法律声明

- 本课件包括演示文稿、示例、代码、题库、视频和声音等内容，小象学院和主讲老师拥有完全知识产权的权利；只限于善意学习者在本课程使用，不得在课程范围外向任何第三方散播。任何其他人或机构不得盗版、复制、仿造其中的创意及内容，我们保留一切通过法律手段追究违反者的权利。

- 课程详情请咨询
  - 微信公众号：小象
  - 新浪微博：ChinaHadoop

Logistic回归

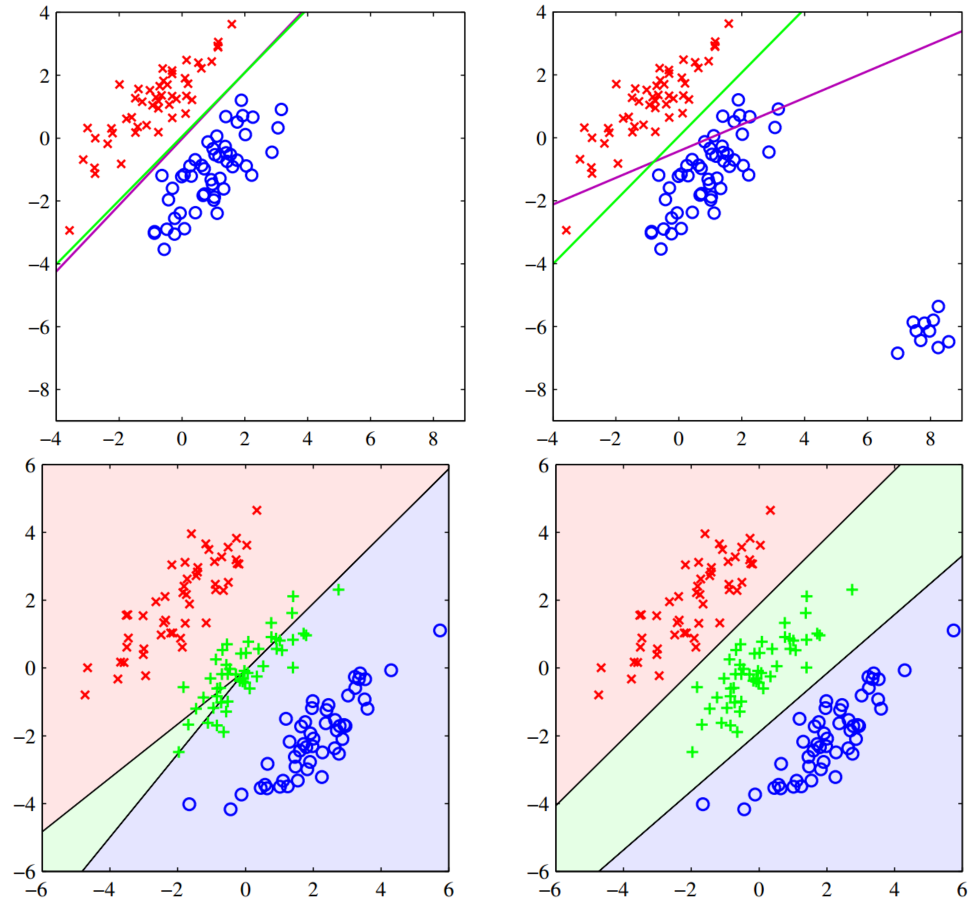
小象学院 ChinaHadoop.cn 邹博

主要内容

- Logistic回归
  - 分类问题的首选算法
- 多分类：Softmax回归
  - 目标函数
- 信息熵
  - 熵
  - 联合熵、条件熵、相对熵
  - 互信息

