第3章　回归算法中的贝叶斯

在第 2 章，我们对多项式模型使用最小二乘估计来拟合曲线，并且将 L_1 正则化和 L_2 正则化手段作为结构风险直接引入，那么在第 3 章，我们会用概率的观点来看待机器学习模型，用简单的例子帮助大家理解判别式模型和生成式模型的区别。通过思考曲线拟合的问题，还会发现习以为常的损失函数和正则化项背后有着深刻的意义。

3.1　快速理解判别式模型和生成式模型

从概率的角度来理解数据有着两个不同的角度，假设我们有 5 个数据点，每个数据都只有一个特征 x 和一个目标值 y：

x	0	0	1	1	1
y	0	0	2	1	2

一种是条件概率的角度，它描述了目标值相对于数据的特征出现的概率，我们表示为：

$$P(y=0 \mid x=0)=1 \tag{3.1}$$

$$P(y=2 \mid x=1)=\frac{2}{3} \tag{3.2}$$

$$P(y=1 \mid x=1)=\frac{1}{3} \tag{3.3}$$

另一种是联合概率的角度，它描述了数据的特征和目标值一起出现的概率，我们表示为：

$$P(y=0,x=0)=\frac{2}{5} \tag{3.4}$$

$$P(y=2,x=1)=\frac{2}{5} \tag{3.5}$$

$$P(y=1,x=1)=\frac{1}{5} \tag{3.6}$$

这两种角度分别代表了两种不同的建模方法，条件概率是将数据特征与目标值直接联系在一起，对于每一个特征我们只需要计算 $P(y|x)$，我们将这样的模型叫作判别式模型(Discriminative Model)，可以看到如果是利用判别式模型去预测新的数据 $x=0$，它会给出 $y=0$。联合概率是综合考虑了整个样本空间，对于每一个特征我们需要计算 $P(y,x)$，我们将这样的模型叫作生成式模型(Generative Model)，如果去预测新的数据 $x=1$，会比较 $y=1$ 的概率是 $\frac{1}{5}$，$y=2$ 的概率是 $\frac{2}{5}$，选择概率较大的 $y=2$。

定理 3.1(贝叶斯定理) 贝叶斯定理(Bayes' theorem)可以从全概率公式推导而来，但含义却更加丰富，简而言之，事件发生的条件可能性和依据此条件发生此事，概率是不一样的。我们分别用 A 和 B 表示事件，贝叶斯定理写作：

$$p(A\mid B)=\frac{p(B\mid A)p(A)}{\int_{A'}p(B\mid A')p(A')\mathrm{d}A'} \tag{3.7}$$

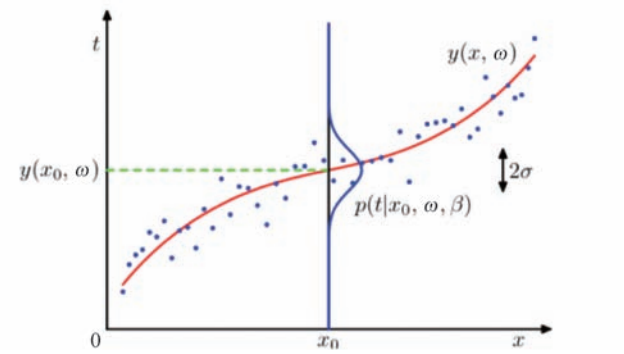
其中，$p(A)$表示先验，是统计量或者只是假设偏好；$p(B|A)$是似然函数，是在条件 A 下的 B 出现的概率；$p(A\mid B)$是后验概率，表示 B 的出现是因为 A 的概率，$\int_{A'}p(B\mid A')P(A')\mathrm{d}A'$ 为证据因子，对条件概率的积分表示将所有的事件发生的条件都考虑在内。

生成式模型和判别式模型往往都与贝叶斯定理相联系，见定理 3.4。因为有 $P(x,y)=P(y|x)P(x)$，联合概率比条件概率还多考虑了数据特征的分布。贝叶斯定理中的后验概率随着我们的任务有着不同的含义，比如在朴素贝叶斯分类器(见第 4 章)中，事件 A 和 B 分别指类别和需要预测的样本，在线性回归或者线性分类算法中，事件 A 和 B 分别指条件分布的参数和训练样本。

