估计值. 因此

$$\hat{x}=\int_0^1 xP(x|m,n)\mathrm{d}x=\frac{\int_0^1 x^{m+1}(1-x)^{n-m}\mathrm{d}x}{\mathrm{B}(m+1,n-m+1)}=\frac{\mathrm{B}(m+2,n-m+1)}{\mathrm{B}(m+1,n-m+1)}=\frac{m+1}{n+2}$$

也即, 我们估计买彩票中奖的概率为 $\dfrac{m+1}{n+2}$. 这是符合直觉的, 因为当 $m=0$ 时, 我们也不应估计中奖概率为 0; 同理, 当 $m=n$ 时, 我们也不应估计中奖概率为 1. 买的越多, 估计的结果就越接近 m/n, 也就越准确.

1.2.3 Bayes 定理在机器学习中的应用

现在, 我们把目光重新放回机器学习上. 把 Bayes 定理写成如下形式:

$$p(\boldsymbol{w}|\mathcal{D})=\frac{p(\mathcal{D}|\boldsymbol{w})p(\boldsymbol{w})}{p(\mathcal{D})}$$

这里的 $\mathcal{D}$ 表示训练集, 而 $\boldsymbol{w}$ 表示模型参数.

定义 1.13 先验分布, 后验分布和似然函数 在上述 Bayes 定理中,$p(\boldsymbol{w})$ 称为**先验分布 (Prior Distribution)**, 表示在给定训练集之前对参数 $\boldsymbol{w}$ 的认识; $p(\boldsymbol{w}|\mathcal{D})$ 称为**后验分布 (Posterior Distribution)**, 表示在给定训练集之后对参数 $\boldsymbol{w}$ 的认识; $p(\mathcal{D}|\boldsymbol{w})$ 称为**似然函数 (Likelihood Function)**, 表示在给定参数 $\boldsymbol{w}$ 的条件下观测到训练集的概率.

公式中的 $p(\mathcal{D})$ 是一个归一化常数, 与参数 $\boldsymbol{w}$ 无关, 通常就不去特意考虑. 因此, 为了获取后验分布, 我们只需要计算似然函数与先验分布的乘积.

频率学派的处理方法 频率学派认为 $\boldsymbol{w}$ 是固定不变的, 不随数据变化, 因此拟合的目的是求出最优的 $\boldsymbol{w}$.

最简单的处理方法, 就是假定先验分布是均匀分布, 即对所有可能的参数 $\boldsymbol{w}$ 都一视同仁. 此时, 后验分布与似然函数成正比, 如果我们取概率最大的 $\boldsymbol{w}$ 作为拟合问题的解, 这时的似然函数就最大. 这就是**极大似然估计**.

定义 1.14 极大似然估计 通过选择使似然函数 $p\left(\mathcal{D}|\boldsymbol{w}\right)$ 最大的参数 $\boldsymbol{w}$ 来拟合数据的方式, 称为**极大似然估计 (Maximum Likelihood Estimation)**.

有时, 我们已经对先验分布 $p(\boldsymbol{w})$ 有一定的估计. 综合考虑 $p(\mathcal{D}|\boldsymbol{w})$ 与 $p(\boldsymbol{w})$ 的乘积, 选择使得后验分布最大的参数 $\boldsymbol{w}$ 作为拟合问题的解, 这就是**极大后验估计**.

定义 1.15 极大后验估计 通过选择使后验分布 $p(\boldsymbol{w}|\mathcal{D})$ 最大的参数 $\boldsymbol{w}$ 来拟合数据的方式, 称为**极大后验估计 (Maximum A Posteriori Estimation)**.

下面给出了极大后验估计的一个例子.

例题 1.2 在过拟合一节中, 我们讲到参数越大越容易过拟合的情形, 即认为 $||\boldsymbol{w}||$ 越大越不可能, 这是与 $\mathcal{D}$ 无关的先验知识.
现在, 假定 $\boldsymbol{w}$ 的范数满足正态分布:

$$p(\boldsymbol{w}|\alpha) = \mathcal{N}\left(||\boldsymbol{w}|| \middle| 0, \frac{1}{\alpha}\right) = \left(\frac{\alpha}{2\pi}\right)^{\frac{M+1}{2}} \exp\left(-\frac{\alpha||\boldsymbol{w}||^2}{2}\right)$$

求极大后验估计的结果.