线性回归-Logistic回归

- 紫色：
  - 线性回归
- 绿色：
  - Logistic回归

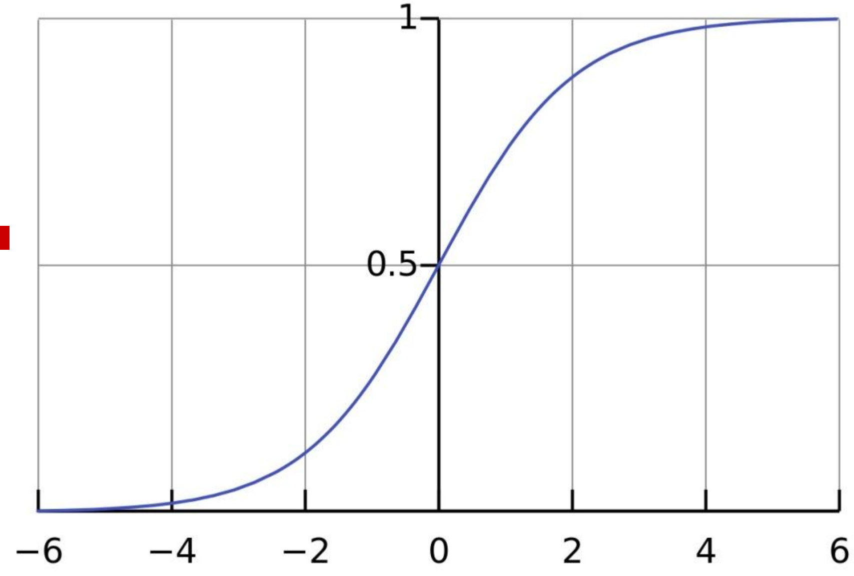


- 左侧：
  - 线性回归
- 右侧：
  - Softmax回归

Logistic回归

- Logistic/sigmoid函数

$$h_\theta(x) = g\left(\theta^T x\right) = \frac{1}{1+e^{-\theta^T x}}$$



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

$$g'(x) = \left(\frac{1}{1+e^{-x}}\right)' = \frac{e^{-x}}{\left(1+e^{-x}\right)^2}$$

$$= \frac{1}{1+e^{-x}} \cdot \frac{e^{-x}}{1+e^{-x}} = \frac{1}{1+e^{-x}} \cdot \left(1 - \frac{1}{1+e^{-x}}\right)$$

$$= g(x) \cdot (1 - g(x))$$

Logistic回归参数估计

- 假定：
$$\begin{aligned} P(y=1 \mid x;\theta) &= h_\theta(x) \\ P(y=0 \mid x;\theta) &= 1-h_\theta(x) \end{aligned}$$

$$p(y \mid x;\theta) = (h_\theta(x))^y \left(1-h_\theta(x)\right)^{1-y}$$

$$\begin{aligned} L(\theta) &= p(\vec{y} \mid X;\theta) \\ &= \prod_{i=1}^{m} p(y^{(i)} \mid x^{(i)};\theta) \\ &= \prod_{i=1}^{m} \left(h_\theta(x^{(i)})\right)^{y^{(i)}} \left(1-h_\theta(x^{(i)})\right)^{1-y^{(i)}} \end{aligned}$$

$$l(\theta)=\log L(\theta)=\sum_{i=1}^{m} y^{(i)}\log h\left(x^{(i)}\right)+\left(1-y^{(i)}\right)\log\left(1-h\left(x^{(i)}\right)\right)$$

对数似然函数

$$\frac{\partial l(\theta)}{\partial\theta_j}=\sum_{i=1}^{m}\left(\frac{y^{(i)}}{h\left(x^{(i)}\right)}-\frac{1-y^{(i)}}{1-h\left(x^{(i)}\right)}\right)\cdot\frac{\partial h\left(x^{(i)}\right)}{\partial\theta_j}$$

$$=\sum_{i=1}^{m}\left(\frac{y^{(i)}}{g\left(\theta^T x^{(i)}\right)}-\frac{1-y^{(i)}}{1-g\left(\theta^T x^{(i)}\right)}\right)\cdot\frac{\partial g\left(\theta^T x^{(i)}\right)}{\partial\theta_j}$$