3.2 极大似然估计和平方损失

回归问题中，我们可以将每一个样本 x 对应的目标值看作一个均值为 ωx 的连续分布，如图 3.1 所示，它只假设分布 $p(y|x)$服从高斯分布，而不关心 $p(x)$，所以训练过程本质上是在对这个条件分布的参数做估计(此章讨论一维变量的情形，下同)。

以这样的视角来重新考虑目标值的分布会发现，每一个样本的目标值 y_i 都服从高斯分布 $N(\omega^{\mathrm{T}}x_i,\sigma^2)$，它的均值为 $\omega^{\mathrm{T}}x_i$，假设样本是独立同分布的，那么目标值的分布就是所有



■图 3.1 每一个样本 x_0 都对应着一个高斯分布，分布的均值作为真实值

样本分布的乘积，形式为：

$$p(y \mid x_0, \omega) = \prod_i N(\omega^{\mathrm{T}} x_i, \sigma^2) \tag{3.8}$$

定理 3.2（极大似然估计） 给定分布的概率密度函数 f，这个概率分布由参数 θ 控制，我们从分布中采样 $X_1, X_2, X_3, \cdots, X_n$，似然函数就是样本关于该参数的条件概率：

$$L(\theta \mid X_1, X_2, \cdots, X_n) = f_\theta(X_1, X_2, \cdots, X_N) \tag{3.9}$$

最大化似然函数的意义就是在参数 θ 的所有的可能取值中，寻找一个使得采样最可能出现的 θ，可能性最大，意味着似然函数也达到了最大值。

因为总的似然函数等于所有样本分布的乘积，大量的小的数连乘会造成数值下溢，所以我们将似然函数取对数，连乘就变为了对数求和：

$$\ln p(y \mid x_0, \omega) = \sum_i \ln N(\omega^{\mathrm{T}} x_i, \sigma^2) \tag{3.10}$$

最大化对数似然就是最大化多个高斯分布的对数和：

$$\underset{\omega}{\operatorname{argmax}} \sum_i^m \ln\left(\frac{1}{\sigma\sqrt{2\pi}} \mathrm{e}^{-\frac{(y_i - \omega^{\mathrm{T}} x_i)^2}{2\sigma^2}}\right) \tag{3.11}$$

利用对数的性质，就可以将其拆开：

$$\underset{\omega}{\operatorname{argmax}} - \sum_i^m \left[\frac{1}{2\sigma^2}(y_i - \omega^{\mathrm{T}} x_i)^2 + \ln(\sigma\sqrt{2\pi})\right] \tag{3.12}$$

其中 $\ln(\sigma\sqrt{2\pi})$ 与 ω 无关，最大化对数似然，相当于最小化其负值，所以，我们有：

$$\underset{\omega}{\operatorname{argmin}} \sum_{i=1}^m \frac{1}{2\sigma^2}(y_i - \omega^{\mathrm{T}} x_i)^2 \tag{3.13}$$

其中标准差 σ 独立于 ω，不参与优化。这样，我们就以极大似然估计的方法得到了均方误差的表达式。极大似然估计是贯穿统计学习和深度学习的参数估计办法，我们会经常使用它来得到损失函数，因为极大似然估计可以获得参数估计的一致性（见第 4 章）。

3.3 最大后验估计和正则化

极大似然估计假设了条件分布，从频率学派的角度看来，这个分布的参数是确定好的，我们只需要找到这个参数。但从贝叶斯学派的角度看来，参数本身就是一个随机变量，我们需要找到的是这个参数的分布。如果我们考虑贝叶斯定理，将最大似然估计做进一步的扩展，我们最大化的不再是似然函数，而是似然函数与先验的乘积，也就得到了极大后验估计，见定理 3.3。

其中，先验概率就表达了参数的分布，只有在这个分布之下，参数才有可能被考虑。所以最后的估计结果会使得参数向先验的方向移动，如果采用极大后验来得到损失函数，先验概率的存在则对应着损失函数的正则项。

定理 3.3（最大后验估计 MAP） 在最大似然估计的基础上，我们选择最大化似然函数和先验概率的乘积：

$$\theta_{\mathrm{MAP}}=\underset{\theta}{\operatorname{argmax}} f_{\theta}(X_1,X_2,\cdots,X_n)*p(\theta) \tag{3.14}$$