解. 由 Bayes 定理, 我们有

$$p(\boldsymbol{w}|\mathcal{D}) = \frac{p(\mathcal{D}|\boldsymbol{w})p(\boldsymbol{w}|\alpha)}{p(\mathcal{D})}$$

由于 $p(\mathcal{D})$ 与 $\boldsymbol{w}$ 无关, 因此我们只需考虑分子部分. 取对数后得到

$$\ln p(\boldsymbol{w}|\mathcal{D}) = \ln p(\mathcal{D}|\boldsymbol{w}) + \ln p(\boldsymbol{w}|\alpha) + \text{const}$$

假定似然函数仍然为

$$P(\mathcal{D}|\boldsymbol{w}, \beta) = \prod_{i=1}^{N} \mathcal{N}\left(t_i | g(x_i, \boldsymbol{w}), \beta^{-1}\right)$$

忽略与 $\boldsymbol{w}$ 无关的项可得

$$\ln p(\boldsymbol{w}|\mathcal{D}) = -\frac{\beta}{2}\sum_{i=1}^{N}\left(g(x_i;\boldsymbol{w}) - t_i\right)^2 - \frac{\alpha}{2}||\boldsymbol{w}||^2 + \text{const}$$

这与前述正则化误差函数

$$\tilde{E}(\boldsymbol{w}) = \frac{1}{2}\sum_{i=1}^{N}\left(g(x_i;\boldsymbol{w}) - t_i\right)^2 + \frac{\lambda}{2}||\boldsymbol{w}||^2$$

在形式上一致. 事实上, 如果取 $\lambda = \dfrac{\alpha}{\beta}$, 那么极大后验估计与正则化误差函数的结果是一样的.

同样不难想到, 对 $p(\boldsymbol{w})$ 的不同估计对应不同的正则化误差函数.

Bayes 学派的处理方法 Bayes 学派与频率学派观点相左. 他们认为数据才是固定的, 模型的参数则是随机的, 并且服从某种分布[3]. 我们只能对模型的参数有一个最初的估计[4], 这一估计会随着数据的增加而改变, 最终得到比较准确的结论.

[3]例如, 抛一枚硬币, 如果已经抛出了连续 100 次正面, 那么我们就有理由怀疑这枚硬币不是均匀的, 而是正面朝上的概率更大.

[4]这和极大后验估计是一样的, 但 Bayes 回归会

定理 1.16 Bayes 回归 设训练集为 $\mathcal{D}$, 根据 Bayes 回归可以得到任意输入 x 的预测值 t 的概率密度函数

$$p(t|x,\mathcal{D})=\int p(t|x,\boldsymbol{w})p(\boldsymbol{w}|\mathcal{D})\mathrm{d}\boldsymbol{w}=\mathcal{N}\left(t|m(x),s^2(x)\right)$$

其中

$$\boldsymbol{\phi}(x)=\left(1,x,\cdots,x^M\right)^{\mathrm{t}}$$

$$m(x)=\beta\,\boldsymbol{\phi}(x)^{\mathrm{t}}\,\boldsymbol{S}\sum_{i=1}^{N}t_i\,\boldsymbol{\phi}(x_i)$$

$$s^2(x)=\beta^{-1}+\boldsymbol{\phi}(x)^{\mathrm{t}}\,\boldsymbol{S}\boldsymbol{\phi}(x)$$

$$\boldsymbol{S}^{-1}=\alpha\boldsymbol{I}+\beta\sum_{i=1}^{N}\boldsymbol{\phi}(x_i)\,\boldsymbol{\phi}(x_i)^{\mathrm{t}}$$

这里的 α,β 为先验分布与似然函数的参数, $\boldsymbol{I}$ 为单位矩阵.

证明. 我们仍然假定观测值 y 可以写成下面的形式:

$$y=g(\boldsymbol{x};\boldsymbol{w})+\epsilon$$

其中 ε 服从均值为 0, 方差为 β^{-1} 的正态分布. 假定 $\boldsymbol{w}$ 的先验也是多元高斯分布[5], 即:

$$p(\boldsymbol{w})=\mathcal{N}\left(\boldsymbol{w}|0,\alpha^{-1}\boldsymbol{I}\right)$$

即各分量 w_i 满足均值为 0, 方差为 α^{-1} 的正态分布, 并且各分量相互独立. 由于噪声是独立同分布, 因此有

$$p\left(\mathcal{D}|\boldsymbol{w}\right)=\prod_{i=1}^{N}\mathcal{N}\left(t_i|g\left(\boldsymbol{x}_i;\boldsymbol{w}\right),\beta^{-1}\right)$$

取对数可得

$$\ln p\left(\mathcal{D}|\boldsymbol{w}\right)=-\frac{\beta}{2}\sum_{i=1}^{N}\left(t_i-g\left(\boldsymbol{x}_i;\boldsymbol{w}\right)\right)^2+\mathrm{const}$$

将上式中的平方项展开并忽略与 $\boldsymbol{w}$ 无关的项, 可得

$$\ln p\left(\mathcal{D}|\boldsymbol{w}\right)=-\frac{\beta}{2}\left(\boldsymbol{w}^{\mathrm{t}}\sum_{i=1}^{N}\boldsymbol{\phi}\left(\boldsymbol{x}_i\right)\boldsymbol{\phi}\left(\boldsymbol{x}_i\right)^{\mathrm{t}}\boldsymbol{w}-2\sum_{i=1}^{N}t_i\,\boldsymbol{\phi}\left(\boldsymbol{x}_i\right)^{\mathrm{t}}\boldsymbol{w}\right)+\mathrm{const}$$

