第6章
逻辑斯谛回归与最大熵模型

目录

A

逻辑斯谛回归

1 逻辑斯谛回归

定义 6.1（逻辑斯谛分布） 设 X 是连续随机变量，X 服从逻辑斯谛分布是指 X 具有下列分布函数和密度函数:

$$F(x) = P(X \leqslant x) = \frac{1}{1+\mathrm{e}^{-(x-\mu)/\gamma}} \tag{6.1}$$

$$f(x) = F'(x) = \frac{\mathrm{e}^{-(x-\mu)/\gamma}}{\gamma(1+\mathrm{e}^{-(x-\mu)/\gamma})^2} \tag{6.2}$$

式中，μ 为位置参数，$\gamma > 0$ 为形状参数。

分布函数属于逻辑斯谛回归函数，其图形是一条S形曲线(sigmoid curve)。该曲线以(μ，1/2)为中心对称，即满足

$$F(-x+\mu)-\frac{1}{2} = -F(x+\mu)+\frac{1}{2}$$

1 逻辑斯谛回归

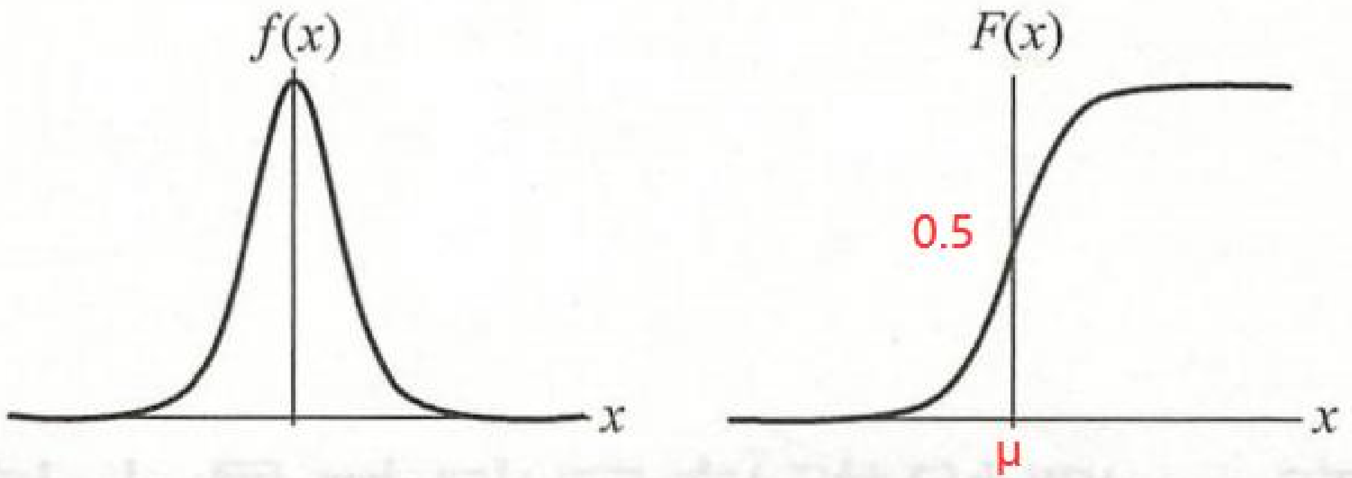


图 6.1 逻辑斯谛分布的密度函数与分布函数

当μ = 0, γ = 1时，得到sigmoid函数

$$F(x)=\frac{1}{1+e^{-x}}$$

1 逻辑斯谛回归

二项逻辑斯谛回归模型：

定义 6.2（逻辑斯谛回归模型） 二项逻辑斯谛回归模型是如下的条件概率分布：

$$P(Y=1|x)=\frac{\exp(w\cdot x+b)}{1+\exp(w\cdot x+b)} \tag{6.3}$$

$$P(Y=0|x)=\frac{1}{1+\exp(w\cdot x+b)} \tag{6.4}$$

这里，$x\in\mathbf{R}^n$ 是输入，$Y\in\{0,1\}$ 是输出，$w\in\mathbf{R}^n$ 和 $b\in\mathbf{R}$ 是参数，w 称为权值向量，b 称为偏置，$w\cdot x$ 为 w 和 x 的内积。

有时为了方便，将权值向量和输入向量加以扩充，仍记作 w，x，即 $w=(w^{(1)},w^{(2)},\cdots,w^{(n)},b)^{\mathrm{T}}$，$x=(x^{(1)},x^{(2)},\cdots,x^{(n)},1)^{\mathrm{T}}$。这时，逻辑斯谛回归模型如下：

$$P(Y=1|x)=\frac{\exp(w\cdot x)}{1+\exp(w\cdot x)} \tag{6.5}$$

$$P(Y=0|x)=\frac{1}{1+\exp(w\cdot x)} \tag{6.6}$$

1 逻辑斯谛回归

线性回归 $P = h(x) = \omega_0 + \omega_1 x_1 + \omega_2 x_2 + \ldots + \omega_n x_n = \vec{\omega}\vec{x}$

使用线性回归可能存在的问题：

- 等式左边值域[0, 1], 右边的范围为无穷
- 实际情况中，可能存在当x很小或者很大时，对h(x)的影响很小，当x达到中间某个阈值时，对h(x)的影响很大。

设 $P(Y=1|x)=p \quad P(Y=0|x)=1-p$

$$\text{logit}(p) = \log\frac{p}{1-p} = \vec{\omega}\cdot\vec{x} \Rightarrow$$

$$P(Y=1|x) = \frac{\exp(w\cdot x)}{1+\exp(w\cdot x)}$$

$$P(Y=0|x) = \frac{1}{1+\exp(w\cdot x)}$$

线性函数值越接近正无穷，概率值越接近1；线性函数中越接近负无穷，概率值越接近0.

1 逻辑斯谛回归

二项逻辑斯谛回归模型参数估计：

$$T=\{(x_1,y_1),(x_2,y_2),\dots,(x_N,y_N)\}, \text{其中} x_i\in R^n, y_i\in\{0,1\}$$

设 $P(Y=1|x)=\pi(x)\quad P(Y=0|x)=1-\pi(x)$

概率密度函数 $P(Y=y_i|x=x_i;\omega)=[\pi(x_i)]^{y_i}[1-\pi(x_i)]^{1-y_i}$

样本数据独立，似然函数 $P(Y=y_i|x=x_i;\omega)=\prod_{i=1}^{N}[\pi(x_i)]^{y_i}[1-\pi(x_i)]^{1-y_i}$

对数似然函数为

$$\begin{aligned}
L(w)&=\sum_{i=1}^{N}[y_i\log\pi(x_i)+(1-y_i)\log(1-\pi(x_i))]\\
&=\sum_{i=1}^{N}\left[y_i\log\frac{\pi(x_i)}{1-\pi(x_i)}+\log(1-\pi(x_i))\right]\\
&=\sum_{i=1}^{N}[y_i(w\cdot x_i)-\log(1+\exp(w\cdot x_i))]
\end{aligned}$$

$$\underbrace{P(Y=1|x)}_{\pi(x)}=\frac{\exp(w\cdot x)}{1+\exp(w\cdot x)}$$

$$P(Y=0|x)=\frac{1}{1+\exp(w\cdot x)}$$

$$\omega^*=\arg\max_{\omega}L(\omega)=-\arg\min_{\omega}L(\omega)$$

1 逻辑斯谛回归

逻辑斯谛回归通常采用梯度下降法或拟牛顿法学习参数ω

梯度下降法：

$$L(\omega) = \sum_{i=1}^{N}[y_i(w \cdot x_i) - \log(1+\exp(w \cdot x_i))]$$

对ω求导

$$\nabla L(\omega) = \sum_{i=1}^{N}[x_i y_i - \frac{e^{\omega x_i}}{1+e^{\omega x_i}} \cdot x_i]$$

$$\omega = \omega - \alpha \nabla L(\omega)$$

$$\omega_j = \omega_j - \alpha \frac{\partial L}{\partial \omega_j}$$

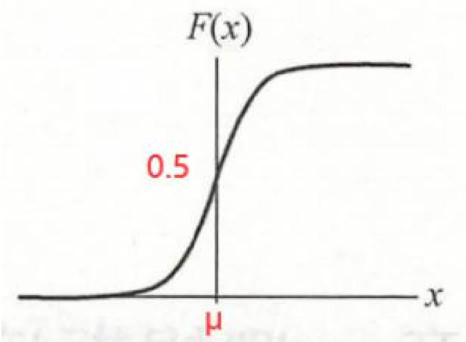
公式里的ω是向量，ω的第j个分量用ω_j表示，每个分量都同时更新。

对于权重向量ω，它的每一个维度的值，代表了对应这个维度的特征对于最终分类结果的贡献大小。假如这个维度是正，说明这个特征对于结果是有正向的贡献，那么它的值越大，说明这个特征对于分类为正起到的作用越重要。

1 逻辑斯谛回归

线性回归： $P = h(x) = \omega_0 + \omega_1 x_1 + \omega_2 x_2 + \ldots + \omega_n x_n = \vec{\omega}^T \cdot \mathrm{X}$

逻辑回归：$\begin{cases} P = h(x) = \mathrm{g}(\vec{\omega}^T X) \\ \mathrm{g}(x) = \dfrac{1}{1+e^{-x}} \end{cases}$



逻辑回归的决策边界：

假设当概率大于等于0.5，分类为1；小于0.5，分类为0。可见，当概率为0.5时为分界线，

若ω只有3个分量

$$\vec{\omega}^T X = 0 \Rightarrow \omega_0 + \omega_1 x_1 + \omega_2 x_2 = 0$$

$$x_2 = \frac{-\omega_0 - \omega_1 x_1}{\omega_2}$$

此时，决策边界是一条直线，由参数ω**决定。训练集确定**参数ω。

1 逻辑斯谛回归

$$\frac{P(Y=1|x)}{P(Y=K|x)}=\omega_1\cdot x$$

$$\frac{P(Y=2|x)}{P(Y=K|x)}=\omega_2\cdot x$$

$$\cdots$$

设离散型随机变量 Y 的取值集合是 $\{1,2,\cdots,K\}$，那么多项逻辑斯谛回归模型是

$$P(Y=k|x)=\frac{\exp(\boxed{w_k}\cdot x)}{1+\sum_{k=1}^{K-1}\exp(\boxed{w_k}\cdot x)},\quad k=1,2,\cdots,\boxed{K-1} \tag{6.7}$$

$$P(Y=K|x)=\frac{1}{1+\sum_{k=1}^{K-1}\exp(\boxed{w_k}\cdot x)} \tag{6.8}$$

这里，$x\in\mathbf{R}^{n+1}, w_k\in\mathbf{R}^{n+1}$。

二项逻辑斯谛回归的参数估计法也可以推广到多项逻辑斯谛回归。

B

最大熵模型

2 最大熵模型

熵：

$$H(P) = -\sum_{x} P(x)\log P(x) \tag{6.9}$$

熵满足下列不等式：

$$0 \leqslant H(P) \leqslant \log|X|$$

|X|是X的取值个数，当X服从均匀分布时，熵最大。

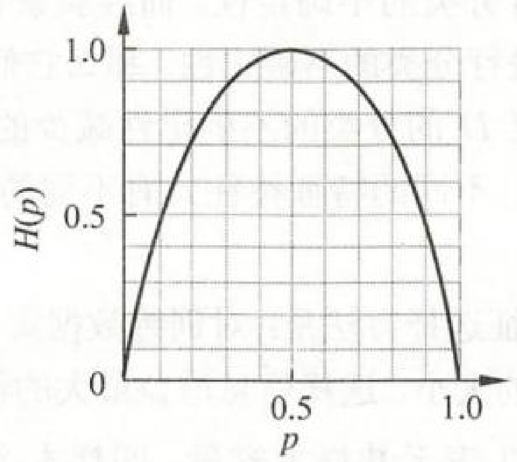


图 5.4　分布为伯努利分布时熵与概率的关系

最大熵原理：

最大熵原理认为概率模型首先必须满足已有的事实，即约束条件。在没有更多的信息的情况下，那些“不确定的部分”都是“等可能的”。最大熵原理通过熵的最大化来表示等可能性。“等可能性”不容易操作，而熵则是一个可优化的的数值指标。

