概率模型简介

机器学习模型主要分为两类：
1）判别模型：通过学习，得到某个f，预测$y_i = f(x_i)$；
2）概率模型：通过学习，得到每一个标签y_i的条件概率$P(y_i|X)$，选择$y = \arg\min_{y_k} \quad R(y_k)$，其中$R(y_k)$是预测标签值为$y_k$所带来的损失。
下文所介绍的模型都是概率模型。
对于概率模型而言，首先我们需要定义损失函数R。一般情况下，我们采用$0-1$损失，即：

$$R(y_{pred}) = \begin{cases} 0 & y_{pred} = y_{true} \\ 1 & y_{pred} \neq y_{true} \end{cases}$$

于是预测$y_{pred} = y_k$所产生的损失为

$$\begin{aligned} R(y_k) &= P(y_k|X) * 0 + \sum_{l \neq k}^{K} P(y_l|X) * 1 \\ &= \sum_{l \neq k}^{K} P(y_l|X) \\ &= 1 - P(y_k|X) \end{aligned}$$

这里利用条件概率值之和为1的性质：$\sum_{l=1}^{K} P(y_l|X) = 1$。
由上文，我们选择$y = \arg\min_{y_k} \quad R(y_k) = \arg\max_{y_k} \quad P(y_k|X)$，即当损失函数是$0-1$损失时，选择条件概率最大的标签作为预测值。

朴素贝叶斯

模型的建立与独立性假设

考虑数据集$D = \{(X_i, y_i)\}$，记数据集大小为N，维度数为n，y有K个取值。根据上文"概率模型"的思路，我们希望计算$\arg\max_{y_k} \quad P(y = y_k|X)$。由贝叶斯公式可知：

$$P(y|X) = \frac{P(X,y)}{P(X)} = \frac{\sum_{k=1}^{K} P(X|y = y_k) \cdot P(y = y_k)}{P(X)}$$

然而，为了估计$P(X)$与$P(X|y = y_k)$，我们需要计算$P(X_1 = x_{l_1 1}, X_2 = x_{l_2 2}, \ldots, X_n = x_{l_n n})$，其中$l_1 = 1, 2, \ldots, N_1; l_2 = 1, 2, \ldots, N_2; \ldots; l_n = 1, 2, \ldots, N_n$，$N_n$是$X$的第$n$列的可能取值数。于是我们可以发现，需要计算$\prod_{j=1}^{n} N_j$次概率，这在实际上是成本过高、行不通的。于是我们不妨做一个较强的假设，即：X的各列之间是独立的。在独立性假设下，$P(X) = \prod_{j=1}^{n} P(X_j)$，我们只需计算$\sum_{j=1}^{n} N_j$次概率，这大大简化了模型的计算量。基于此，$P(y|X)$可以表示为：

$$
\begin{aligned}
P(y|X)=\frac{P(X,y)}{P(X)}&=\frac{\sum\limits_{k=1}^{K}P(X|y=y_k)\cdot P(y=y_k)}{P(X)}\\
&=\frac{\sum\limits_{k=1}^{K}\left(\prod\limits_{j=1}^{n}P(X_j|y=y_k)\right)\cdot P(y=y_k)}{\prod\limits_{j=1}^{n}P(X_j)}
\end{aligned}
$$

在实际计算时，我们需要用频率来估算概率：

$$
\begin{aligned}
P(y=y_k)&=\frac{\nu(y=y_k)}{N}\\
P(X_j=x_{l_jj})&=\frac{\nu(X_j=x_{l_jj})}{N}\\
P(X_j=x_{l_jj}|y=y_k)&=\frac{\nu(X_j=x_{l_jj}|y=y_k)}{N}
\end{aligned}
$$

其中$\nu(\cdot)$代表了某条件出现的频数。

在预测时，我们需要计算$P(y=y_k|X)$：

$$
P(y=y_k|X)=\frac{\frac{\nu(y=y_k)}{N}\cdot\prod\limits_{j=1}^{n}\frac{\nu(X_j=x_{l_jj}|y=y_k)}{N}}{\prod\limits_{j=1}^{n}\frac{\nu(X_j=x_{l_jj})}{N}}
$$

然后选择$y_{pred}=\arg\max\limits_{y_k}\quad P(y=y_k|X)$。

平滑因子的添加

从上式可以发现，对于某条被预测的训练集观测，只要任一特征的取值在训练集中没有出现过，都会导致其对每一类的预测概率都为0，使得无法准确的预测。为此，我们可以添加一个平滑因子λ，在计算频率时，做如下的改进：

$$
\begin{aligned}
P(y=y_k)=\frac{\nu(y=y_k)}{N}&\Rightarrow\frac{\nu(y=y_k)+\lambda}{N+K\lambda}\\
P(X_j=x_{l_jj})=\frac{\nu(X_j=x_{l_jj})}{N}&\Rightarrow\frac{\nu(X_j=x_{l_jj})+\lambda}{N+N_l\lambda}\\
P(X_j=x_{l_jj}|y=y_k)=\frac{\nu(X_j=x_{l_jj}|y=y_k)}{N}&\Rightarrow\frac{\nu(X_j=x_{l_jj}|y=y_k)+\lambda}{N+N_{j,k}\lambda}
\end{aligned}
$$