$$=\sum_{i=1}^{m}\left(\frac{y^{(i)}}{g\left(\theta^T x^{(i)}\right)}-\frac{1-y^{(i)}}{1-g\left(\theta^T x^{(i)}\right)}\right)\cdot g\left(\theta^T x^{(i)}\right)\cdot\left(1-g\left(\theta^T x^{(i)}\right)\right)\cdot\frac{\partial\theta^T x^{(i)}}{\partial\theta_j}$$

$$=\sum_{i=1}^{m}\left(y^{(i)}\left(1-g\left(\theta^T x^{(i)}\right)\right)-\left(1-y^{(i)}\right)g\left(\theta^T x^{(i)}\right)\right)\cdot x_j^{(i)}$$

$$=\sum_{i=1}^{m}\left(y^{(i)}-g\left(\theta^T x^{(i)}\right)\right)\cdot x_j^{(i)}$$

参数的迭代

- Logistic回归参数的学习规则：

$$\theta_j := \theta_j + \alpha\left(y^{(i)} - h_\theta(x^{(i)})\right)x_j^{(i)}$$

- 比较上面的结果和线性回归的结论的差别：
  - 它们具有相同的形式！

Repeat until convergence {

$$\theta_j := \theta_j + \alpha \sum_{i=1}^{m}\left(y^{(i)} - h_\theta(x^{(i)})\right)x_j^{(i)}$$

}

Loop {

for i=1 to m, {

$$\theta_j := \theta_j + \alpha\left(y^{(i)} - h_\theta(x^{(i)})\right)x_j^{(i)}$$

}

}

对数线性模型

- 一个事件的几率odds，是指该事件发生的概率与该事件不发生的概率的比值。
- 对数几率：logit函数

$$
\begin{aligned}
P(y=1 \mid x;\theta) &= h_\theta(x) \\
P(y=0 \mid x;\theta) &= 1-h_\theta(x)
\end{aligned}
$$

$$
\log it(p) = \log\frac{p}{1-p} = \log\frac{h_\theta(x)}{1-h_\theta(x)} = \log\left(\frac{\frac{1}{1+e^{-\theta^T x}}}{\frac{e^{-\theta^T x}}{1+e^{-\theta^T x}}}\right) = \theta^T x
$$

Logistic回归的损失函数 $y_i \in \{0,1\}$

$$\hat{y}_i = \begin{cases} p_i & y_i = 1 \\ 1 - p_i & y_i = 0 \end{cases}$$

$$L(\theta) = \prod_{i=1}^{m} p_i^{y_i} (1 - p_i)^{1-y_i}$$

$$\Rightarrow l(\theta) = \sum_{i=1}^{m} \ln\left[p_i^{y_i} (1 - p_i)^{1-y_i}\right]$$

$$\xrightarrow{p_i = \frac{1}{1+e^{-f_i}}} l(\theta) = \sum_{i=1}^{m} \ln\left[\left(\frac{1}{1+e^{-f_i}}\right)^{y_i} \left(\frac{1}{1+e^{f_i}}\right)^{1-y_i}\right]$$

$$\therefore \ loss(y_i, \hat{y}_i) = -l(\theta)$$

$$= \sum_{i=1}^{m} \left[y_i \ln\left(1 + e^{-f_i}\right) + (1 - y_i) \ln\left(1 + e^{f_i}\right)\right]$$