2 最大熵模型

例 6.1 假设随机变量 X 有 5 个取值 $\{A,B,C,D,E\}$，要估计取各个值的概率 $P(A),P(B),P(C),P(D),P(E)$。

无约束条件：

$$P(A)+P(B)+P(C)+P(D)+P(E)=1$$

$$P(A)=P(B)=P(C)=P(D)=P(E)=\frac{1}{5}$$

有约束条件：有时，能从一些先验知识中得到一些对概率值的约束条件，例如：

$$P(A)+P(B)=\frac{3}{10}$$

$$P(A)+P(B)+P(C)+P(D)+P(E)=1$$

$$P(A)=P(B)=\frac{3}{20}$$

$$P(C)=P(D)=P(E)=\frac{7}{30}$$

样本中概率分布估计整体

$$\mathrm{P}(A)=\tilde{P}(A), \quad P(B)=\tilde{P}(B)$$

$$\mathrm{P}(A)+P(B)=\tilde{P}(A)+\tilde{P}(B)=\frac{3}{10}$$

2 最大熵模型

最大熵模型定义：

训练集 $T=\{(x_1,y_1),(x_2,y_2),\cdots,(x_N,y_N)\}$ 特征函数 $f_i(x,y)=\begin{cases}1，x和y满足某一事实\\0，否则\end{cases}$

样本中特征函数关于 $\widetilde{P}(x,y)$ 的数学期望：

$$E_{\tilde{p}}(f)=\sum_{x,y}\widetilde{P}(x,y)f(x,y)$$

$$\tilde{P}(X=x,Y=y)=\frac{\nu(X=x,Y=y)}{N}$$

$$\tilde{P}(X=x)=\frac{\nu(X=x)}{N}$$

总体中的特征函数关于 $P(x,y)$的数学期望：

$$E_p(f)=\sum_{x,y}P(x,y)f(x,y)=\sum_{x,y}P(x)P(y\mid x)f(x,y)=\sum_{x,y}\widetilde{P}(x)P(y\mid x)f(x,y)$$

根据大数定律，总体中特征函数的数学期望=样本中特征函数的数学期望

$$\sum_{x,y}\tilde{P}(x)P(y|x)f(x,y)=\sum_{x,y}\tilde{P}(x,y)f(x,y)$$

作为模型学习的约束条件

2 最大熵模型

条件熵：

$$\mathrm{H}(Y|X)=\sum_{i=1}^{n}p\ (x_i)H(Y|X=x_i)=-\sum_{x\in X}p(x)\sum_{y\in Y}p(y|x)\log p(y|x)=-\sum_{x\in X}\sum_{y\in Y}p(x,y)\log p(y|x)$$

$$p(x_i)=p(X=x_i),\quad \mathrm{i}=1,2,..,\mathrm{n}$$

最优化问题：

$$\max_{P\in C} H(P)=-\sum_{x,y}\widetilde{P}(x)P(y|x)\log P(y|x)$$ 最大熵

$$s.t.\quad E_P(f_i)=E_{\widetilde{P}}(f_i)\quad i=1,2,\ldots,\mathrm{n}$$ n个约束条件

$$\sum_{y}P(y|x)=1$$ 给定x，它所属所有的类别的条件概率之和为1

2 最大熵模型

最大熵模型的学习：

$$\max_{P\in C} H(P) = -\sum_{x,y} \widetilde{P}(x)P(y|x)\log P(y|x)$$

$$s.t.\quad E_P(f_i) = E_{\widetilde{P}}(f_i) \quad i = 1,2,\dots,n$$

$$\sum_y P(y|x) = 1$$

n个约束条件

$$\min_{P\in C} -H(P) = \sum_{x,y} \widetilde{P}(x)P(y|x)\log P(y|x)$$

$$s.t.\quad E_P(f_i) - E_{\widetilde{P}}(f_i) = 0 \quad i = 1,2,\dots,n$$

$$\sum_y P(y|x) = 1$$

首先，引进拉格朗日乘子 $w_0, w_1, w_2, \cdots, w_n$，定义拉格朗日函数 $L(P, w)$：

$$L(P,w) \equiv -H(P) + w_0\left(1-\sum_y P(y|x)\right) + \sum_{i=1}^{n} w_i(E_{\tilde{P}}(f_i) - E_P(f_i))$$

$$= \sum_{x,y} \tilde{P}(x)P(y|x)\log P(y|x) + w_0\left(1-\sum_y P(y|x)\right) +$$

$$\sum_{i=1}^{n} w_i\left(\sum_{x,y}\tilde{P}(x,y)f_i(x,y) - \sum_{x,y}\tilde{P}(x)P(y|x)f_i(x,y)\right) \qquad (6.17)$$

最优化的原始问题是

$$\min_{P\in C}\max_{w} L(P,w)$$

对偶问题是

$$\max_{w}\min_{P\in C} L(P,w)$$

p*=d*=L（x*，α *，β *）

2 最大熵模型

最大熵模型的学习：

首先，求解对偶问题 (6.19) 内部的极小化问题 $\min_{P \in \mathbf{C}} L(P, w)$。$\min_{P \in \mathbf{C}} L(P, w)$ 是 w 的函数，将其记作

$$\Psi(w) = \min_{P \in \mathbf{C}} L(P, w) = L(P_w, w) \tag{6.20}$$

$\Psi(w)$ 称为对偶函数。同时，将其解记作

$$P_w = \arg \min_{P \in \mathbf{C}} L(P, w) = P_w(y|x) \tag{6.21}$$

具体地，求 $L(P, w)$ 对 $P(y|x)$ 的偏导数

$$\begin{aligned}
\frac{\partial L(P, w)}{\partial P(y|x)} &= \sum_{x,y} \tilde{P}(x) (\log P(y|x) + 1) - \sum_{y} w_0 - \sum_{x,y} \left(\tilde{P}(x) \sum_{i=1}^{n} w_i f_i(x, y) \right) \\
&= \sum_{x,y} \tilde{P}(x) \left(\log P(y|x) + 1 - w_0 - \sum_{i=1}^{n} w_i f_i(x, y) \right)
\end{aligned}$$

令偏导数为0，在$\tilde{P}(x) > 0$ 的情况下，解得

$$P(y|x) = \exp\left(\sum_{i=1}^{n} w_i f_i(x, y) + w_0 - 1 \right) = \frac{\exp\left(\sum_{i=1}^{n} w_i f_i(x, y) \right)}{\exp(1 - w_0)}$$

2 最大熵模型

最大熵模型的学习：

$$\sum_y P(y\mid x)=1\text{即}\sum_y \frac{\exp(\sum_{i=1}^{n}\omega_i f_i(x,y))}{\exp(1-\omega_0)}=1$$

$$P(y|x)=\exp\left(\sum_{i=1}^{n} w_i f_i(x,y)+w_0-1\right)=\frac{\exp\left(\sum_{i=1}^{n} w_i f_i(x,y)\right)}{\exp(1-w_0)}$$

$$\exp(1-\omega_0)=\sum_y \exp(\sum_{i=1}^{n}\omega_i f_i(x,y))$$

$$P_w(y|x)=\frac{1}{Z_w(x)}\exp\left(\sum_{i=1}^{n} w_i f_i(x,y)\right) \tag{6.22}$$

其中，

$$Z_w(x)=\sum_y \exp\left(\sum_{i=1}^{n} w_i f_i(x,y)\right) \tag{6.23}$$

$Z_w(x)$ 称为规范化因子；$f_i(x,y)$ 是特征函数；w_i 是特征的权值。由式 (6.22)、式 (6.23) 表示的模型 $P_w=P_w(y|x)$ 就是最大熵模型。这里，w 是最大熵模型中的参数向量。

之后，求解对偶问题外部的极大化问题

$$\max_w \Psi(w)$$

将其解记为 w^*，即

$$w^*=\arg\max_w \Psi(w)$$

$$\mathrm{P}^*=P_{\omega^*}=P_{\omega^*}(y\mid x)$$

原始问题的解是对偶问题的解，最大熵模型的学习归结为对偶函数的最大化

2 最大熵模型

最大熵例子：

例 6.1 假设随机变量 X 有 5 个取值 $\{A,B,C,D,E\}$，要估计取各个值的概率 $P(A),P(B),P(C),P(D),P(E)$。

最优化问题

$$\begin{aligned}
\min \quad & -H(P)=\sum_{i=1}^{5} P(y_i)\log P(y_i) \\
\text{s.t.} \quad & P(y_1)+P(y_2)=\tilde{P}(y_1)+\tilde{P}(y_2)=\frac{3}{10} \\
& \sum_{i=1}^{5} P(y_i)=\sum_{i=1}^{5} \tilde{P}(y_i)=1
\end{aligned}$$

定义拉格朗日函数

$$L(P,w)=\sum_{i=1}^{5} P(y_i)\log P(y_i)+w_1\left(P(y_1)+P(y_2)-\frac{3}{10}\right)+w_0\left(\sum_{i=1}^{5} P(y_i)-1\right)$$

对偶问题

根据拉格朗日对偶性，可以通过求解对偶最优化问题得到原始最优化问题的解，所以求解

$$\max_{w}\min_{P} L(P,w)$$

2 最大熵模型

首先求解 $L(P,w)$ 关于 P 的极小化问题。为此，固定 w_0，w_1，求偏导数：

$$\frac{\partial L(P,w)}{\partial P(y_1)} = 1 + \log P(y_1) + w_1 + w_0$$

$$\frac{\partial L(P,w)}{\partial P(y_2)} = 1 + \log P(y_2) + w_1 + w_0$$

$$\frac{\partial L(P,w)}{\partial P(y_3)} = 1 + \log P(y_3) + w_0$$

$$\frac{\partial L(P,w)}{\partial P(y_4)} = 1 + \log P(y_4) + w_0$$

$$\frac{\partial L(P,w)}{\partial P(y_5)} = 1 + \log P(y_5) + w_0$$

令各偏导数等于 0，解得

$$P(y_1) = P(y_2) = \mathrm{e}^{-w_1-w_0-1}$$

$$P(y_3) = P(y_4) = P(y_5) = \mathrm{e}^{-w_0-1}$$

于是，

$$\min_P L(P,w) = L(P_w,w) = -2\mathrm{e}^{-w_1-w_0-1} - 3\mathrm{e}^{-w_0-1} - \frac{3}{10}w_1 - w_0$$