其中$N_{j,k}=\nu(X_j=x_{l_j,j},y=y_k)$。

当$\lambda=1$时，称以上的平滑方法为Laplace平滑。这种方法保证了$P(y=y_k|X)$至少可以取到一个很小的常数，从而增加了模型的泛化能力。

显然，朴素贝叶斯模型更适用于各列之间独立、数据维度较少，维度取值也较少的数据集。

逻辑斯蒂回归

二分类逻辑斯蒂回归

对于数据集$D=\{(X_i,y_i)\},y_i\in\{0,1\}$而言，我们只需知道$P(y=1|X)$（$P(y=0|X)=1-P(y=1|X)$），就能得到关于该数据集的概率模型。我们可以对$P(y=1|X)$设定一些参数，例如设定：

$$P(y=1|X_i)=\frac{1}{1+e^{-\omega\cdot X_i}}$$
$$P(y=0|X_i)=\frac{e^{-\omega\cdot X_i}}{1+e^{-\omega\cdot X_i}}$$

$i=1,2,\dots,N$。

我们用最大似然估计来计算参数：

$$\omega=\arg\max_{\omega}\quad\prod_{i=1}^{N}P(y_i|X_i)=\arg\max_{\omega}\quad\sum_{i=1}^{N}\log P(y_i|X_i)$$

其中$\log P(y_i|X_i)$等价于$y_i\cdot\log P(y=1|X_i)+(1-y_i)\cdot\log P(y=0|X_i)$。于是，对数似然函数$\log L(\omega)$可以表示为：

$$\log L(\omega)=\sum_{i=1}^{N}y_i\cdot\log P(y=1|X_i)+(1-y_i)\cdot\log P(y=0|X_i)$$

为了求$\arg\max\limits_{\omega}\quad\log L(\omega)$，我们可以采用梯度下降法、牛顿法与拟牛顿法等。接下来我们将简要介绍梯度下降法，牛顿法与拟牛顿法将在下一节详细介绍。

顾名思义，梯度下降法是将函数不断沿着梯度的反方向前进，直到收敛到最小值的过程。使用梯度下降法时，需要选择学习率α与精度ϵ。当下降的量小于ϵ时，认为函数已经收敛。梯度下降法的算法如下：

1）对函数$f(x)$，取初值x_0并计算$f(x_0)$；读取学习率α与精度ϵ，设置迭代次数为0；

2）计算下降值为$\alpha\frac{\partial f}{\partial x}|_{x=x_k}$，若$|\alpha\frac{\partial f}{\partial x}|_{x=x_k}|<\epsilon$，则结束迭代；否则，$f_{k+1}=f_k-\alpha\frac{\partial f}{\partial x}|_{x=x_k}$；

3）$k\to k+1$。

注：对于凸函数，梯度下降法能收敛到全局最优解；但对于非凸函数，梯度下降法可能在局部最优解收敛。

如果我们采用梯度下降法来求解，需要计算$\frac{\log L(\omega)}{\partial\omega}$：

$$\begin{aligned}\frac{\partial\log L(\omega)}{\partial\omega}&=\sum_{i=1}^{N}\frac{y_i}{P(y=1|X_i)}*\frac{\partial P(y=1|X_i)}{\partial\omega}+\frac{1-y_i}{P(y=0|X_i)}*\frac{\partial P(y=0|X_i)}{\partial\omega}\\&=\sum_{i=1}^{N}\frac{y_i}{P(y=1|X_i)}*\frac{\partial P(y=1|X_i)}{\partial\omega}-\frac{1-y_i}{1-P(y=1|X_i)}*\frac{\partial P(y=1|X_i)}{\partial\omega}\\&\xlongequal{\frac{\partial P(y=1|X_i)}{\partial\omega}=P(y=1|X_i)(1-P(y=1|X_i))}\sum_{i=1}^{N}y_i(1-P(y=1|X_i))-(1-y_i)P(y=1|X_i)\\&=\sum_{i=1}^{N}\left(y_i-\frac{1}{1+e^{-\omega\cdot X_i}}\right)\end{aligned}$$

然后令$\omega_{k+1}=\omega_k-\alpha\frac{\partial\log L(\omega)}{\partial\omega}|_{\omega=\omega_k}$即可。

牛顿法与拟牛顿法

考虑最优化问题

$$\min_x \quad f(x)$$

其中x是n维向量$(x_1, x_2, \dots, x_N)^{\mathrm{T}}$。

假设在第k轮迭代时的$x = x^{(k)}$，那么将$f(x)$在$x^{(k)}$处做二阶Taylor展开：

$$f(x) \sim f(x^{(k)}) + g(x^{(k)})(x - x^{(k)}) + \frac{1}{2}(x - x^{(k)})^{\mathrm{T}} H(x^{(k)})(x - x^{(k)}) \equiv \tilde{f}(x)$$

其中

$$g(x) = \boldsymbol{\nabla}_x f = \left(\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \dots, \frac{\partial f}{\partial x_n} \right)$$

$$H(x) = \boldsymbol{\nabla}_x (\boldsymbol{\nabla}_x f) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \cdots & \frac{\partial^2 f}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \frac{\partial^2 f}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix}$$