如果先验分布为均匀分布，先验项会在结果上变为一个常数，在此条件下，最大后验估计和极大似然估计给出的结果虽然一致，但是在论述上仍然有着本质的不同。

如图 3.2 所示，先假设参数的先验分布为均值为零的拉普拉斯分布：

$$p(\omega)=\frac{1}{2\eta}\mathrm{e}^{-\frac{|\omega|}{\eta}} \tag{3.15}$$

其中，η 为拉普拉斯分布的尺度参数。同样因为取对数，乘积变为了求和：

$$\underset{\omega}{\operatorname{argmax}}\left(\sum_{i}^{m}\ln\left(\frac{1}{\sigma\sqrt{2\pi}}\mathrm{e}^{-\frac{(y_i-\omega^{\mathrm{T}}x_i)^2}{2\sigma^2}}\right)+\sum_{j}^{d}\ln\left(\frac{1}{2\eta}\mathrm{e}^{-\frac{|\omega|}{\eta}}\right)\right) \tag{3.16}$$

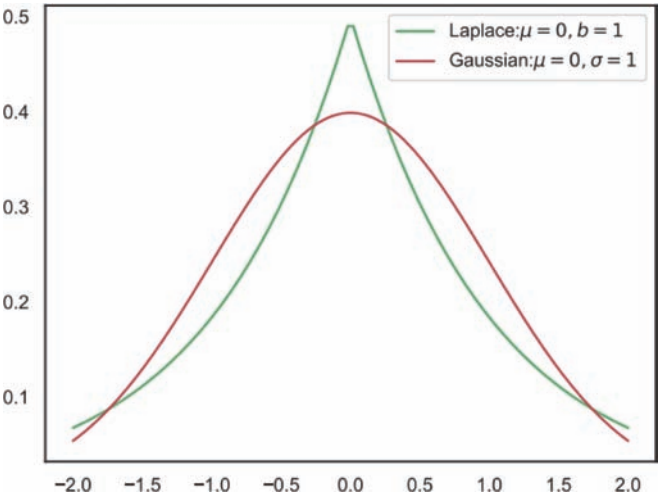


图 3.2 红线和绿线表示标准高斯分布和拉普拉斯分布

其中，d 表示参数的个数，我们可以延续上述步骤化解得到：

$$\underset{\omega}{\operatorname{argmax}}\left(-\sum_{i}^{m}\left[\frac{1}{2\sigma^2}(y_i-\omega^{\mathrm{T}}x_i)2+\ln(\sigma\sqrt{2\pi})\right]-\sum_{j}^{d}\left[\frac{1}{\eta}|\omega_j|+\ln(2\eta)\right]\right) \tag{3.17}$$

最大化对数似然就是最小化其负值，同时省略其中的常数项，就会得到：

$$\underset{\omega}{\operatorname{argmin}}\left(\sum_{i=1}^{m}\frac{1}{2\sigma^2}(y_i-\omega^{\mathrm{T}}x_i)^2+\sum_{j}^{d}\frac{1}{\eta}|\omega_j|\right) \tag{3.18}$$

其中，σ 和 η 是常数，不参与优化。我们就利用最大后验估计得到了 L_1 正则化的形式。

同理，我们将先验概率替换为均值为零的高斯分布，其方差由 τ^2 所控制，继续上述的步骤会得到 L_2 正则化：

$$\underset{\omega}{\operatorname{argmin}}\sum_{i=1}^{m}\frac{1}{2\sigma^2}(y_i-\omega^{\mathrm{T}}x_i)^2+\sum_{j}^{d}\frac{1}{2\tau^2}(\omega_j)^2 \tag{3.19}$$

我们会发现，正则化将权重系数缩减到零的操作恰恰对应了先验分布中概率密度最大的区域，我们将均值设为零，估计的参数会更偏好零。所以，我们可以自由的控制参数向我们期望的方向移动，只需要调节先验的均值。并且，先验分布的尺度参数（我们将均值叫作分布的位置参数）对应着正则化项系数，它越小，表示分布的尺度越小，精度越高，正则化项起到的作用也就越大。