再求解 $L(P_w,w)$ 关于 w 的极大化问题：

$$\max_w L(P_w,w) = -2\mathrm{e}^{-w_1-w_0-1} - 3\mathrm{e}^{-w_0-1} - \frac{3}{10}w_1 - w_0$$

分别求 $L(P_w,w)$ 对 w_0，w_1 的偏导数并令其为 0，得到

$$\mathrm{e}^{-w_1-w_0-1} = \frac{3}{20}$$

$$\mathrm{e}^{-w_0-1} = \frac{7}{30}$$

于是得到所要求的概率分布为

$$P(y_1) = P(y_2) = \frac{3}{20}$$

$$P(y_3) = P(y_4) = P(y_5) = \frac{7}{30}$$

证明对偶函数的极大化等价于最大熵模型的极大似然估计：

对数似然函数： $\mathrm{L}(x_1,x_2,\ldots,x_n;\theta)=\prod_{i=1}^{n}p(x_i;\theta)$

设X的取值有k个，分别用v_1, v_2, . . . , v_k表示。C(X=v_i)表示观测值中样本v_i出现的频数。

$$\mathrm{L}(x_1,x_2,\ldots,x_n;\theta)=\prod_{i=1}^{n}p(\mathrm{v}_i;\theta)^{C(X=v_i)}$$

L()

经验概率

$$\tilde{\mathrm{p}}(x)=\frac{C(X=v_i)}{n}$$

两边取1/n次方： $\mathrm{L}(x_1,x_2,\ldots,x_n;\theta)^{\frac{1}{\mathrm{n}}}=\prod_{i=1}^{k}p(\mathrm{v}_i;\theta)^{\frac{C(X=v_i)}{n}}$

$$\mathrm{L}(x_1,x_2,\ldots,x_n;\theta)^{\frac{1}{\mathrm{n}}}=\prod_{i=1}^{\mathrm{k}}p(\mathrm{v}_i;\theta)^{\mathrm{p}(X=v_i)}=\prod_{i=1}^{\mathrm{n}}p(\mathrm{x}_i;\theta)^{\mathrm{p}(x_i)}$$

取1/n不影响最大值：

$$\mathrm{L}(x_1,x_2,\ldots,x_n;\theta)=\prod_{\mathrm{x}}p(x)^{\tilde{p}(x)}$$

最大熵模型的极大似然估计：

$$L(x_1,x_2,\ldots,x_n;\theta)=\prod_{x} p(x)^{\tilde{p}(x)}$$

$$P_w(y|x)=\frac{1}{Z_w(x)}\exp\left(\sum_{i=1}^{n} w_i f_i(x,y)\right) \tag{6.22}$$

$$Z_w(x)=\sum_{y}\exp\left(\sum_{i=1}^{n} w_i f_i(x,y)\right) \tag{6.23}$$

已知训练数据的经验概率分布 $\tilde{P}(X,Y)$，条件概率分布 $P(Y|X)$ 的对数似然函数表示为

$$L_{\tilde{P}}(P_w)=\log\prod_{x,y} P(y|x)^{\tilde{P}(x,y)}=\sum_{x,y}\tilde{P}(x,y)\log P(y|x)$$

当条件概率分布 $P(y|x)$ 是最大熵模型 (6.22) 和 (6.23) 时，对数似然函数 $L_{\tilde{P}}(P_w)$ 为

$$\sum_{y} P(y|x)=1$$

$$\begin{aligned} L_{\tilde{P}}(P_w) &= \sum_{x,y}\tilde{P}(x,y)\log P(y|x) \\ &= \sum_{x,y}\tilde{P}(x,y)\sum_{i=1}^{n} w_i f_i(x,y)-\sum_{x,y}\tilde{P}(x,y)\log Z_w(x) \\ &= \sum_{x,y}\tilde{P}(x,y)\sum_{i=1}^{n} w_i f_i(x,y)-\sum_{x}\tilde{P}(x)\log Z_w(x) \end{aligned} \tag{6.26}$$

对偶函数的极大化：

再看对偶函数 $\Psi(w)$。由式 (6.17) 及式 (6.20) 可得

$$\begin{aligned} \Psi(w) &= \sum_{x,y}\tilde{P}(x)P_w(y|x)\log P_w(y|x)+ \\ &\quad \sum_{i=1}^{n} w_i\left(\sum_{x,y}\tilde{P}(x,y)f_i(x,y)-\sum_{x,y}\tilde{P}(x)P_w(y|x)f_i(x,y)\right) \\ &= \sum_{x,y}\tilde{P}(x,y)\sum_{i=1}^{n} w_i f_i(x,y)+\sum_{x,y}\tilde{P}(x)P_w(y|x)\left(\log P_w(y|x)-\sum_{i=1}^{n} w_i f_i(x,y)\right) \\ &= \sum_{x,y}\tilde{P}(x,y)\sum_{i=1}^{n} w_i f_i(x,y)-\sum_{x,y}\tilde{P}(x)P_w(y|x)\log Z_w(x) \\ &= \sum_{x,y}\tilde{P}(x,y)\sum_{i=1}^{n} w_i f_i(x,y)-\sum_{x}\tilde{P}(x)\log Z_w(x) \end{aligned} \tag{6.27}$$

$$\sum_{y} P(y|x)=1$$

$$\Psi(w)=L_{\tilde{P}}(P_w)$$

2 最大熵模型

最大熵模型和逻辑回归的联系：

数据集 $\vec{X} = (x_1, x_2, \ldots, x_n)^T$ Y={0, 1}

特征函数：$f_i(x,y) = \begin{cases} x_i & y=1 \\ 0 & y=0 \end{cases}$

最大熵：

$$P_w(y|x) = \frac{1}{Z_w(x)} \exp\left(\sum_{i=1}^{n} w_i f_i(x,y)\right)$$

$$Z_w(x) = \sum_y \exp\left(\sum_{i=1}^{n} w_i f_i(x,y)\right)$$

$$Z_\omega(x) = \sum_y \exp(\sum_{i=1}^{n} \omega_i f_i(x,y)) = \exp(\sum_{i=1}^{n} \omega_i f_i(x, y=0)) + \exp(\sum_{i=1}^{n} \omega_i f_i(x, y=1)) = 1 + \exp(\vec{\omega}\vec{x})$$

$$P_\omega(y=1|x) = \frac{\exp(\vec{\omega}\vec{x})}{1+\exp(\vec{\omega}\vec{x})}$$

$$P_\omega(y=0|x) = \frac{1}{1+\exp(\vec{\omega}\vec{x})}$$

逻辑回归本质上说仍为最大熵

C

拉格朗日对偶性

3 拉格朗日对偶性

原始问题： $\min_{x \in R^n} f(x)$ 优化问题（C. 1）

优化变量x s.t. $c_i(x) \le 0, \quad i=1,2,\dots,k$ 不等式约束(C. 2) $\alpha_1, \alpha_2, \dots, \alpha_k$

$h_j(x)=0, \quad j=1,2,\dots,l$ 等式约束(C. 3) $\beta_1, \beta_2, \dots, \beta_l$

α 是k维向量

$$L(x,\alpha,\beta)=f(x)+\sum_{i=1}^{k}\alpha_i c_i(x)+\sum_{j=1}^{l}\beta_j h_j(x) \qquad (C.4)$$

其中$x=(x^{(1)},x^{(2)},\dots,x^{(n)})^T \in R^n$, α_i, β_j是拉格朗日乘子，$\alpha_i \ge 0$

考虑： $\theta_P(x)=\max_{\alpha,\beta:\alpha_i\ge 0} L(x,\alpha,\beta)=\max_{\alpha,\beta:\alpha_i\ge 0}[f(x)+\sum_{i=1}^{k}\alpha_i c_i(x)+\sum_{j=1}^{l}\beta_j h_j(x)]$

当$c_i(x)>0$或者$h_j(x)\ne 0, \theta_p(x)=\infty$

满足约束，由于取max $\sum_{i=1}^{k}\alpha_i c_i(x)=0$ ，$\sum_{j=1}^{l}\beta_j h_j(x)=0$

$$\theta_P(x)=\begin{cases} f(x), & x \text{ 满足原始问题约束} \\ +\infty, & \text{其他} \end{cases}$$

3 拉格朗日对偶性

原始问题：

$$\min_{x \in R^n} f(x)$$

$$\text{s.t.} \quad c_i(x) \le 0, \quad i = 1, 2, \ldots, k$$

$$h_j(x) = 0, \quad j = 1, 2, \ldots, l$$

有共同解

极小极大问题：

$$\min_x \theta_P(x) = \min_x \max_{\alpha, \beta: \alpha_i \ge 0} L(x, \alpha, \beta)$$

$$\theta_P(x) = \begin{cases} f(x), & x \text{ 满足原始问题约束} \\ +\infty, & \text{其他} \end{cases}$$

$$P^* = \min_x \theta_P(x) = \min_x \max_{\alpha, \beta: \alpha_i \ge 0} L(x, \alpha, \beta)$$

一般是原始问题不好求解，选择它的对偶问题求解。

3 拉格朗日对偶性

对偶问题：

定义

$$\theta_D(\alpha,\beta)=\min_x L(x,\alpha,\beta) \tag{C.10}$$

再考虑极大化 $\theta_D(\alpha,\beta)=\min\limits_x L(x,\alpha,\beta)$，即

$$\max_{\alpha,\beta:\alpha_i\geqslant 0}\theta_D(\alpha,\beta)=\max_{\alpha,\beta:\alpha_i\geqslant 0}\min_x L(x,\alpha,\beta) \tag{C.11}$$

对偶问题的最优值：

$$d^*=\max_{\alpha,\beta:\alpha_i\geqslant 0}\theta_D(\alpha,\beta)$$

3 拉格朗日对偶性

原始问题和对偶问题：
（弱对偶性）

定理 C.1 若原始问题和对偶问题都有最优值，则

$$d^* = \max_{\alpha,\beta:\alpha_i\geqslant 0} \min_x L(x,\alpha,\beta) \leqslant \min_x \max_{\alpha,\beta:\alpha_i\geqslant 0} L(x,\alpha,\beta) = p^* \tag{C.15}$$

证明 由式 (C.12) 和式 (C.5)，对任意的 α,β 和 x，有

$$\theta_D(\alpha,\beta) = \min_x L(x,\alpha,\beta) \leqslant L(x,\alpha,\beta) \leqslant \max_{\alpha,\beta:\alpha_i\geqslant 0} L(x,\alpha,\beta) = \theta_P(x) \tag{C.16}$$

即

$$\theta_D(\alpha,\beta) \leqslant \theta_P(x) \tag{C.17}$$

由于原始问题和对偶问题均有最优值，所以，

$$\max_{\alpha,\beta:\alpha_i\geqslant 0} \theta_D(\alpha,\beta) \leqslant \min_x \theta_P(x) \tag{C.18}$$