我们称$H(x)$为函数f的Hessian矩阵。

我们通过二阶Taylor展开将$f(x)$近似为一个二次函数$\tilde{f}(x)$。对于该二次函数，我们可以得到其解析解。当$H(x)$正定时，$\tilde{f}(x)$是一个开口向上的二次函数，其导数为

$$\frac{\partial \tilde{f}(x)}{\partial x} = g(x^{(k)}) + H(x^{(k)})(x - x^{(k)})$$

自然地，为了求$\min_x \quad \tilde{f}(x)$，我们令$\frac{\partial \tilde{f}(x)}{\partial x} = 0$，即

$$g(x^{(k)}) + H(x^{(k)})(x - x^{(k)}) = 0$$

解得

$$x = x^{(k)} - H(x^{(k)})^{-1} g(x^{(k)})$$

下面我们记$H(x^{(k)}) \equiv H_k$，$g(x^{(k)}) \equiv g_k$。

自然地，我们选择使$\frac{\partial \tilde{f}(x)}{\partial x} = 0$的点作为下一轮迭代的$x$，即

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

一直迭代直到$H_k^{-1} g_k$小于预先设定的精度ϵ（由于H_k正定，所以$H_k^{-1} g_k$恒为正）。

与梯度下降法类似，我们也可以设置学习率（在牛顿法中称为“步长”）λ。但与梯度下降法不同的是，步长不是预先设定的，而是通过如下的一维搜索确定的：

$$\lambda_k = \arg\max_{\lambda \geq 0} \quad f(x^{(k)}) - f(x^{(k)} - \lambda_k H_k^{-1} g_k)$$

然而，对于维度较高的向量而言，H_k将会非常大，从而导致计算H_k^{-1}的成本会非常高（计算逆矩阵的时间复杂度为$\Theta(n^3)$）。如果我们能找到一个矩阵近似代替H_k^{-1}或找到一个易于求逆的矩阵代替H_k，这样就可以降低计算的时间复杂度。我们把这种方法称为拟牛顿法。

首先介绍Davidon-Fletcher-Powell(DFP)拟牛顿算法。该算法采用矩阵D_k来近似代替H_k^{-1}。

由于H_k是正定的，所以H_k^{-1}也是正定的，也就是说，用于近似的矩阵D_k也是正定的。于是我们可以假设$D_k = \alpha uu^{\mathrm{T}} + \beta vv^{\mathrm{T}}$，其中$u, v$是单位向量。在每次迭代时，需要更新$D_k$：

$$D_{k+1} = D_k + \Delta D_k$$

我们设定D_0为单位阵，然后随着迭代的进行，D_k会越来越接近H_k^{-1}。那么我们只需要解出ΔD_k。

将$f(x)$在$x^{(k+1)}$处做二阶Taylor展开：

$$f(x) \sim \tilde{f}(x) = f(x^{(k+1)}) + g_{k+1}(x - x^{(k+1)}) + \frac{1}{2}(x - x^{(k+1)})^{\mathrm{T}} H_{k+1}(x - x^{(k+1)})$$

$$\frac{\partial f(x)}{\partial x} \sim \frac{\partial \tilde{f}(x)}{\partial x} = g_{k+1} + H_{k+1}(x - x^{(k+1)})$$

将$x = x_k$代入得

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

此式称为**拟牛顿条件**，它表示了对H_{k+1}的约束。将D_{k+1}作为H_{k+1}的近似代入得：

$$\begin{aligned} x^{(k+1)} - x^{(k)} &= D_{k+1}(g_{k+1} - g_k) \\ &= (D_k + \Delta D_k)(g_{k+1} - g_k) \\ &= (D_k + \alpha uu^{\mathrm{T}} + \beta vv^{\mathrm{T}})(g_{k+1} - g_k) \\ &= D_k(g_{k+1} - g_k) + \alpha uu^{\mathrm{T}}(g_{k+1} - g_k) + \beta vv^{\mathrm{T}}(g_{k+1} - g_k) \end{aligned}$$

由于$u^{\mathrm{T}}(g_{k+1} - g_k)$与$v^{\mathrm{T}}(g_{k+1} - g_k)$都是标量，所以可以提出：

$$\begin{aligned} x^{(k+1)} - x^{(k)} &= D_{k+1}(g_{k+1} - g_k) \\ &= D_k(g_{k+1} - g_k) + \alpha u^{\mathrm{T}}(g_{k+1} - g_k)u + \beta v^{\mathrm{T}}(g_{k+1} - g_k)v \end{aligned}$$

令$\alpha u^{\mathrm{T}}(g_{k+1} - g_k) = 1$，$\beta v^{\mathrm{T}}(g_{k+1} - g_k) = -1$，则有$\alpha = \frac{1}{u^{\mathrm{T}}(g_{k+1} - g_k)}$，$\beta = -\frac{1}{v^{\mathrm{T}}(g_{k+1} - g_k)}$，上式可以进一步简化：

$$(x^{(k+1)} - x^{(k)}) - D_k(g_{k+1} - g_k) = u - v$$

自然地，令$u = x^{(k+1)} - x^{(k)}$，$v = D_k(g_{k+1} - g_k)$即可满足条件。当然这样的α, β, u, v不唯一。