3.4 贝叶斯线性估计

最大似然估计是点估计，在参数空间上得到一个点作为最后的结果，最大后验估计虽然添加了先验，但其估计的只是后验分布的众数，最后得到的仍然是一个点。单纯地采用极大似然估计会带来模型容量和过拟合的问题，采用最大后验来添加正则化项虽然可以防止过拟合，但却引入了额外的超参数，需要做交叉验证的重复训练来大致确定超参数的值，并且先验分布的不合理会在小数据集上带来致命的后果。

参数的整个分布包含着比单纯的一个点更多的信息，如果根据贝叶斯定理可以得到参数的后验分布，贝叶斯线性估计的第一步就是获得参数的后验分布。然而根据贝叶斯定理，后验分布需要计算整个参数空间的证据因子，这往往是一个高维积分，需要用一些数值模拟的技术来计算。但是我们可以采用高斯分布自共轭的性质，见定理 3.4，就可以直接写出后验分布的封闭形式，进而简化了计算。

定理 3.4（共轭先验） 在贝叶斯定理中，如果先验分布和后验分布属于同一类分布，则此时的先验分布和后验分布为一对共轭分布，此时的先验分布为似然函数的共轭先验：

$$p(\theta\mid x)=\frac{p(x\mid\theta)p(\theta)}{\int_{\theta'}p(x\mid\theta)p(\theta')\mathrm{d}\theta} \tag{3.20}$$

讨论共轭分布，需要指明似然函数的形式。高斯分布在高斯的似然函数下为自共轭分布，也就是说，如果似然函数和先验分布均为高斯，那么后验分布也为高斯分布。

我们考虑参数的高斯先验分布和似然函数：

$$p(\omega)=N(\omega \mid \mu_0,\sigma_0^{-1}) \tag{3.21}$$

$$p(y \mid \boldsymbol{\omega})=N(\omega^{\mathrm{T}}x,\beta^{-1}) \tag{3.22}$$

均值和标准差一旦确定，整个分布就确定了。如果我们设置先验的均值为零，将大大地简化问题，此时最大化后验分布的对数会得到 L_2 正则化同样的结果：

$$\underset{\omega}{\operatorname{argmin}}\sum_{i=1}^{m}\frac{\beta}{2}(y_i-\omega^{\mathrm{T}}x_i)^2+\sum_{j}^{d}\frac{\sigma_0}{2}(\omega_j)^2 \tag{3.23}$$

我们将这样的形式叫作贝叶斯岭回归(Bayesian Ridge Regression)。

为了更充分地利用数据和预测分布，从数据中直接提取先验，我们会采取增量计算的方式，将数据分批处理，此种增量计算的基本过程是：首先确定一个先验分布，利用少量的数据点去计算后验分布，然后将前一步计算而来的后验作为下一步的先验，接着导入部分数据点，再继续得到后验分布，再将此时的后验当作下一步的先验，直到计算完整个数据集。与一次性计算完全部的数据不同，增量计算的好处在于随时可以根据数据来调整先验分布的参数，在样本并不能很好地反映总体(这是经常发生的，我们把这种情况叫作数据的不一致性)，或者数据量较少时，会有很不错的性能。

此时我们还并未真正接触到贝叶斯线性回归的威力。后验分布就是关于参数的条件分布，如果我们不使用最大化后验分布来得到损失函数，而是将得到的后验分布与似然函数相结合，就能最终实现对新样本的预测。为什么后验分布可以直接与似然函数相乘呢？因为贝叶斯的增量计算会将上一步的后验当作下一步的先验，当我们用来做新样本的预测，使用的先验分布就是训练完成后的后验分布。

假设新的样本为 y_{new}，对于新样本的预测也呈现一个分布，为：

$$P(y_{\mathrm{new}} \mid \mu_N,\sigma_N)=\int P(y_{\mathrm{new}} \mid \omega)P(\omega \mid \mu_N,\sigma_N)\mathrm{d}\omega \tag{3.24}$$

可以看出，预测分布成为了两个高斯分布的卷积，卷积的本质就是加权平均(卷积神经网络中的卷积操作概念本质上也是加权平均)，在参数的所有可能位置上，后验分布被看作权重，似然函数给出的预测就被平均化。点估计只考虑了一个参数值，参数值的不恰当就会导致过拟合，而贝叶斯估计是在整个参数空间上取平均，使得最后的结果更加稳定。