即

$$d^* = \max_{\alpha,\beta:\alpha_i\geqslant 0} \min_x L(x,\alpha,\beta) \leqslant \min_x \max_{\alpha,\beta:\alpha_i\geqslant 0} L(x,\alpha,\beta) = p^* \tag{C.19}$$

这个定理说明了对偶问题的最优值是原始问题最优值的下界。

3 拉格朗日对偶性

推论 C.1 设 x^* 和 α^*, β^* 分别是原始问题 (C.1)~(C.3) 和对偶问题 (C.12)~(C.13) 的可行解，并且 $d^* = p^*$，则 x^* 和 α^*, β^* 分别是原始问题和对偶问题的最优解。

在某些条件下，原始问题和对偶问题的最优值相等，$d^* = p^*$。这时可以用解对偶问题替代解原始问题。下面以定理的形式叙述有关的重要结论而不予证明。

强对偶性-slater条件：

定理 C.2 考虑原始问题 (C.1)~(C.3) 和对偶问题 (C.12)~(C.13)。假设函数 $f(x)$ 和 $c_i(x)$ 是凸函数，$h_j(x)$ 是仿射函数；并且假设不等式约束 $c_i(x)$ 是严格可行的，即存在 x，对所有 i 有 $c_i(x) < 0$，则存在 x^*, α^*, β^*，使 x^* 是原始问题的解，α^*, β^* 是对偶问题的解，并且

$$p^* = d^* = L(x^*, \alpha^*, \beta^*) \tag{C.20}$$

原始问题L是凸优化问题：可行域是凸集；L是凸函数

slater条件：凸集中有内点（不是边界上的点）

强对偶性：

p*=d*=L(x*，α*，β*)

3 拉格朗日对偶性

强对偶性-KKT条件：

定理 C.3 对原始问题 (C.1)~(C.3) 和对偶问题 (C.12)~(C.13)，假设函数 $f(x)$ 和 $c_i(x)$ 是凸函数，$h_j(x)$ 是仿射函数，并且不等式约束 $c_i(x)$ 是严格可行的，则 x^* 和 α^*, β^* 分别是原始问题和对偶问题的解的充分必要条件是 x^*, α^*, β^* 满足下面的 Karush-Kuhn-Tucker (KKT) 条件：

拉格朗日取得可行解必要条件

$$\nabla_x L(x^*, \alpha^*, \beta^*) = 0 \tag{C.21}$$

对偶互补条件

$$\alpha_i^* c_i(x^*) = 0, \quad i = 1, 2, \cdots, k \tag{C.22}$$

原始问题约束条件

$$c_i(x^*) \leqslant 0, \quad i = 1, 2, \cdots, k \tag{C.23}$$

不等式约束中需满足条件

$$\alpha_i^* \geqslant 0, \quad i = 1, 2, \cdots, k \tag{C.24}$$

原始问题约束条件

$$h_j(x^*) = 0 \quad j = 1, 2, \cdots, l \tag{C.25}$$

特别指出，式 (C.22) 称为 KKT 的对偶互补条件。由此条件可知：若 $\alpha_i^* > 0$，则 $c_i(x^*) = 0$。

逻辑斯谛回归回顾及实例

1 逻辑回归形式

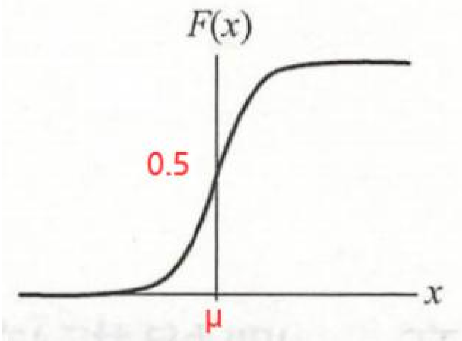
两种回归形式

线性回归：$P = h(x) = \omega_0 + \omega_1 x_1 + \omega_2 x_2 + \ldots + \omega_n x_n = \vec{\omega}^T \cdot \mathrm{X}$

逻辑回归：

$$P = h(x) = \mathrm{g}(\vec{\omega}^T X)$$

$$\mathrm{g}(x) = \frac{1}{1+e^{-x}}$$



二项逻辑回归解决线性回归问题所作的变换：

设 $P(Y=1 \mid x) = p \quad P(Y=0 \mid x) = 1-p$

$$\operatorname{logit}(p) = \log\frac{p}{1-p} = \vec{\omega} \cdot \vec{x}$$

$$P(Y=1|x) = \frac{\exp(w \cdot x)}{1+\exp(w \cdot x)}$$

$$P(Y=0|x) = \frac{1}{1+\exp(w \cdot x)}$$

二项逻辑斯谛回归模型参数ω估计：

$$T=\{(x_1,y_1),(x_2,y_2),\ldots,(x_N,y_N)\},\text{其中}x_i\in R^n,y_i\in\{0,1\}$$

设 $P(Y=1\mid x)=\pi(x)\quad P(Y=0\mid x)=1-\pi(x)$

概率密度函数 $P(Y=y_i\mid x=x_i;\omega)=[\pi(x_i)]^{y_i}[1-\pi(x_i)]^{1-y_i}$

样本数据独立，似然函数 $P(Y=y_i\mid x=x_i;\omega)=\prod_{i=1}^{N}[\pi(x_i)]^{y_i}[1-\pi(x_i)]^{1-y_i}$

对数似然函数为

$$\begin{aligned}L(w)&=\sum_{i=1}^{N}[y_i\log\pi(x_i)+(1-y_i)\log(1-\pi(x_i))]\\&=\sum_{i=1}^{N}\left[y_i\log\frac{\pi(x_i)}{1-\pi(x_i)}+\log(1-\pi(x_i))\right]\\&=\sum_{i=1}^{N}[y_i(w\cdot x_i)-\log(1+\exp(w\cdot x_i))]\end{aligned}$$

$$P(Y=1|x)=\frac{\exp(w\cdot x)}{1+\exp(w\cdot x)}$$

π(x)

$$P(Y=0|x)=\frac{1}{1+\exp(w\cdot x)}$$

$$\omega^*=\arg\max_{\omega}L(\omega)=-\arg\min_{\omega}L(\omega)$$

2 逻辑斯谛参数估计

逻辑斯谛回归通常采用梯度下降法或拟牛顿法学习参数ω

梯度下降法：

$$L(\omega)=\sum_{i=1}^{N}[y_i(w\cdot x_i)-\log(1+\exp(w\cdot x_i))]$$

对ω求导 ⟹

$$\nabla L(\omega)=\sum_{i=1}^{N}[x_i y_i-\frac{e^{\omega x_i}}{1+e^{\omega x_i}}\cdot x_i]$$

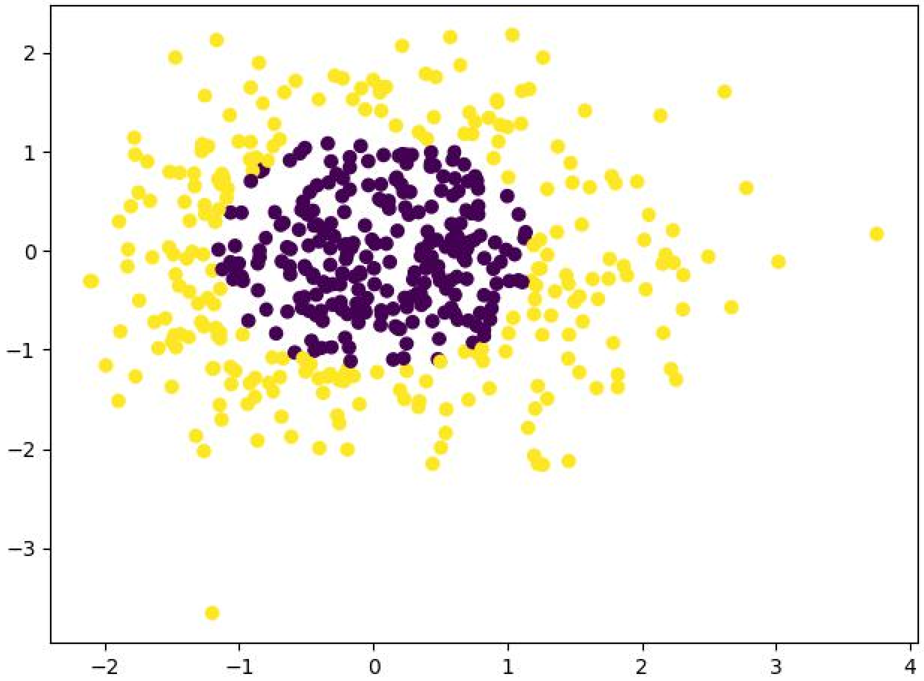
$$\omega=\omega-\alpha\nabla L(\omega)$$

$$\omega_j=\omega_j-\alpha\frac{\partial L}{\partial \omega_j}$$

公式里的ω是向量，ω的第j个分量用ω_j表示，每个分量都同时更新。

3 多项式逻辑回归实例

```
"""
make_gaussian-quantiles: 将一个单高斯分布的点集划分为两个数量均等的点集，作为两类
                         总共500个样本，每类250个样本，一起2个样本特征
"""
x_data, y_data = make_gaussian_quantiles(n_samples=500, n_features=2, n_classes=2)
plt.scatter(x_data[:, 0], x_data[:, 1], c=y_data)
plt.show()
```