记$y_k = g_{k+1} - g_k$，$\delta_k = x^{(k+1)} - x^{(k)}$，可以得到DFP法的近似Hessian逆矩阵$D_k$的递推式：

$$\begin{aligned} D_{k+1} &= D_k + \Delta D_k \\ &= D_k + \alpha uu^{\mathrm{T}} + \beta vv^{\mathrm{T}} \\ &= D_k + \frac{\delta_k \delta_k^{\mathrm{T}}}{\delta_k^{\mathrm{T}} y_k} - \frac{D_k y_k y_k^{\mathrm{T}} D_k}{y_k^{\mathrm{T}} D_k y_k} \end{aligned}$$

于是我们可以得到DFP算法：

1）初始化：$x^{(0)} = 0$，$D_0 = I_n$，$k = 0$，读取$f(x)$与精度ϵ；

2）计算g_k；

3）计算步长$\lambda_k = \underset{\lambda \geq 0}{\arg\max} \quad f(x^{(k)}) - f(x^{(k)} - \lambda_k D_k g_k)$；

4）如果下降值$\lambda_k D_k g_k < \epsilon$，结束程序；否则计算$x^{(k+1)} = x^{(k)} - \lambda_k D_k g_k$；

5）计算$y_k = g_{k+1} - g_k$，$\delta_k = x^{(k+1)} - x^{(k)}$，$D_{k+1} = D_k + \frac{\delta_k \delta_k^{\mathrm{T}}}{\delta_k^{\mathrm{T}} y_k} - \frac{D_k \delta_k \delta_k^{\mathrm{T}} D_k}{\delta_k^{\mathrm{T}} D_k \delta_k}$；

6）$k \to k + 1$

接下来我们介绍类似的一种拟牛顿算法：Broyden-Fletcher-Goldfarb-Shanno(BFGS)算法。该算法采用一个易于求逆的矩阵B_k来近似代替H_k。

与DFP算法的情况类似，B_k作为H_k的近似，应当满足拟牛顿条件：

$$\begin{aligned} g_{k+1}-g_k &= B_{k+1}(x^{(k+1)}-x^{(k)}) \\ &= (B_k+\Delta B_k)(x^{(k+1)}-x^{(k)}) \\ &= (B_k+\alpha uu^{\mathrm{T}}+\beta vv^{\mathrm{T}})(x^{(k+1)}-x^{(k)}) \\ &= B_k(x^{(k+1)}-x^{(k)})+\alpha uu^{\mathrm{T}}(x^{(k+1)}-x^{(k)})+\beta vv^{\mathrm{T}}(x^{(k+1)}-x^{(k)}) \\ &= B_k(x^{(k+1)}-x^{(k)})+\alpha u^{\mathrm{T}}(x^{(k+1)}-x^{(k)})u+\beta v^{\mathrm{T}}(x^{(k+1)}-x^{(k)})v \end{aligned}$$

令$\alpha u^{\mathrm{T}}(x^{(k+1)}-x^{(k)})=1$，$\beta v^{\mathrm{T}}(x^{(k+1)}-x^{(k)})=-1$，则有$\alpha=\frac{1}{u^{\mathrm{T}}(x^{(k+1)}-x^{(k)})}$，$\beta=-\frac{1}{v^{\mathrm{T}}(x^{(k+1)}-x^{(k)})}$，上式可以进一步简化：

$$\begin{gathered} (g_{k+1}-g_k)-B_k(x^{(k+1)}-x^{(k)})=u-v \\ u=g_{k+1}-g_k,\quad v=B_k(x^{(k+1)}-x^{(k)}) \end{gathered}$$

同样地，记$y_k=g_{k+1}-g_k$，$\delta_k=x^{(k+1)}-x^{(k)}$，可以得到BFGS法的近似Hessian矩阵$B_k$的递推式：

$$\begin{aligned} B_{k+1} &= B_k+\Delta B_k \\ &= B_k+\alpha uu^{\mathrm{T}}+\beta vv^{\mathrm{T}} \\ &= B_k+\frac{y_ky_k^{\mathrm{T}}}{y_k^{\mathrm{T}}\delta_k}-\frac{B_k\delta_k\delta_k^{\mathrm{T}}B_k}{\delta_k^{\mathrm{T}}B_k\delta_k} \end{aligned}$$

根据Sherman-Morrison公式：当A是n阶可逆矩阵，u，v是n维向量，且$A+uv^{\mathrm{T}}$可逆，则$A+uv^{\mathrm{T}}$的逆可以如此计算：

$$(A+uv^{\mathrm{T}})^{-1}=A^{-1}-\frac{A^{-1}uv^{\mathrm{T}}A^{-1}}{1+v^{\mathrm{T}}A^{-1}u}$$

有（推导过程特别复杂且无意义，略去）：

$$B_{k+1}^{-1}=\left(I-\frac{\delta_ky_k^{\mathrm{T}}}{\delta_k^{\mathrm{T}}y_k}\right)B_k^{-1}\left(I-\frac{\delta_ky_k^{\mathrm{T}}}{\delta_k^{\mathrm{T}}y_k}\right)^{\mathrm{T}}+\frac{\delta_k\delta_k^{\mathrm{T}}}{\delta_k^{\mathrm{T}}y_k}$$