Logistic回归的损失：$y_i \in \{-1,1\}$

$$L(\theta)=\prod_{i=1}^{m} p_i^{(y_i+1)/2}(1-p_i)^{-(y_i-1)/2} \Rightarrow l(\theta)=\sum_{i=1}^{m}\ln\left[p_i^{(y_i+1)/2}(1-p_i)^{-(y_i-1)/2}\right]$$

$$\xrightarrow{p_i=\frac{1}{1+e^{-f_i}}} l(\theta)=\sum_{i=1}^{m}\ln\left[\left(\frac{1}{1+e^{-f_i}}\right)^{(y_i+1)/2}\left(\frac{1}{1+e^{f_i}}\right)^{-(y_i-1)/2}\right]$$

$$\therefore\ loss(y_i,\hat{y}_i)=-l(\theta)$$

$$=\sum_{i=1}^{m}\left[\frac{1}{2}(y_i+1)\ln\left(1+e^{-f_i}\right)-\frac{1}{2}(y_i-1)\ln\left(1+e^{f_i}\right)\right]$$

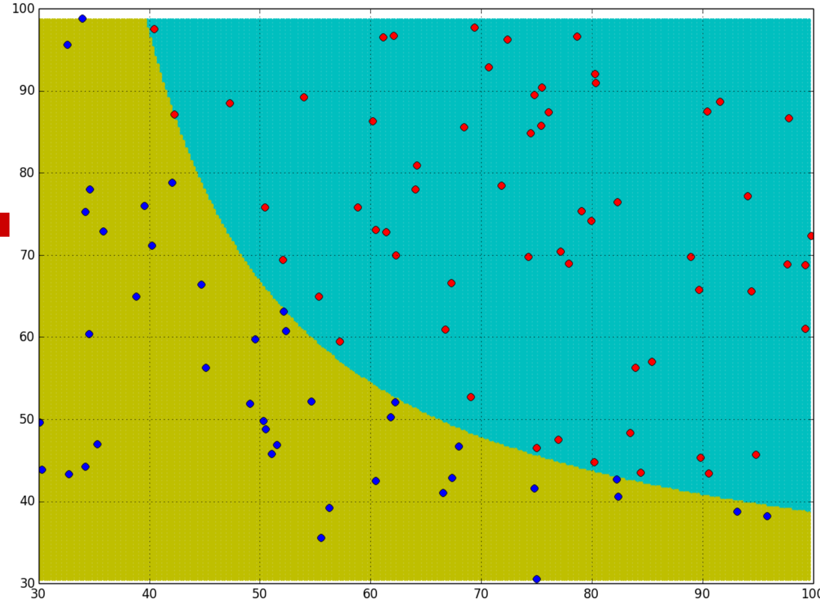
$$=\begin{cases}\sum_{i=1}^{m}\left[\ln\left(1+e^{-f_i}\right)\right] & y_i=1\\ \sum_{i=1}^{m}\left[\ln\left(1+e^{f_i}\right)\right] & y_i=-1\end{cases} \Rightarrow loss(y_i,\hat{y}_i)=\sum_{i=1}^{m}\left[\ln\left(1+e^{-y_i\cdot f_i}\right)\right]$$

$$y_i=\{-1,1\}$$

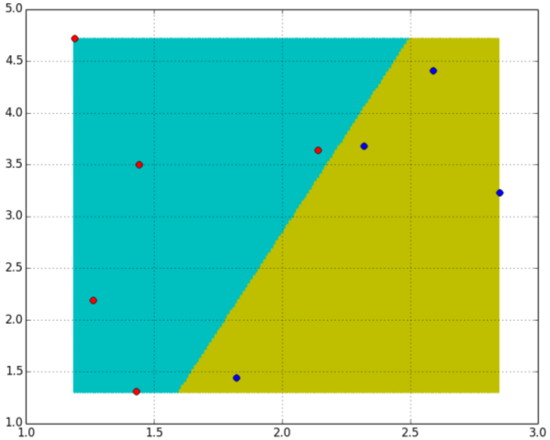
$$\hat{y}_i=\begin{cases}p_i & y_i=1\\ 1-p_i & y_i=-1\end{cases}$$