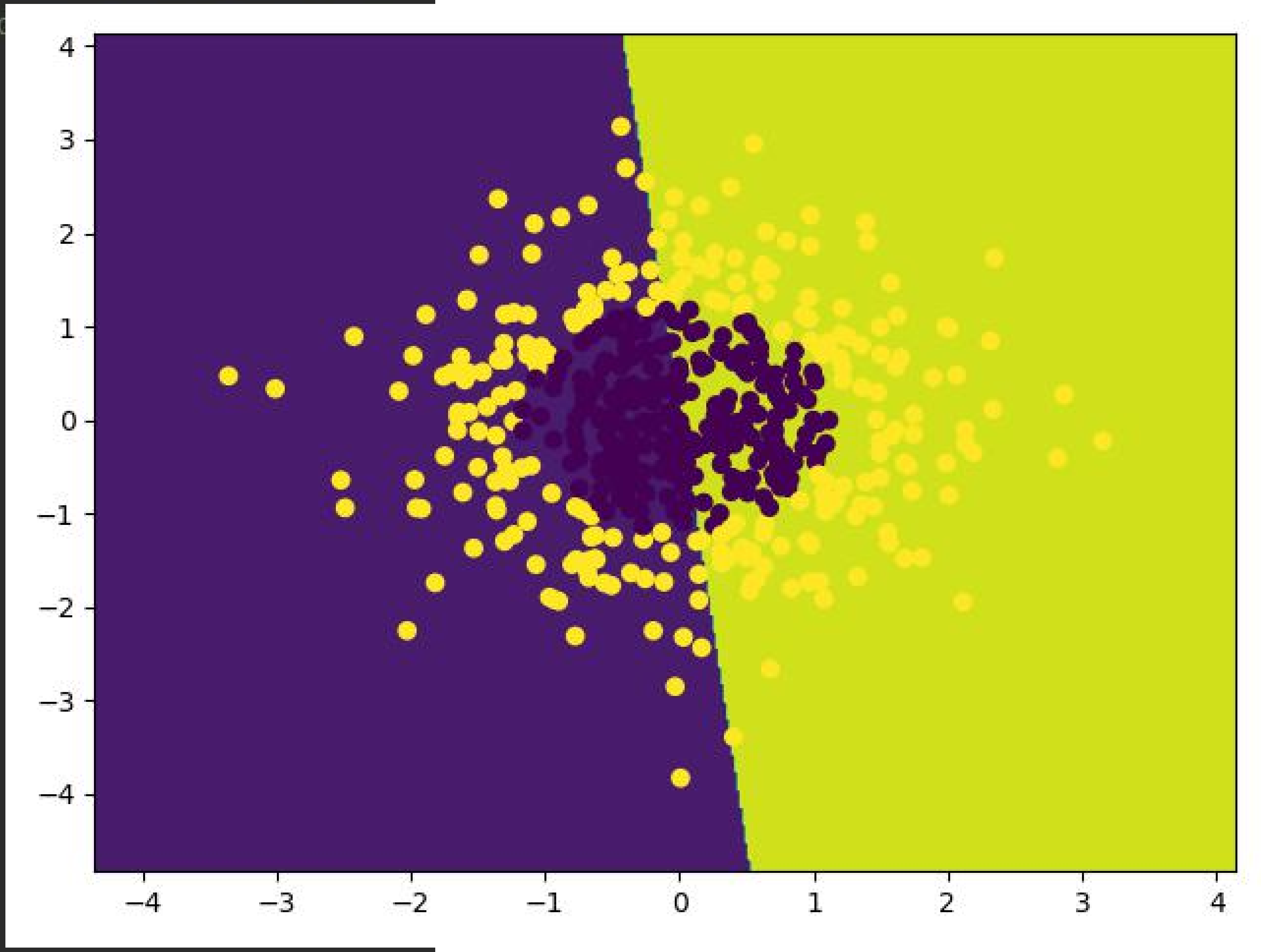
$$P = h(x) = \omega_0 + \omega_1 x_1 + \omega_2 x_2 + \ldots + \omega_n x_n$$

$$P = h(x) = \omega_0 + \omega_1 x^1 + \omega_2 x^2 + \ldots + \omega_n x^n$$

$$P = h(x) = \omega_0 + \omega_1 x_1 + \omega_2 x_2 + \omega_3 x_1^{\,2} + \omega_4 x_2^{\,2} + \omega_5 x_1 x_2$$

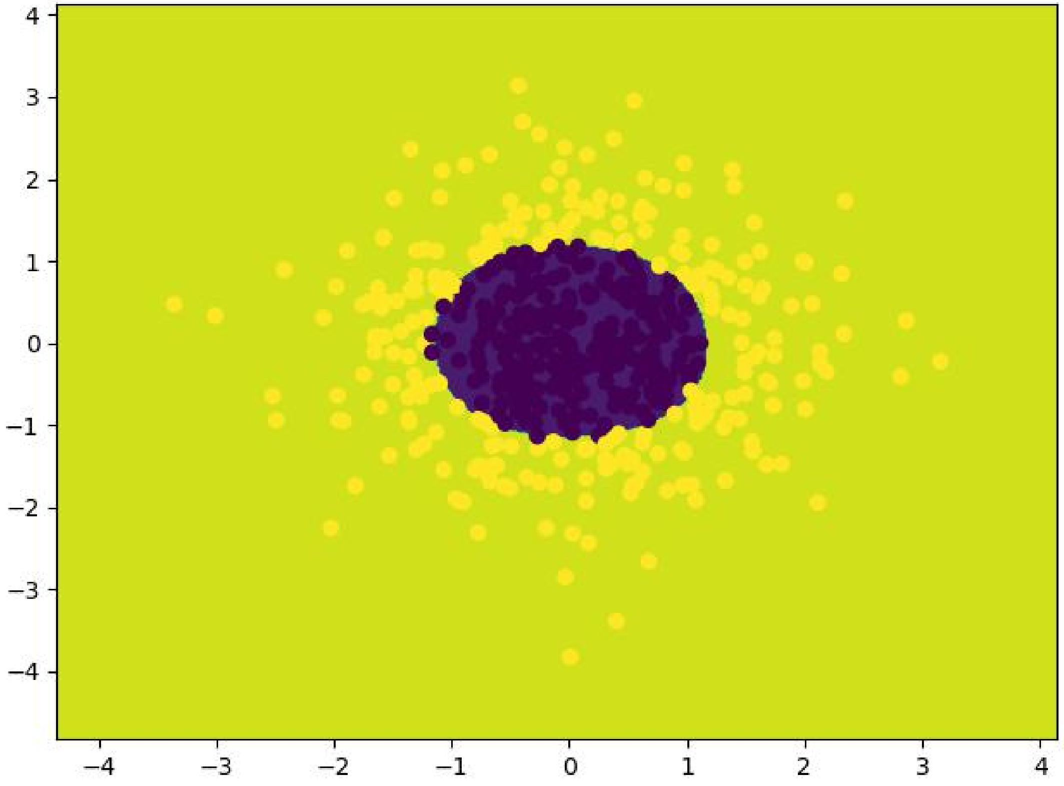
3 多项式逻辑回归实例

```
"""
solver:是逻辑回归选择的优化算法
        {'newton-cg', 'lbfgs', 'liblinear', 'sag', 'saga'}, optic

先看未定义多项式回归的情况
"""
logistic = LogisticRegression(solver='liblinear')
logistic.fit(x_data, y_data)
# 获取数据值所在的范围
x_min, x_max = x_data[:, 0].min() - 1, x_data[:, 0].max() + 1
y_min, y_max = x_data[:, 1].min() - 1, x_data[:, 1].max() + 1
# 生成网格矩阵
xx, yy = np.meshgrid(np.arange(x_min, x_max, 0.02),
                     np.arange(y_min, y_max, 0.02))

z = logistic.predict(np.c_[xx.ravel(), yy.ravel()])
z = z.reshape(xx.shape)
# 等高线图
cs = plt.contourf(xx, yy, z)
# 样本散点图
plt.scatter(x_data[:, 0], x_data[:, 1], c=y_data)
plt.show()
print('未定义多项式回归 score:', logistic.score(x_data, y_data))
```



3 多项式逻辑回归实例

```python
"""
定义多项式回归
"""
# 定义多项式回归,degree的值可以调节多项式的特征
poly_reg = PolynomialFeatures(degree=5)
# 特征处理
x_poly = poly_reg.fit_transform(x_data)
# 定义逻辑回归模型
logistic = LogisticRegression(solver='liblinear')
# 训练模型
logistic.fit(x_poly, y_data)
# 获取数据值所在的范围
x_min, x_max = x_data[:, 0].min() - 1, x_data[:, 0].max() + 1
y_min, y_max = x_data[:, 1].min() - 1, x_data[:, 1].max() + 1
# 生成网格矩阵
xx, yy = np.meshgrid(np.arange(x_min, x_max, 0.02),
                     np.arange(y_min, y_max, 0.02))

z = logistic.predict(poly_reg.fit_transform(np.c_[xx.ravel(), yy.ravel()]))
z = z.reshape(xx.shape)
# 等高线图
cs = plt.contourf(xx, yy, z)
# 样本散点图
plt.scatter(x_data[:, 0], x_data[:, 1], c=y_data)
plt.show()
print('多项式回归 score', logistic.score(x_poly, y_data))
```




```
未定义多项式回归 score: 0.524
多项式回归 score 0.986
```

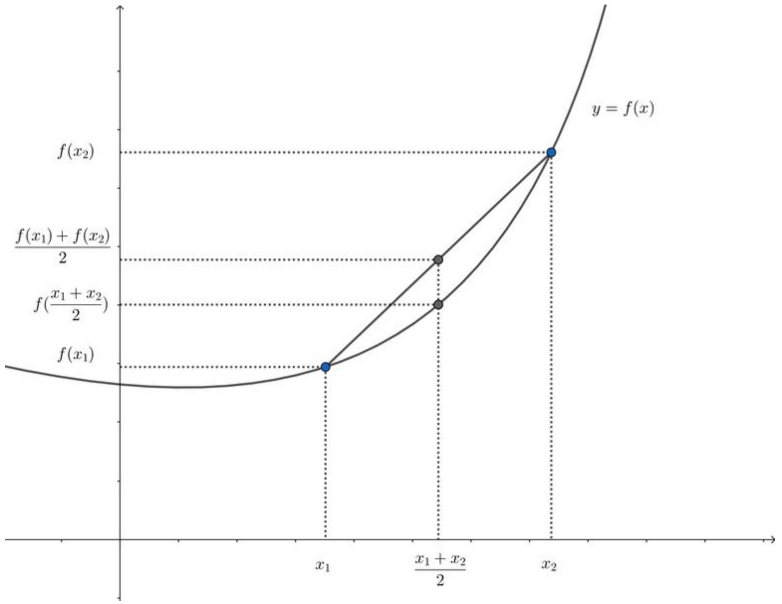
E

模型学习的最优化算法

1 导读

逻辑斯谛回归模型、最大熵模型学习归结为似然函数为目标函数的最优化问题，通常通过迭代算法求解。从最优化的观点看，这时的目标函数具有很好的性质。它是光滑的凸函数，因此多种最优化的方法都适用，保证能找到全局最优解。

常用的方法有改进的迭代尺度法、梯度下降法、牛顿法或拟牛顿法。牛顿法或拟牛顿法一般的收敛速度更快。



严格凸函数：$f(\frac{x_1+x_2}{2}) < \frac{f(x_1)+f(x_2)}{2}$

若对于任意的x, y, z, 其中满足下式，则f是几乎凸的

$$f(z) \le \max\{f(x), f(y)\}, \ \forall x, y, z \ \ x \le z \le y$$

2 改进的迭代尺度法

已知最大熵模型为

$$P_w(y|x) = \frac{1}{Z_w(x)} \exp\left(\sum_{i=1}^{n} w_i f_i(x,y)\right)$$

其中，

$$Z_w(x) = \sum_y \exp\left(\sum_{i=1}^{n} w_i f_i(x,y)\right)$$

对数似然函数为

$$\uparrow \quad L(w) = \sum_{x,y} \tilde{P}(x,y) \sum_{i=1}^{n} w_i f_i(x,y) - \sum_x \tilde{P}(x) \log Z_w(x)$$

目标是通过极大似然估计学习模型参数，即求对数似然函数的极大值 $\hat{w}$。

给定ω初值，不断更新ω使得L(ω)达到最大值

给个ω的增量，作差，使得差大于0直到为0：

最大熵模型当前的参数向量是 $w = (w_1, w_2, \cdots, w_n)^{\mathrm{T}}$

新的参数向量 $w + \delta = (w_1 + \delta_1, w_2 + \delta_2, \cdots, w_n + \delta_n)^{\mathrm{T}}$

$$\underbrace{L(w+\delta) - L(w)}_{\text{大于0直到为0}} = \sum_{x,y} \tilde{P}(x,y) \log P_{w+\delta}(y|x) - \sum_{x,y} \tilde{P}(x,y) \log P_w(y|x)$$

$$= \sum_{x,y} \tilde{P}(x,y) \sum_{i=1}^{n} \delta_i f_i(x,y) - \sum_x \tilde{P}(x) \log \frac{Z_{w+\delta}(x)}{Z_w(x)}$$