于是我们可以得到BFGS算法：

1）初始化：$x^{(0)}=0$，$B_0=B_0^{-1}=I_n$，$k=0$，读取$f(x)$与精度ϵ；

2）计算g_k；

3）计算步长$\lambda_k=\underset{\lambda\geq 0}{\arg\max}\quad f(x^{(k)})-f(x^{(k)}-\lambda_kB_k^{-1}g_k)$；

4）如果下降值$\lambda_kB_k^{-1}g_k<\epsilon$，结束程序；否则计算$x^{(k+1)}=x^{(k)}-\lambda_kB_k^{-1}g_k$；

5）计算$y_k=g_{k+1}-g_k$，$\delta_k=x^{(k+1)}-x^{(k)}$，$B_{k+1}^{-1}=\left(I-\frac{\delta_ky_k^{\mathrm{T}}}{\delta_k^{\mathrm{T}}y_k}\right)B_k^{-1}\left(I-\frac{\delta_ky_k^{\mathrm{T}}}{\delta_k^{\mathrm{T}}y_k}\right)^{\mathrm{T}}+\frac{\delta_k\delta_k^{\mathrm{T}}}{\delta_k^{\mathrm{T}}y_k}$；

6）$k\to k+1$

记$G_k^{BFGS}=B_k^{-1}$，则对$\forall\alpha\in[0,1]$，G_k^{DFP}和G_k^{BFGS}的加权平均都满足拟牛顿条件：

$$G_k=\alpha G_k^{DFP}+(1-\alpha)G_k^{BFGS}$$

记以G_k代替H_k^{-1}的方法为类Broyden算法。DFP算法与BFGS算法都是类Broyden算法的一种特例。

在实际应用中，BFGS算法比DFP算法更为常用，是因为BFGS算法在迭代时，有自校正的效果，即在B_k偏离收敛方向时，下一次迭代能令它回到收敛方向（不是很理解，详见http://terminus.sdsu.edu/SDSU/Math693a/Lectures/18/lecture.pdf P22-23）。

多分类逻辑斯蒂回归

当二分类问题拓展到多分类时，$P(y=y_k|X_i)$可以被设定为

$$P(y=y_k|X_i)=\begin{cases}\frac{e^{\omega_k\cdot X_i}}{1+\sum\limits_{k=1}^{K}e^{\omega_k\cdot X_i}} & k>0\\ \frac{1}{1+\sum\limits_{k=1}^{K}e^{\omega_k\cdot X_i}} & k=0\end{cases}$$

当$K=2$时，情形与二分类时的逻辑斯蒂回归相同。形如上式概率的函数被称为$softmax$函数：

$$softmax(x)=\begin{cases}\frac{e^{\omega_k\cdot X_i}}{1+\sum\limits_{k=1}^{K}e^{\omega_k\cdot X_i}} & k>0\\ \frac{1}{1+\sum\limits_{k=1}^{K}e^{\omega_k\cdot X_i}} & k=0\end{cases}$$

二分类时的$softmax$函数被称为$sigmoid$函数：

$$sigmoid(x)=\begin{cases}\frac{e^{\omega\cdot X_i}}{1+e^{\omega\cdot X_i}} & k=1\\ \frac{1}{1+e^{\omega\cdot X_i}} & k=0\end{cases}$$

参数$\omega_1,\omega_2,\ldots,\omega_k$可以同样地通过最大似然估计与梯度下降法/牛顿法/拟牛顿法求解。

最大熵模型

最优化问题的提出

对于数据集$D=\{(X_i,y_i)\},y_i=y_1,y_2,\ldots,y_K$而言，我们挑选最优模型的原则是：在满足模型本身的特征的前提下，模型在概率空间中的取值是等可能的。例如一个六面骰子，我们自然认为每面朝上的概率都是$\frac{1}{6}$；但如果我们从数据中发现，1点朝上出现的频率大概是一半，我们猜测可能是因为骰子不均匀，即存在特征：1点朝上的概率是$\frac{1}{2}$，那么另外5个面朝上的概率是多少呢？因为我们不知道其他信息，所以最好的推断就是：剩下5个面朝上的概率是等可能的，即$\frac{1}{10}$。

那么，在数学上如何表示等可能的模型呢？我们从决策树的章节引入了熵与条件熵的概念：当变量是均匀分布时，熵最大。因为我们最后希望给出的是概率模型，即$P(y|X)$，所以我们希望在给定X时，数据的熵最大，即最大化条件熵。其中条件熵的定义为：

$$\begin{aligned}H(P(y|X))&=-\sum_x\tilde{P}(x)\cdot H(P(y|X),X=x)\\&=-\sum_x\tilde{P}(x)\cdot\left(\sum_y P(y|X)\log P(y|X)\right)\\&=-\sum_{x,y}\tilde{P}(x)P(y|X)\log P(y|X)\end{aligned}$$