最后我们来讨论预测分布的性质，见定理 3.5，通过卷积两个高斯分布得到的方差为：

$$\begin{aligned}\sigma_{\mathrm{new}}^2&=\beta^{-1}+\sigma_N^2\\&=\frac{1}{\beta}+\frac{1}{\sigma_0^{-1}+\beta x^{\mathrm{T}}x}\end{aligned} \tag{3.25}$$

定理 3.5(高斯分布的卷积) *两个高斯分布的卷积仍然是一个高斯分布，新的高斯分布的方差为两个高斯分布的方差之和。*

可以证明，随着样本的增加，预测分布的方差会越来越小。在此章中，我们使用了一维变量假设分布，来使得问题尽可能地简单，实际应用中，我们会遇到多维分布，只需要将方差推广到协方差即可。

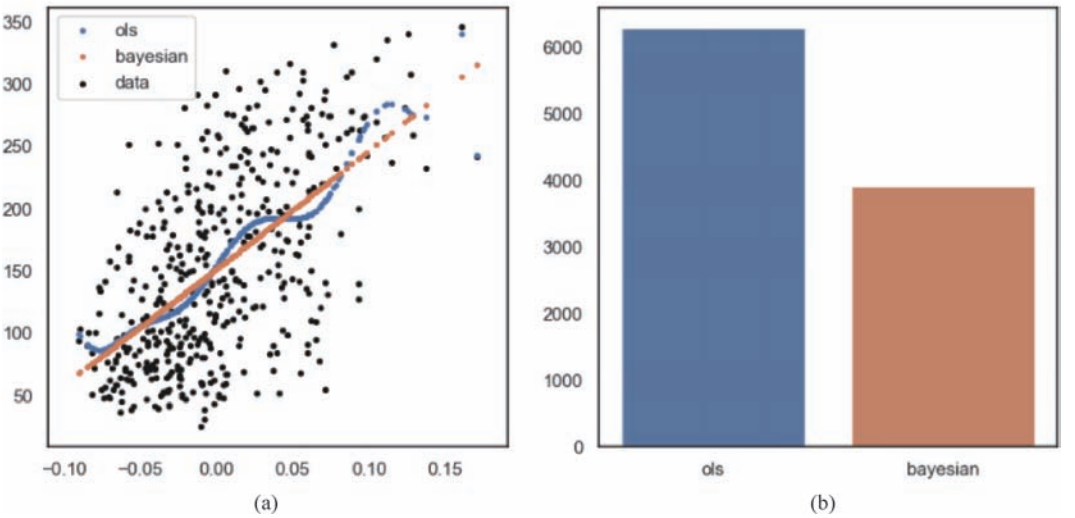
3.5 使用 scikit-learn

在 sklearn 中我们通过 BayesianRidge 类可以很方便地实现贝叶斯岭回归，我们仍然使用 sklearn 的 diabetes 数据集，选取 BMI 指数作为特征，为了尽可能地将模型变得复杂，我们使用多项式回归的办法将项数扩展为 10，依次采用普通的线性回归和贝叶斯岭回归的办法，观察其在样本空间的曲线形式，同时为了详细验证贝叶斯和普通线性回归的性能差异，我们对这两种算法分别作 10 折的交叉验证，把 10 次交叉的均方误差的平均值作为性能指标，来观察两种算法的泛化能力。

```
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import PolynomialFeatures as PF
from sklearn.linear_model import BayesianRidge as BR
from sklearn.linear_model import LinearRegression as LR
from sklearn import datasets
from sklearn.model_selection import cross_val_score

data = datasets.load_diabetes()
X = data['data'][:,2][:,np.newaxis]
y = data['target']

poly = PF(degree = 10)
X_poly = poly.fit_transform(X)

Regressors = dict(ols = LR(),bayesian = BR())

sns.set(style = 'white')
plt.subplot(1,2,1)
plt.scatter(X,y,s = 8,color = 'k',label = 'data')
for name,reg in Regressors.items():
    reg.fit(X_poly,y)
    y_pred = reg.predict(X_poly)
    plt.plot(X,y_pred,'.',markersize = 5,label = name)
plt.legend()
plt.subplot(1,2,2)
cross_scores = []
cross_names = []
for name,reg in Regressors.items():
    score = cross_val_score(reg,X_poly,y,cv = 10,scoring = 'neg_mean_squared_error')
    cross_scores.append(-score.mean())
    cross_names.append(name)
sns.barplot(cross_names,cross_scores)
```

```
plt.show()
```