2 改进的迭代尺度法

利用不等式 $-\log\alpha \geqslant 1-\alpha, \quad \alpha > 0$

$$-\log\frac{Z_{\omega+\sigma}(x)}{Z_\omega(x)} \geq 1-\frac{Z_{\omega+\sigma}(x)}{Z_\omega(x)}$$

$$-\sum_x \widetilde{P}(x)\log\frac{Z_{\omega+\sigma}(x)}{Z_\omega(x)} \geq \sum_x \widetilde{P}(x) - \sum_x \widetilde{P}(x)\frac{Z_{\omega+\sigma}(x)}{Z_\omega(x)}$$

=1

原来式子：

$$\sum_{x,y}\tilde{P}(x,y)\sum_{i=1}^n \delta_i f_i(x,y) - \sum_x \tilde{P}(x)\log\frac{Z_{w+\delta}(x)}{Z_w(x)}$$

建立对数似然函数改变量的下界：

$$L(w+\delta)-L(w) \geqslant \sum_{x,y}\tilde{P}(x,y)\sum_{i=1}^n \delta_i f_i(x,y)+1-\sum_x \tilde{P}(x)\frac{Z_{w+\delta}(x)}{Z_w(x)}$$

$$= \sum_{x,y}\tilde{P}(x,y)\sum_{i=1}^n \delta_i f_i(x,y)+1-\sum_x \tilde{P}(x)\sum_y P_w(y|x)\exp\sum_{i=1}^n \delta_i f_i(x,y)$$

$$\frac{Z_{\omega+\sigma}(x)}{Z_\omega(x)} = \frac{\sum_y \exp(\sum_{i=1}^n(\omega_i+\sigma_i)f_i)}{Z_\omega(x)}$$

$$= \frac{\sum_y\left[\exp(\sum_{i=1}^n \omega_i f_i) * \exp(\sum_{i=1}^n \sigma_i f_i)\right]}{Z_\omega(x)}$$

$$= \sum_y\left[P_\omega(y|x)\exp(\sum_{i=1}^n \sigma_i f_i)\right]$$

$$P_w(y|x) = \frac{1}{Z_w(x)}\exp\left(\sum_{i=1}^n w_i f_i(x,y)\right)$$

2 改进的迭代尺度法

将右端记为

$$A(\delta|w)=\sum_{x,y}\tilde{P}(x,y)\sum_{i=1}^{n}\delta_i f_i(x,y)+1-\sum_x\tilde{P}(x)\sum_y P_w(y|x)\exp\sum_{i=1}^{n}\delta_i f_i(x,y)$$

$$L(w+\delta)-L(w)\geqslant A(\delta|w)$$

即 $A(\delta|w)$ 是对数似然函数改变量的一个下界。

我们想通过找到δ，使得下界尽可能大。δ是一个向量，每次对δ一个分量 δ_i 求偏导，但是指数函数求导会保留其他的 δ_j 分量。所以我们希望可以指数函数中只保留 δ_i 保不含其他分量 δ_j。

琴生(Jensen)不等式：

$\{x_1, x_2, \ldots, x_n\}$

$\varphi(a)$是凸函数

$\sum_{i=1}^{n}\lambda_i=1$

$$\varphi\left(\sum_{i=1}^{n}g(x_i)\lambda_i\right)\le\sum_{i=1}^{n}\varphi(g(x_i))\lambda_i$$

$\varphi(a)$内部的诸多 $g(x_i)$ 变为只有一个 $g(x_i)$

2 改进的迭代尺度法

将右端记为

$$A(\delta\,|w)=\sum_{x,y}\tilde{P}(x,y)\sum_{i=1}^{n}\delta_i f_i(x,y)+1-\sum_{x}\tilde{P}(x)\sum_{y}P_w(y|x)\exp\sum_{i=1}^{n}\delta_i f_i(x,y)$$

$$\varphi\left(\sum_{i=1}^{n}g(x_i)\lambda_i\right)\leq\sum_{i=1}^{n}\varphi(g(x_i))\lambda_i \qquad f_{\#}=\sum_{i}f_i$$

$$\exp(\sum_{i=1}^{n}\delta_i f_i)=\exp(\sum_{i=1}^{n}\frac{f_i}{f_{\#}}\cdot\delta_i f_{\#})\leq\sum_{i=1}^{n}\frac{f_i}{f_{\#}}\cdot\exp(\delta_i f_{\#})$$

$$A(\delta|w)\geqslant\sum_{x,y}\tilde{P}(x,y)\sum_{i=1}^{n}\delta_i f_i(x,y)+1-$$
$$\sum_{x}\tilde{P}(x)\sum_{y}P_w(y|x)\sum_{i=1}^{n}\left(\frac{f_i(x,y)}{f^{\#}(x,y)}\right)\exp(\delta_i f^{\#}(x,y))$$

从A()->B()得到了新的下界：

$$L(w+\delta)-L(w)\geqslant B(\delta|w)$$

记不等式 (6.31) 右端为

$$B(\delta|w)=\sum_{x,y}\tilde{P}(x,y)\sum_{i=1}^{n}\delta_i f_i(x,y)+1-\sum_{x}\tilde{P}(x)\sum_{y}P_w(y|x)\sum_{i=1}^{n}\left(\frac{f_i(x,y)}{f^{\#}(x,y)}\right)\exp(\delta_i f^{\#}(x,y))$$

$$B(\delta|w)=\sum_{x,y}\tilde{P}(x,y)\sum_{i=1}^{n}\delta_i f_i(x,y)+1-\sum_{x}\tilde{P}(x)\sum_{y}P_w(y|x)\sum_{i=1}^{n}\left(\frac{f_i(x,y)}{f^{\#}(x,y)}\right)\exp(\delta_i f^{\#}(x,y))$$

求 $B(\delta|w)$ 对 δ_i 的偏导数：

$$\frac{\partial B(\delta|w)}{\partial \delta_i}=\sum_{x,y}\tilde{P}(x,y)f_i(x,y)-\sum_{x}\tilde{P}(x)\sum_{y}P_w(y|x)f_i(x,y)\exp(\delta_i f^{\#}(x,y)) \tag{6.32}$$

在式 (6.32) 里，除 δ_i 外不含任何其他变量。令偏导数为 0 得到

$$\sum_{x,y}\tilde{P}(x)P_w(y|x)f_i(x,y)\exp(\delta_i f^{\#}(x,y))=E_{\tilde{P}}(f_i) \tag{6.33}$$

于是，依次对 δ_i 求解方程 (6.33) 可以求出 δ。

这就给出了一种求 w 的最优解的迭代算法，即改进的迭代尺度算法 IIS。

$$P_w(y|x)=\frac{1}{Z_w(x)}\exp\left(\sum_{i=1}^{n}w_i f_i(x,y)\right)$$

$$Z_w(x)=\sum_{y}\exp\left(\sum_{i=1}^{n}w_i f_i(x,y)\right)$$

2 改进的迭代尺度法

算法 6.1（改进的迭代尺度算法 IIS）

输入：特征函数 $f_1, f_2, \cdots, f_n$；经验分布 $\tilde{P}(X,Y)$，模型 $P_w(y|x)$；

输出：最优参数值 w_i^*；最优模型 P_{w^*}。

（1）对所有 $i \in \{1,2,\cdots,n\}$，取初值 $w_i = 0$。

（2）对每一 $i \in \{1,2,\cdots,n\}$

（a）令 δ_i 是方程

$$\sum_{x,y} \tilde{P}(x)P(y|x)f_i(x,y)\exp(\delta_i f^{\#}(x,y)) = E_{\tilde{P}}(f_i)$$

的解，这里，

$$f^{\#}(x,y) = \sum_{i=1}^{n} f_i(x,y)$$

（b）更新 w_i 值：$w_i \leftarrow w_i + \delta_i$。

（3）如果不是所有 w_i 都收敛，重复步 (2)。 ■

这一算法关键的一步是 (a)，即求解方程 (6.33) 中的 δ_i。如果 $f^{\#}(x,y)$ 是常数，即对任何 x, y，有 $f^{\#}(x,y) = M$，那么 δ_i 可以显式地表示成

$$\delta_i = \frac{1}{M}\log\frac{E_{\tilde{P}}(f_i)}{E_P(f_i)} \tag{6.34}$$

如果 $f^{\#}(x,y)$不是常数，那么必须通过数值计算求 δ_i。简单有效的方法是牛顿法。

左边结束继续：

以 $g(\delta_i)=0$ 表示方程 (6.33)，牛顿法通过迭代求得 δ_i^*，使得 $g(\delta_i^*)=0$。迭代公式是

$$\delta_i^{(k+1)} = \delta_i^{(k)} - \frac{g(\delta_i^{(k)})}{g'(\delta_i^{(k)})} \tag{6.35}$$