其中$\tilde{P}(x)=\frac{\nu(X=x)}{N}$，代表$X$的经验分布。注意这里的$\sum\limits_{x,y}$是对$X,y$内的所有去重取值求和。

接下来我们准备抽象“特征”这一概念。如果训练集中的某一现象$f_i(x,y)$确实是总体的某一特征，那么在训练集和总体中应当都具有这一现象，亦即在训练集和总体中，二者期望相同：

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

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

$$E_{\tilde{P}}(f_i) = E_P(f_i)$$

其中$f_i(x,y)$是形如

$$f_i(x,y) = \begin{cases} some\ value & x,y\ meet\ some\ condition(s) \\ other\ value & else \end{cases}$$

的函数。

于是我们抽象出了最优化问题，即在特征函数约束下，取条件熵最大的条件概率作为概率模型（当然，要保证条件概率之和为1）：

$$\begin{aligned} P(y|X) = \arg\max_{P(y|X)} \quad & -\sum_{x,y} \tilde{P}(x) P(y|X) \log P(y|X) \\ s.t. \quad & \sum_{x,y} (\tilde{P}(x,y) - \tilde{P}(x) P(y|X)) f_i(x,y) = 0 \\ & \sum_y P(y|X) = 1 \end{aligned}$$

将条件熵的负号提出来，等价于

$$\begin{aligned} P(y|X) = \arg\min_{P(y|X)} \quad & \sum_{x,y} \tilde{P}(x) P(y|X) \log P(y|X) \\ s.t. \quad & \sum_{x,y} (\tilde{P}(x,y) - \tilde{P}(x) P(y|X)) f_i(x,y) = 0 \\ & \sum_y P(y|X) = 1 \end{aligned}$$

模型求解

与SVM问题类似，我们采用Lagrange乘子法求解。首先写出Lagrange函数（为简便起见，记$P(y|X) = P$）：

$$\begin{aligned} L(P, \omega_0, \omega_i) = & \sum_{x,y} \tilde{P}(x) P(y|X) \log P(y|X) \\ & + \omega_0 (1 - \sum_y P(y|X)) \\ & + \sum_i \omega_i \sum_{x,y} (\tilde{P}(x,y) - \tilde{P}(x) P(y|X)) f_i(x,y) \end{aligned}$$

我们可以通过求解对偶问题：$\max\limits_{\omega_0,\omega_i} \min\limits_{P}\ L(P,\omega_0,\omega_i)$来求解原问题：$\min\limits_{P} \max\limits_{\omega_0,\omega_i}\ L(P,\omega_0,\omega_i)$（问题的凸性、对偶问题与原问题的关系详见SVM部分）。首先求$\min\limits_{P}\ L(P,\omega_0,\omega_i)$，令$\frac{\partial L(P,\omega_0,\omega_i)}{\partial P} = 0$：

$$
\begin{aligned}
\frac{\partial L(P,\omega_0,\omega_i)}{\partial P} &= \sum_{x,y}\tilde{P}(x)(\log P(y|X)+1)-\sum_{y}\omega_0+\sum_{i}\omega_i(-\sum_{x,y}\tilde{P}(x)f_i(x,y)) \\
&= \sum_{x,y}\tilde{P}(x)(\log P(y|X)+1)-\sum_{x,y}\tilde{P}(x)\omega_0-\sum_{x,y}\tilde{P}(x)\sum_{i}\omega_i f_i(x,y) \\
&= \sum_{x,y}\tilde{P}(x)\left(\log P(y|X)+1-\omega_0-\sum_{i}\omega_i f_i(x,y)\right) \\
&= 0
\end{aligned}
$$

解得

$$
P(y|X)=e^{\omega_0+\sum_i\omega_i f_i(x,y)-1}=\frac{e^{\sum_i\omega_i f_i(x,y)}}{e^{1-\omega_0}}
$$

考虑$\sum_y P(y|X)=1$，于是有

$$
\begin{aligned}
\sum_y P(y|X)=\frac{\sum_y e^{\sum_i\omega_i f_i(x,y)}}{e^{1-\omega_0}}&=1 \\
\Rightarrow e^{1-\omega_0}&=\sum_y e^{\sum_i\omega_i f_i(x,y)} \\
\Rightarrow P(y|X)&=\frac{\exp\left(\sum_i\omega_i f_i(x,y)\right)}{\sum_y\exp\left(\sum_i\omega_i f_i(x,y)\right)}
\end{aligned}
$$

令$Z_\omega(x)=\sum_y\exp\left(\sum_i\omega_i f_i(x,y)\right)$，记为规范化因子。

将解得的$P(y|X)$代入$L(P,\omega_0,\omega_i)$可以得到$\min\limits_P\ L(P,\omega_0,\omega_i)$：