先验概率 $p(\boldsymbol{w})$ 的对数为

$$\ln p(\boldsymbol{w})=-\frac{\alpha}{2}\boldsymbol{w}^{\mathrm{t}}\boldsymbol{w}+\mathrm{const}$$

[5]多元高斯分布的概率密度函数为

$$p(\boldsymbol{x})=\frac{1}{(2\pi)^{n/2}|\boldsymbol{\Sigma}|^{1/2}}\exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^{\mathrm{t}}\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right)$$

其中 $\boldsymbol{\mu}$ 为均值矢量, $\boldsymbol{\Sigma}$ 为协方差矩阵. 对于这里的先验分布, 均值矢量为 $\boldsymbol{0}$, 协方差矩阵为 $\alpha^{-1}\boldsymbol{I}$. 事实上, 多元高斯分布等价于每个分量满足独立的一元高斯分布.

由 Bayes 定理, 我们有

$$\begin{aligned}\ln p(\boldsymbol{w}|\mathcal{D}) &= \ln p\left(\mathcal{D}|\boldsymbol{w}\right) + \ln p(\boldsymbol{w}) + \text{const} \\ &= -\frac{1}{2}\left(\beta \boldsymbol{w}^{\mathrm{t}} \sum_{i=1}^{N} \boldsymbol{\phi}\left(\boldsymbol{x}_i\right) \boldsymbol{\phi}\left(\boldsymbol{x}_i\right)^{\mathrm{t}} \boldsymbol{w} + \alpha \boldsymbol{w}^{\mathrm{t}} \boldsymbol{w}\right) + \beta \sum_{i=1}^{N} t_i \, \boldsymbol{\phi}\left(\boldsymbol{x}_i\right)^{\mathrm{t}} \boldsymbol{w} + \text{const} \\ &= -\frac{1}{2} \boldsymbol{w}^{\mathrm{t}} \left(\beta \sum_{i=1}^{N} \boldsymbol{\phi}\left(\boldsymbol{x}_i\right) \boldsymbol{\phi}\left(\boldsymbol{x}_i\right)^{\mathrm{t}} + \alpha \boldsymbol{I}\right) \boldsymbol{w} + \left(\beta \sum_{i=1}^{N} t_i \boldsymbol{\phi}\left(\boldsymbol{x}_i\right)^{\mathrm{t}}\right) \boldsymbol{w} + \text{const}\end{aligned}$$

令

$$\boldsymbol{S}^{-1} = \alpha \boldsymbol{I} + \beta \sum_{i=1}^{N} \boldsymbol{\phi}\left(\boldsymbol{x}_i\right) \boldsymbol{\phi}\left(\boldsymbol{x}_i\right)^{\mathrm{t}} \qquad \boldsymbol{b} = \beta \sum_{i=1}^{N} t_i \boldsymbol{\phi}\left(\boldsymbol{x}_i\right)^{\mathrm{t}}$$

则有

$$\ln p(\boldsymbol{w}|\mathcal{D}) = -\frac{1}{2} \boldsymbol{w}^{\mathrm{t}} \boldsymbol{S}^{-1} \boldsymbol{w} + \boldsymbol{b}^{\mathrm{t}} \boldsymbol{w} + \text{const}$$

我们可以发现, 上述式子恰好可以写作平方展开的形式. 令 $\boldsymbol{S}^{-1}\boldsymbol{m} = \boldsymbol{b}$, 则有

$$\ln p(\boldsymbol{w}|\mathcal{D}) = -\frac{1}{2}\left(\boldsymbol{w} - \boldsymbol{m}\right)^{\mathrm{t}} \boldsymbol{S}^{-1} \left(\boldsymbol{w} - \boldsymbol{m}\right) + \text{const}$$

这说明后验分布 $p(\boldsymbol{w}|\mathcal{D})$ 也是一个高斯分布, 其均值为 $\boldsymbol{m}$, 协方差矩阵为 $\boldsymbol{S}$:

$$p(\boldsymbol{w}|\mathcal{D}) = \mathcal{N}\left(\boldsymbol{w}|\boldsymbol{m}, \boldsymbol{S}\right)$$