从图 3.3 可以看出，两种算法都通过多项式来增加特征，普通线性回归所得到的结果较为弯曲，表现出了一定的过拟合，而贝叶斯岭回归的结果却近似为一条直线，意味着拥有更好的泛化能力。因为多项式回归中模型的复杂度就取决于其项数，普通线性回归使用极大似然来估计参数，无法避免数据与模型的匹配问题，过于复杂的模型面对简单的数据，很容易产生过拟合。而贝叶斯方法的先验参数由训练数据所提供，相当于用训练数据本身来限制模型的复杂度，并且我们可以清晰地看到贝叶斯回归的泛化性能要优于普通的线性回归。



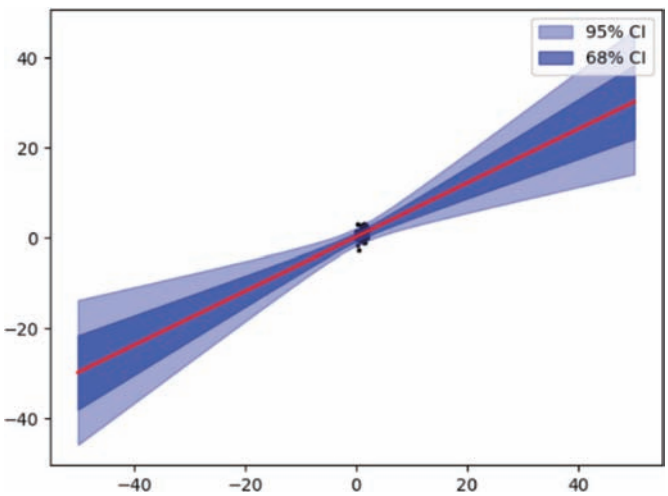
■图 3.3 (a)为两种算法在样本空间的行为，蓝色的线表示普通线性回归的结果，橙色的线表示贝叶斯岭回归的结果，(b)为 10 折交叉验证后的均方误差平均值对比，蓝色的直方为普通线性回归，橙色的直方为贝叶斯的结果

接下来，我们来验证贝叶斯岭回归的预测分布的方差是否与训练样本有关。我们生成 100 个数据点，服从正比关系：$y=x$，并对 y 添加小的高斯噪声，用贝叶斯岭回归做训练，然后用其来预测更多的数据，观察它在样本空间的方差：

```
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.linear_model import BayesianRidge as BR
from sklearn.preprocessing import PolynomialFeatures as PF

np.random.seed(2019)
X = np.linspace(0,2,100)
y = X + np.random.randn(100)

br = BR()
```

```
br.fit(X[:,np.newaxis],y)

X_ = np.linspace(-50, 50, 1000)
y_mean, y_std = br.predict(X_[:,np.newaxis],return_std=True)

plt.figure()
plt.scatter(X,y,s=3,c='k')
plt.plot(X_, y_mean, 'r', lw=2, zorder=9)
plt.fill_between(X_, y_mean-1.96*y_std, y_mean + 1.96*y_std, \
          color='blue',alpha=0.3,label='95% CI')
plt.fill_between(X_, y_mean-1*y_std, y_mean + 1*y_std,\
          color='blue',alpha=0.5,label='68% CI')
plt.legend()
plt.show()
```

从图 3.4 可以看出，只有经过训练数据以及与训练数据相近的点可以缩小预测分布的方差，远离训练数据的点在预测分布的方差在逐渐增大，表示预测分布的不确定性越来越高。侧面证明了样本的增多可以缩小预测分布的方差，但也证明了训练样本的多样化（在不同的 x 均有样本）与训练集的规模大小同样重要，因为多样化可以减小整体的不确定性。



■图 3.4 红线为拟合结果，深蓝色为 68%置信区间，浅蓝色为 95%置信区间