$$
\begin{aligned}
\min_P\ L(P,\omega_0,\omega_i) &= \sum_{x,y}\tilde{P}(x)\frac{\exp\left(\sum_i\omega_i f_i(x,y)\right)}{Z_\omega(x)}\left(\sum_i\omega_i f_i(x,y)-\log Z_\omega(x)\right) \\
&\quad+\sum_{x,y}\left(\tilde{P}(x,y)-\tilde{P}(x)\frac{\exp\left(\sum_i\omega_i f_i(x,y)\right)}{Z_\omega(x)}\right)\sum_i\omega_i f_i(x,y) \\
&= -\sum_{x,y}\tilde{P}(x)\frac{\exp\left(\sum_i\omega_i f_i(x,y)\right)}{Z_\omega(x)}\log Z_\omega(x)+\sum_{x,y}\tilde{P}(x,y)\sum_i\omega_i f_i(x,y) \\
&= -\sum_x\tilde{P}(x)\log Z_\omega(x)\sum_y P(y|X)+\sum_{x,y}\tilde{P}(x,y)\sum_i\omega_i f_i(x,y) \\
&= \sum_{x,y}\tilde{P}(x,y)\sum_i\omega_i f_i(x,y)-\sum_x\tilde{P}(x)\log Z_\omega(x)
\end{aligned}
$$

对偶问题与极大似然估计的关系

有趣的是，对偶问题的解等价于最大熵模型的极大似然估计的解：

$$L_P(\omega)=\prod_{i=1}^{N}P(y_i|X_i)=\prod_{x,y}P(y|x)^{\nu(x,y)}$$

$$\begin{aligned}\omega&=\arg\max_{\omega}L_P(\omega)=\arg\max_{\omega}\prod_{x,y}P(y|x)^{\nu(x,y)}\\&=\arg\max_{\omega}\prod_{x,y}P(y|x)^{\frac{\nu(x,y)}{N}}=\arg\max_{\omega}\prod_{x,y}P(y|x)^{\tilde{P}(x,y)}\\&=\arg\max_{\omega}\sum_{x,y}\tilde{P}(x,y)\log P(y|x)\end{aligned}$$

将原问题的解代入：

$$\begin{aligned}\omega&=\arg\max_{\omega}\sum_{x,y}\tilde{P}(x,y)\log P(y|x)\\&=\arg\max_{\omega}\sum_{x,y}\tilde{P}(x,y)\left(\sum_i\omega_i f_i(x,y)-\log Z_\omega(x)\right)\\&=\arg\max_{\omega}\sum_{x,y}\tilde{P}(x,y)\sum_i\omega_i f_i(x,y)-\left(\sum_{x,y}\tilde{P}(x,y)\right)\log Z_\omega(x)\\&=\arg\max_{\omega}\sum_{x,y}\tilde{P}(x,y)\sum_i\omega_i f_i(x,y)-\sum_x\tilde{P}(x)\log Z_\omega(x)\end{aligned}$$

于是证明了：最大熵模型的极大似然估计与对偶问题是等价的。

对偶问题的求解——改进的迭代尺度法

我们接下来需要求$\arg\max_{\omega_0,\omega_i}\min_{P}\ L(P,\omega_0,\omega_i)$，即求

$$\arg\max_{\omega_0,\omega_i}\sum_{x,y}\tilde{P}(x,y)\sum_i\omega_i f_i(x,y)-\sum_x\tilde{P}(x)\log Z_\omega(x)$$

令$L(\omega)=\sum_{x,y}\tilde{P}(x,y)\sum_i\omega_i f_i(x,y)-\sum_x\tilde{P}(x)\log Z_\omega(x)$。我们将用改进的迭代尺度法（Improved Iterative Scaling, IIS）求解该问题。其主要思路是：

对于某次迭代前的参数为ω，假设迭代后的参数为$\omega+\delta$，我们选择迭代的方向为：

$$\arg\max_{\delta}\quad L(\omega+\delta)-L(\omega)$$

其中迭代后似然函数的下降量为：

$$L(\omega+\delta)-L(\omega)=\sum_{x,y}\tilde{P}(x,y)\sum_i\delta_i f_i(x,y)-\sum_x\tilde{P}(x)\log\frac{Z_{\omega+\delta}(x)}{Z_\omega(x)}$$

$$\begin{aligned}Z_{\omega+\delta}(x)&=\sum_y\exp\left(\sum_i(\omega_i+\delta_i)f_i(x,y)\right)=\sum_y\left(\exp\sum_i\omega_i f_i(x,y)\cdot\exp\sum_i\delta_i f_i(x,y)\right)\\&=\sum_y\left(Z_\omega(x)\cdot P(y|X)\cdot\exp\sum_i\delta_i f_i(x,y)\right)=Z_\omega(x)\sum_y P(y|X)\cdot\exp\sum_i\delta_i f_i(x,y)\end{aligned}$$

其中由$-\log\alpha\geq\alpha-1$知，$L(\omega+\delta)-L(\omega)$有下界$A(\delta|\omega)$：

$$
\begin{aligned}
A(\delta|\omega) &= \sum_{x,y}\tilde{P}(x,y)\sum_i \delta_i f_i(x,y) + \sum_x \tilde{P}(x)\left(1-\frac{Z_{\omega+\delta}(x)}{Z_\omega(x)}\right) \\
&= \sum_{x,y}\tilde{P}(x,y)\sum_i \delta_i f_i(x,y) + \sum_x \tilde{P}(x) - \sum_x \tilde{P}(x)\frac{Z_{\omega+\delta}(x)}{Z_\omega(x)} \\
&= \sum_{x,y}\tilde{P}(x,y)\sum_i \delta_i f_i(x,y) + 1 - \sum_x \tilde{P}(x)\sum_y P(y|X)\exp\sum_i \delta_i f_i(x,y)
\end{aligned}
$$