只要适当选取初始值 $\delta_i^{(0)}$，由于 δ_i 的方程 (6.33) 有单根，因此牛顿法恒收敛，而且收敛速度很快。

$$E_p(f_i) = \sum_{x,y} P(x,y)f_i(x,y) = \sum_{x,y} P(x)P(y|x)f_i(x,y)$$

$$= \sum_{x,y} \tilde{P}(x)P(y|x)f_i(x,y)$$

$$E_{\tilde{p}}(f_i) = \sum_{x,y} \tilde{P}(x,y)f_i(x,y)$$

3 牛顿法-求解方程根

方程： $f(x)=0$

f(x)在x_0处泰勒展开，只要展开到一阶：

$$f(x)=f(x_0)+f'(x_0)(x-x_0)=0$$

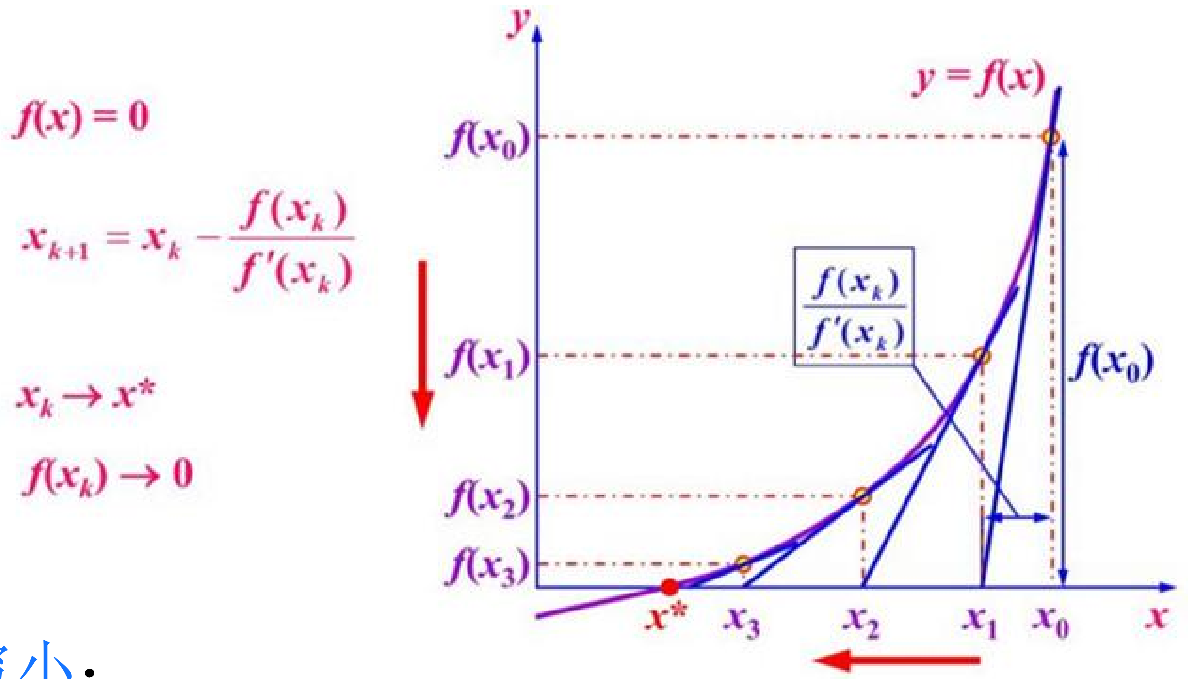
方程的解： $x=x_0-\dfrac{f(x_0)}{f'(x_0)}$

f(x)泰勒展开到一阶，后面其实还有高阶无穷小：

$$f(x)\approx f(x_0)+f'(x_0)(x-x_0)$$

上述求出的x实际并不能让 $f(x)=0$

所以迭代求解 $x_{k+1}=x_k-\dfrac{f(x_k)}{f'(x_k)}$



$$x_1=x_0-\frac{f(x_0)}{f'(x_0)}=x_0-\frac{f(x_0)}{\dfrac{f(x_0)}{|x_1x_0|}}=x_0-|x_1x_0|$$

3 牛顿法-求解方程根

方程g(x)=0：

$$\sum_{x,y} \tilde{P}(x) P_w(y|x) f_i(x,y) \exp(\delta_i f^{\#}(x,y)) = E_{\tilde{P}}(f_i) \tag{6.33}$$

于是，依次对 δ_i 求解方程 (6.33) 可以求出 δ。

以 $g(\delta_i)=0$ 表示方程 (6.33)，牛顿法通过迭代求得 δ_i^*，使得 $g(\delta_i^*)=0$。迭代公式是

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$$

$$\delta_i^{(k+1)} = \delta_i^{(k)} - \frac{g(\delta_i^{(k)})}{g'(\delta_i^{(k)})} \tag{6.35}$$

只要适当选取初始值 $\delta_i^{(0)}$，由于 δ_i 的方程 (6.33) 有单根，因此牛顿法恒收敛，而且收敛速度很快。

4 牛顿法-无约束最优化问题

无约束的极小化问题：$\min_{x \in R^N} f(x)$

当N=1时，即一元函数问题

$$f(x)=f(x_0)+f'(x_0)(x-x_0)+\frac{1}{2}f''(x_0)(x-x_0)^2+\ldots+\frac{1}{n!}f^{(n)}(x_0)(x-x_0)^n\ldots$$

忽略2次以上的项（即展开到2阶）

$$f(x)=f(x_0)+f'(x_0)(x-x_0)+\frac{1}{2}f''(x_0)(x-x_0)^2$$

求x使得f取极小值，对上面式子再求导，有一阶导数为0：

$$f'(x)=f'(x_0)+f''(x_0)(x-x_0)=0$$

解得

$$x=x_0-\frac{f'(x_0)}{f''(x_0)}$$

更一般的迭代形式

$$x_{k+1}=x_k-\frac{f'(x_k)}{f''(x_{k+1})}$$

4 牛顿法-无约束最优化问题

$x=\{x_1, x_2, ..., x_N\}$ 是函数f(x)的变元

附录B： 当N>1时，f在点$x^{(k)}$的泰勒展开式（这里$x^{(k)}$是迭代k次的点而不是f的一个元）：

$$f(x)=f(x^{(k)})+g_k^{\mathrm{T}}(x-x^{(k)})+\frac{1}{2}(x-x^{(k)})^{\mathrm{T}}H(x^{(k)})(x-x^{(k)})$$

f的一阶导数

$$g_k=g(x^{(k)})=\nabla f=\begin{bmatrix}\frac{\partial f}{\partial x_1}\\ \frac{\partial f}{\partial x_2}\\ \vdots\\ \frac{\partial f}{\partial x_N}\end{bmatrix}$$

f的二阶导数:黑塞(Hessian)矩阵

$$H(x)=\nabla^2 f=\begin{bmatrix}\frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1\partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1\partial x_N}\\ \frac{\partial^2 f}{\partial x_2\partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \cdots & \frac{\partial^2 f}{\partial x_2\partial x_N}\\ \vdots & \vdots & \ddots & \vdots\\ \frac{\partial^2 f}{\partial x_N\partial x_1} & \frac{\partial^2 f}{\partial x_N\partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_N^2}\end{bmatrix}_{N\times N}$$

对称矩阵

忽略2次以上的项，要求f(x)的极小值，有f(x)的一阶导数为0。对上式两边同时求梯度

$$f'(x)=g_k+H_k(x-x^{(k)})=0$$

$$x=x^{(k)}-H_k^{-1}g_k \Rightarrow x^{(k+1)}=x^{(k)}-H_k^{-1}g_k=x^{(k)}+p_k$$

4 牛顿法-无约束最优化问题

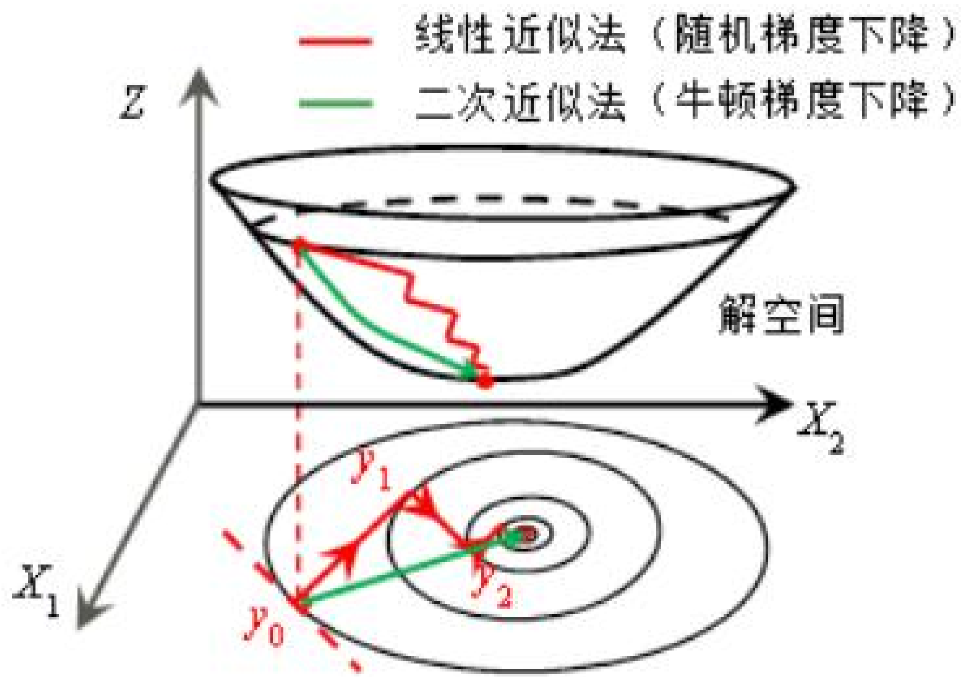


图1、线性近似与二阶近似优化方法示意图

牛顿法优点：不仅利用一阶偏导数，还利用了二阶偏导数，确定了更优的搜索方向，更快地收敛

4 牛顿法-无约束最优化问题

在点 $x^{(k)}$ 的值。函数 $f(x)$ 有极值的必要条件是在极值点处一阶导数为 0，即梯度向量为 0。特别是当 $H(x^{(k)})$ 是正定矩阵时，函数 $f(x)$ 的极值为极小值。

正定矩阵

（1）广义定义：设M是n阶方阵，如果对任何非零向量z，都有$z^TMz>0$，其中z^T表示z的转置，就称M为正定矩阵。

例如：B为n阶矩阵，E为单位矩阵，a为正实数。在a充分大时，aE+B为正定矩阵。（B必须为对称阵）

（2）狭义定义：一个n阶的实对称矩阵M是正定的的条件是当且仅当对于所有的非零实系数向量z，都有$z^TMz>0$。其中z^T表示z的转置。

正定矩阵有以下性质 [1] ：

（1）正定矩阵的行列式恒为正；

（2）实对称矩阵A正定当且仅当A与单位矩阵合同；

（3）若A是正定矩阵，则A的逆矩阵也是正定矩阵；

（4）两个正定矩阵的和是正定矩阵；

（5）正实数与正定矩阵的乘积是正定矩阵。

$$x=x^{(k+1)}=x^{(k)}-H_k^{-1}g_k$$

$$f(x)=f(x^{(k)})+g_k^{\mathrm{T}}(x-x^{(k)})+\frac{1}{2}(x-x^{(k)})^{\mathrm{T}}H(x^{(k)})(x-x^{(k)})$$

$$f(x)\approx f(x^{(k)})+g_k^T(-H_k^{-1}g_k)=f(x^{(k)})-g_k^TH_k^{-1}g_k$$

在一定程度上保证了向f(x)减小的方向上迭代

如果黑塞矩阵不可逆？

4 牛顿法-无约束最优化问题

算法 B.1（牛顿法）

输入：目标函数 $f(x)$，梯度 $g(x)=\nabla f(x)$，黑塞矩阵 $H(x)$，精度要求 ε；

输出：$f(x)$ 的极小点 x^*。

（1）取初始点 $x^{(0)}$，置 $k=0$。

（2）计算 $g_k=g(x^{(k)})$。 计算x(k)的一阶导数

（3）若 $\|g_k\|<\varepsilon$，则停止计算，得近似解 $x^*=x^{(k)}$。

（4）计算 $H_k=H(x^{(k)})$，并求 p_k

$$H_k p_k=-g_k$$

（5）置 $x^{(k+1)}=x^{(k)}+p_k$。

（6）置 $k=k+1$，转（2）。 ■

步骤（4）求 p_k，$p_k=-H_k^{-1}g_k$，要求 H_k^{-1}，计算比较复杂，所以有其他改进的方法。

牛顿法的迭代公式：

$$x^{(k+1)}=x^{(k)}-H_k^{-1}g_k=x^{(k)}+p_k$$

$$p_k=-H_k^{-1}g_k$$

牛顿法缺点：

- 函数必须有二阶偏导数，黑塞矩阵为正定
- 需要计算黑塞矩阵和它的逆矩阵，计算量大
- 固定步长更新，不能保证迭代方向(p_k)一定沿着函数值下降方向

5 阻尼牛顿法

阻尼牛顿法的迭代方向仍为 $p_k = -H_k^{-1} g_k$

但每次迭代需沿此方向作一维搜索（line search），寻求最优的步长因子 λ_k

$$\lambda_k = \arg\min_{\lambda \in R} f(x_k + \lambda p_k)$$

阻尼牛顿法保证迭代方向(p_k)一定沿着函数值下降方向。

- 函数必须有二阶偏导数，黑塞矩阵为正定
- 需要计算黑塞矩阵和它的逆矩阵，计算量大

拟牛顿法

基本思想：不用二阶偏导数，而是构造出可以近似黑塞矩阵（或黑塞矩阵的逆）的正定对称矩阵，在“拟牛顿条件”下优化目标函数.