最后用 Bayes 定理计算预测值 t 的概率密度函数:

$$\begin{aligned}p(t|x, \mathcal{D}) &= \int p(t|x, \boldsymbol{w}) p(\boldsymbol{w}|\mathcal{D}) \mathrm{d}\boldsymbol{w} \\ &= \int \mathcal{N}\left(t|\boldsymbol{w}^{\mathrm{t}} \boldsymbol{\phi}\left(x\right), \beta^{-1}\right) \mathcal{N}\left(\boldsymbol{w}|\boldsymbol{m}, \boldsymbol{S}\right) \mathrm{d}\boldsymbol{w} \\ &= \mathcal{N}\left(t|m(x), s^2(x)\right)\end{aligned}$$

□

1.3 基函数

1.3.1 基函数的概念

定义 1.17 基函数 回归模型的函数 $f\left(\boldsymbol{x}; \boldsymbol{w}\right)$ 可以表示为一组函数的线性组合:

$$f\left(\boldsymbol{x}; \boldsymbol{w}\right) = w_0 \phi_0\left(\boldsymbol{x}\right) + \cdots + w_M \phi_M\left(\boldsymbol{x}\right) = \sum_{j=0}^{M} w_j \phi_j\left(\boldsymbol{x}\right) = \boldsymbol{w}^{\mathrm{t}} \boldsymbol{\phi}\left(\boldsymbol{x}\right)$$

这组函数 $\phi_0\left(\boldsymbol{x}\right), \cdots, \phi_M\left(\boldsymbol{x}\right)$ 称为**基函数 (Basis Functions)**.

1.3.2 常见的基函数模型

多项式基函数 多项式基函数的形式为

$$\phi_j\left(x\right) = x^j (j = 0, 1, \cdots, M)$$

多项式基函数具有全局性, 一处的改变会影响所有函数的改变.

高斯基函数 高斯基函数的形式为

$$\phi_j(x) = \exp\left(-\frac{(x-\mu_j)^2}{2\sigma^2}\right)$$

其中 μ_j 为高斯基函数的中心, σ 为宽度. 高斯基函数具有局部性, 一处的改变只会影响与其中心相近的函数.

Sigmoid 基函数 Sigmoid 基函数的形式为

$$\phi_j(x) = \frac{1}{1+\exp(-\beta(x-\mu_j))}$$

其中 μ_j 为 Sigmoid 基函数的中心, β 为宽度. Sigmoid 基函数具有平滑性, 在中心附近变化较大, 而在两侧变化较小.

Sigmoid 基函数常用于神经网络中, 作为激活函数.

1.3.3 基函数用于回归

如果仍然采用前面的假设, 即观测值 y 可以写成下面的形式:

$$y = f(\boldsymbol{x}; \boldsymbol{w}) + \epsilon$$

其中 $f(\boldsymbol{x}; \boldsymbol{w})$ 是由基函数构成的线性组合:

$$f(\boldsymbol{x}; \boldsymbol{w}) = \sum_{j=0}^{M} w_j \phi_j(\boldsymbol{x}) = \boldsymbol{w}^{\mathrm{t}} \boldsymbol{\phi}(\boldsymbol{x})$$

那么通过极大似然估计可以得到

$$\boldsymbol{w}_{ML} = \left(\boldsymbol{\Phi}^{\mathrm{t}}\boldsymbol{\Phi}\right)^{-1}\boldsymbol{\Phi}^{\mathrm{t}}\boldsymbol{y} = \boldsymbol{\Phi}^{\dagger}\boldsymbol{y}$$

其中**设计矩阵 $\boldsymbol{\Phi}$** 为

$$\boldsymbol{\Phi} = \begin{bmatrix} \phi_0(\boldsymbol{x}_1) & \cdots & \phi_M(\boldsymbol{x}_1) \\ \vdots & \ddots & \vdots \\ \phi_0(\boldsymbol{x}_N) & \cdots & \phi_M(\boldsymbol{x}_N) \end{bmatrix}$$

当数据太多时, 可以采用循序学习的方式, 即每次只用一组数据进行更新.

定义 1.18 循序学习 **循序学习 (Sequential Learning)** 是指每次只用一组数据进行更新的学习方式. 对于给定的一组数据 $(\boldsymbol{x}_n, t_n)$, 我们有

$$\boldsymbol{w}^{(n)} = \boldsymbol{w}^{(n-1)} + \eta\left(t_n - \left(\boldsymbol{w}^{(n-1)}\right)^{\mathrm{t}}\boldsymbol{\phi}(\boldsymbol{x}_n)\right)\boldsymbol{\phi}(\boldsymbol{x}_n)$$

这里的 η 为学习率, 控制每次更新的幅度.

循序学习也称为在线学习, 在神经网络与深度学习中被称作随机梯度下降.

另外, 如果采用正则化误差函数, 例如岭回归, 那么结果为

$$\boldsymbol{w}_{ML} = \left(\lambda \boldsymbol{I} + \boldsymbol{\Phi}^{\mathrm{t}} \boldsymbol{\Phi}\right)^{-1} \boldsymbol{\Phi}^{\mathrm{t}} \boldsymbol{y}$$