我们希望进一步降低$L(\omega+\delta)-L(\omega)$的下界，且希望且下界易于求导。于是利用Jensen不等式：当$\sum a_i = 1, a_i \in [0,1]$，且$f(x)$为凸函数时，有

$$
f\left(\sum_i a_i x_i\right) \le \sum_i a_i f(x_i)
$$

令$f=\exp, a_i = f_i(x,y), x_i = \delta_i$，有

$$
\exp\sum_i \delta_i f_i = \exp\sum_i \frac{f_i}{\sum f_i}(\delta_i \sum f_i) \le \sum_i \frac{f_i}{\sum f_i}\exp\left(\delta_i \sum f_i\right)
$$

代入$A(\delta|\omega)$：

$$
\begin{aligned}
A(\delta|\omega) \ge B(\delta|\omega) &= \sum_{x,y}\tilde{P}(x,y)\sum_i \delta_i f_i(x,y) + 1 - \sum_x \tilde{P}(x)\sum_y P(y|X)\exp\sum_i \delta_i f_i(x,y) \\
&= \sum_{x,y}\tilde{P}(x,y)\sum_i \delta_i f_i(x,y) + 1 - \sum_x \tilde{P}(x)\sum_y P(y|X)\sum_i \frac{f_i\exp\left(\delta_i\sum f_i\right)}{\sum f_i}
\end{aligned}
$$

改进的迭代尺度法不会同时更新δ的每一个分量，而是逐个更新δ_i。求新的下界$B(\delta|\omega)$对δ_i的导数：

$$
\frac{\partial B(\delta|\omega)}{\partial \delta_i} = \sum_{x,y}\tilde{P}(x,y)f_i(x,y) - \sum_x \tilde{P}(x)\sum_y P(y|X)f_i\exp\left(\delta_i\sum f_i\right)
$$

令$\frac{\partial B(\delta|\omega)}{\partial \delta_i}=0$即可由上式通过梯度下降法/牛顿法/拟牛顿法解得$\delta_i$。

有趣的是，如果对$\forall x,y$，$\sum f_i(x,y)$为常数C，那么可以显式地解出δ_i：

$$
\delta_i = \frac{1}{C}\log\frac{\sum\limits_{x,y}\tilde{P}(x,y)f_i(x,y)}{\sum\limits_x \tilde{P}(x)\sum\limits_y P(y|X)f_i(x,y)} = \frac{1}{C}\log\frac{\sum\limits_{x,y}\tilde{P}(x,y)f_i(x,y)}{\sum\limits_{x,y}\tilde{P}(x)P(y|X)f_i(x,y)} \equiv \frac{1}{C}\log\frac{E_{\tilde{P}}(f_i)}{E_P(f_i)}
$$

于是，我们得到了IIS算法的完整步骤：

1）初始化$\omega=0, k=0$，读取特征函数$f_i(x,y)$，精度ϵ；

2）计算经验分布$\tilde{P}(x,y), \tilde{P}(x)$与条件概率的初值$P_{\omega,0}(y|X)$；

3）对ω的每一个分量：

3-1）令$\delta_{i,k}$是方程

$$
\sum_{x,y}\tilde{P}(x,y)f_i(x,y) - \sum_x \tilde{P}(x)\sum_y P_{\omega,k}(y|X)f_i\exp\left(\delta_{i,k}\sum f_i\right)=0
$$

的解；

3-2）$\omega_i \to w_i + \delta_{i,k}$

4）若有$|\delta_{i,k}|<\epsilon, \forall i$，则结束程序；否则重复3）。

最大熵模型与逻辑斯蒂回归的联系

如上文，由Lagrange乘子法，我们得到了最优的概率模型：

$$P(y|X)=\frac{\exp\left(\sum\limits_i \omega_i f_i(x,y)\right)}{Z_\omega(x)}$$

其中$Z_\omega(x)$是归一化因子，ω由最大似然估计或求$对偶问题解出。

假设数据集D有K个类别，那么提出$K-1$个特征函数：

$$f_k(x,y)=\begin{cases}x_i & y=y_k\\0 & y\neq y_k\end{cases}$$

代入上式得到：

$$P(y=y_k|X)=\frac{\exp\sum\limits_k \omega_k x_k}{Z_\omega(x)},\quad k>0$$

由$\sum\limits_y P(y|X)=1$可解得$P(y=y_0|X)$与$Z_\omega(x)$，综合得到：

$$P(y=y_k|X)=\begin{cases}\frac{\exp\sum\limits_k \omega_k x_k}{1+\exp\sum\limits_k \omega_k x_k} & k>0\\\frac{1}{1+\exp\sum\limits_k \omega_k x_k} & k=0\end{cases}$$

这与多分类逻辑斯蒂回归的形式完全一致，也就是说，逻辑斯蒂回归正是在特征函数为（$k>0$）

$$f_k(x,y)=\begin{cases}x_i & y=y_k\\0 & y\neq y_k\end{cases}$$

时的最大熵模型。