BFGP

DFP

6 拟牛顿法（BFGS）

在点x (k+1) 二阶展开

$$f(x) \approx f(x^{(k+1)}) + \nabla f(x^{(k+1)})(x - x^{(k+1)}) + \frac{1}{2}(x - x^{(k+1)})^T \nabla^2 f(x^{(k+1)})(x - x^{(k+1)})$$

两边求导

$$\nabla f(x) \approx \nabla f(x^{(k+1)}) + H_{k+1}(x - x^{(k+1)})$$

当x=x (k)

$$g_{k+1} - g_k \approx H_{k+1}(x^{(k+1)} - x^{(k)})$$

令 $y_k = g_{k+1} - g_k, \delta_k = x^{(k+1)} - x^{(k)}$

拟牛顿条件：

$$\left\{\begin{array}{l} \delta_k = H_{k+1}^{-1} y_k \quad \Rightarrow \quad \text{DFP} \\ y_k = H_{k+1}\delta_k \quad \Rightarrow \quad \text{BFGS} \end{array}\right.$$

BFGS：

用B表示黑塞矩阵H的近似

$$B_k = H_k$$

$$B_{k+1} = B_k + \Delta B_k, k = 0, 1, 2, \cdots$$

$$\Delta B_k = \alpha \boldsymbol{\mu}\boldsymbol{\mu}^T + \beta \boldsymbol{v}\boldsymbol{v}^T$$

B_0示通常取单位矩阵

α 和 β 是待定系数

μ，ν，y_k，δ $_k$是nx1的向量

$$y_k = B_k\delta_k + (\alpha\mu^T\mu)\delta_k + (\beta v^T v)\delta_k = B_k\delta_k + (\underline{\alpha\mu^T\delta_k})\mu + (\underline{\beta v^T\delta_k})v$$

6 拟牛顿法（BFGS）

$$y_k = B_k\delta_k + (\alpha\mu^T\mu)\delta_k + (\beta v^T v)\delta_k = B_k\delta_k + (\alpha\mu^T\delta_k)\mu + (\beta v^T\delta_k)v$$

令 $\alpha\mu^T\delta_k = 1, \beta v^T\delta_k = -1$　　$y_k = B_k\delta_k + \mu - v$

令 $\mu = y_k, v = B_k\delta_k$

$$\alpha = \frac{1}{y_k{}^T\delta_k}, \beta = \frac{-1}{(B_k\delta_k)^T\delta_k} = \frac{-1}{\delta_k{}^T B_k\delta_k}$$

$$\Delta B_k = \alpha\boldsymbol{\mu}\boldsymbol{\mu}^T + \beta\boldsymbol{v}\boldsymbol{v}^T$$

$$\Delta B_k = \frac{y_k y_k{}^T}{y_k{}^T\delta_k} - \frac{B_k\delta_k\delta_k{}^T B_k}{\delta_k{}^T B_k\delta_k}$$

6 拟牛顿法

最大熵模型的极大似然估计：

已知训练数据的经验概率分布 $\tilde{P}(X,Y)$，条件概率分布 $P(Y|X)$ 的对数似然函数表示为

$$L_{\tilde{P}}(P_w)=\log\prod_{x,y}P(y|x)^{\tilde{P}(x,y)}=\sum_{x,y}\tilde{P}(x,y)\log P(y|x)$$

当条件概率分布 $P(y|x)$ 是最大熵模型 (6.22) 和 (6.23) 时，对数似然函数 $L_{\tilde{P}}(P_w)$ 为

$$\begin{aligned}L_{\tilde{P}}(P_w)&=\sum_{x,y}\tilde{P}(x,y)\log P(y|x)\\&=\sum_{x,y}\tilde{P}(x,y)\sum_{i=1}^{n}w_if_i(x,y)-\sum_{x,y}\tilde{P}(x,y)\log Z_w(x)\\&=\sum_{x,y}\tilde{P}(x,y)\sum_{i=1}^{n}w_if_i(x,y)-\sum_{x}\tilde{P}(x)\log Z_w(x)\end{aligned}\tag{6.26}$$

对于最大熵模型而言，

$$P_w(y|x)=\frac{\exp\left(\sum_{i=1}^{n}w_if_i(x,y)\right)}{\sum_y\exp\left(\sum_{i=1}^{n}w_if_i(x,y)\right)}$$

目标函数：

$$\min_{w\in\mathbf{R}^n}\quad f(w)=\sum_x\tilde{P}(x)\log\sum_y\exp\left(\sum_{i=1}^{n}w_if_i(x,y)\right)-\sum_{x,y}\tilde{P}(x,y)\sum_{i=1}^{n}w_if_i(x,y)$$

梯度：

$$g(w)=\left(\frac{\partial f(w)}{\partial w_1},\frac{\partial f(w)}{\partial w_2},\cdots,\frac{\partial f(w)}{\partial w_n}\right)^{\mathrm{T}}$$

其中

$$\frac{\partial f(w)}{\partial w_i}=\sum_{x,y}\tilde{P}(x)P_w(y|x)f_i(x,y)-E_{\tilde{P}}(f_i),\quad i=1,2,\cdots,n$$

算法 6.2（最大熵模型学习的 BFGS 算法）

输入：特征函数 $f_1, f_2, \cdots, f_n$；经验分布 $\tilde{P}(x,y)$，目标函数 $f(w)$，梯度 $g(w) = \nabla f(w)$，精度要求 ε；

输出：最优参数值 w^*；最优模型 $P_{w^*}(y|x)$。

（1）选定初始点 $w^{(0)}$，取 B_0 为正定对称矩阵，置 $k=0$；

（2）计算 $g_k = g(w^{(k)})$。若 $\|g_k\| < \varepsilon$，则停止计算，得 $w^* = w^{(k)}$；否则转 (3)；

（3）由 $B_k p_k = -g_k$ 求出 p_k；

（4）一维搜索：求 λ_k 使得

$$f(w^{(k)} + \lambda_k p_k) = \min_{\lambda \geqslant 0} f(w^{(k)} + \lambda p_k)$$

（5）置 $w^{(k+1)} = w^{(k)} + \lambda_k p_k$；

（6）计算 $g_{k+1} = g(w^{(k+1)})$，若 $\|g_{k+1}\| < \varepsilon$，则停止计算，得 $w^* = w^{(k+1)}$；否则，按下式求出 B_{k+1}：

$$B_{k+1} = B_k + \frac{y_k y_k^{\mathrm{T}}}{y_k^{\mathrm{T}} \delta_k} - \frac{B_k \delta_k \delta_k^{\mathrm{T}} B_k}{\delta_k^{\mathrm{T}} B_k \delta_k}$$

其中，

$$y_k = g_{k+1} - g_k, \quad \delta_k = w^{(k+1)} - w^{(k)}$$

（7）置 $k = k+1$，转 (3)。 ■

$$B_{k+1} = B_k + \Delta B_k, k = 0, 1, 2, \cdots$$

$$\Delta B_k = \frac{y_k y_k^T}{y_k^T \delta_k} - \frac{B_k \delta_k \delta_k^T B_k}{\delta_k^T B_k \delta_k}$$

7 总结

最大熵：有条件约束的原始问题

$\Downarrow$

对偶问题的最大化或极大似然估计

模型参数ω的学习算法：

- 改进的迭代尺度法
- 牛顿法 $\Rightarrow$ 阻尼牛顿法 $\Rightarrow$ 拟牛顿法（BFGS）

➢ 找方程根
➢ 无条件最优化问题

8 代码

no	sunny	hot	high	FALSE
no	sunny	hot	high	TRUE
yes	overcast	hot	high	FALSE
yes	rainy	mild	high	FALSE
yes	rainy	cool	normal	FALSE
no	rainy	cool	normal	TRUE
yes	overcast	cool	normal	TRUE
no	sunny	mild	high	FALSE
yes	sunny	cool	normal	FALSE
yes	rainy	mild	normal	FALSE
yes	sunny	mild	normal	TRUE
yes	overcast	mild	high	TRUE
yes	overcast	hot	normal	FALSE
no	rainy	mild	high	TRUE

```python
def __init__(self):
    self._samples = []  # 样本集, 元素是[y,x1,x2,...,xn]的元组
    self._Y = set([])  # 标签集合,相当于去重之后的y
    self._numXY = defaultdict(int)  # Key是(xi,yi)对, Value是count(xi,yi)
    self._N = 0  # 样本数量
    self._n = 0  # 特征对(xi,yi)总数量
    self._xyID = {}  # 对(x,y)对做的顺序编号(ID), Key是(xi,yi)对,Value是ID
    self._C = 0  # 样本最大的特征数量,用于求参数时的迭代, 见IIS原理说明
    self._ep_ = []  # 样本分布的特征期望值
    self._ep = []  # 模型分布的特征期望值
    self._w = []  # 对应n个特征的权值
    self._lastw = []  # 上一轮迭代的权值
    self._EPS = 0.01  # 判断是否收敛的阈值
```

是否出门玩　天气　温度　湿度　是否有风

8 代码

```
def train(self, maxiter=1000):
    self._initparams()
    for i in range(0, maxiter):
        # print("Iter:%d..." % i)
        self._lastw = self._w[:]  # 保存上一轮权值
        self._model_ep()
        # 更新每个特征的权值
        for i, w in enumerate(self._w):
            # 参考公式(19)
            self._w[i] += 1.0 / self._C * math.log(self._ep_[i] / self._ep[i])
        # print(self._w)
        # 检查是否收敛
        if self._convergence():
            break

def predict(self, input):
    X = input.strip().split("\t")
    prob = self._pyx(X)
    return prob
```

基于GIS方法

$$\lambda_i^{(t+1)} = \lambda_i^{(t)} + \frac{1}{C} \log \frac{E_{\tilde{p}} f_i}{E_{p^{(n)}} f_i}, i \in \{1, 2, ..., n\} \text{ ——— } (19)$$

8 代码

```python
if __name__ == "__main__":
    maxent = MaxEnt()
    maxent.load_data('data.txt')
    maxent.train()

    print("sunny\thot\thigh\tFALSE")
    print(maxent.predict("sunny\thot\thigh\tFALSE"))
    print("overcast\thot\thigh\tFALSE")
    print(maxent.predict("overcast\thot\thigh\tFALSE"))
    print("sunny\tcool\thigh\tTRUE")
    print(maxent.predict("sunny\tcool\thigh\tTRUE"))
    sys.exit(0)
```

```
sunny	hot high	FALSE
[('yes', 0.0041626518719797), ('no', 0.9958373481280207)]
overcast	hot high	FALSE
[('yes', 0.9943682102360447), ('no', 0.00563178976395368)]
sunny	cool	high	TRUE
[('yes', 1.4464465173635736e-07), ('no', 0.9999998553553483)]
```

THANK YOU