分类：Logistic回归

- 沿似然函数正梯度上升
- 维度提升



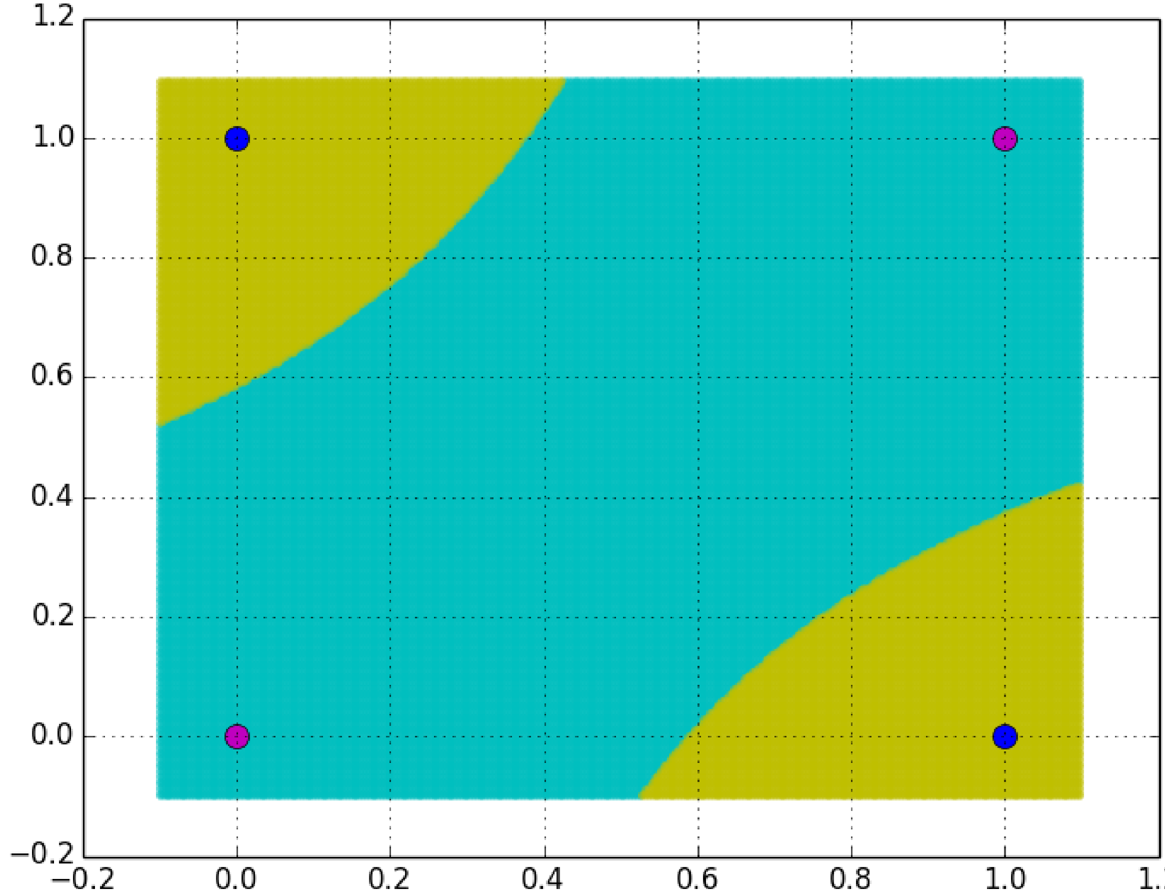

```
def logistic_regression(data, alpha, lamda):
    n = len(data[0]) - 1
    w = [0 for x in range(n)]
    w2 = [0 for x in range(n)]
    for times in range(10000):
        for d in data:
            for i in range(n):
                w2[i] = w[i] + alpha * (d[n] - h(w,d))*d[i] + lamda * w[i]
            for i in range(n):
                w[i] = w2[i]
            print w
    return w
```




```
def feature_whole(x1, x2, e):
    d = [1]
    for n in range(1,e+1):
        for i in range(n+1):
            d.append(pow(x1, n-i) * pow(x2, i))
    return d
```

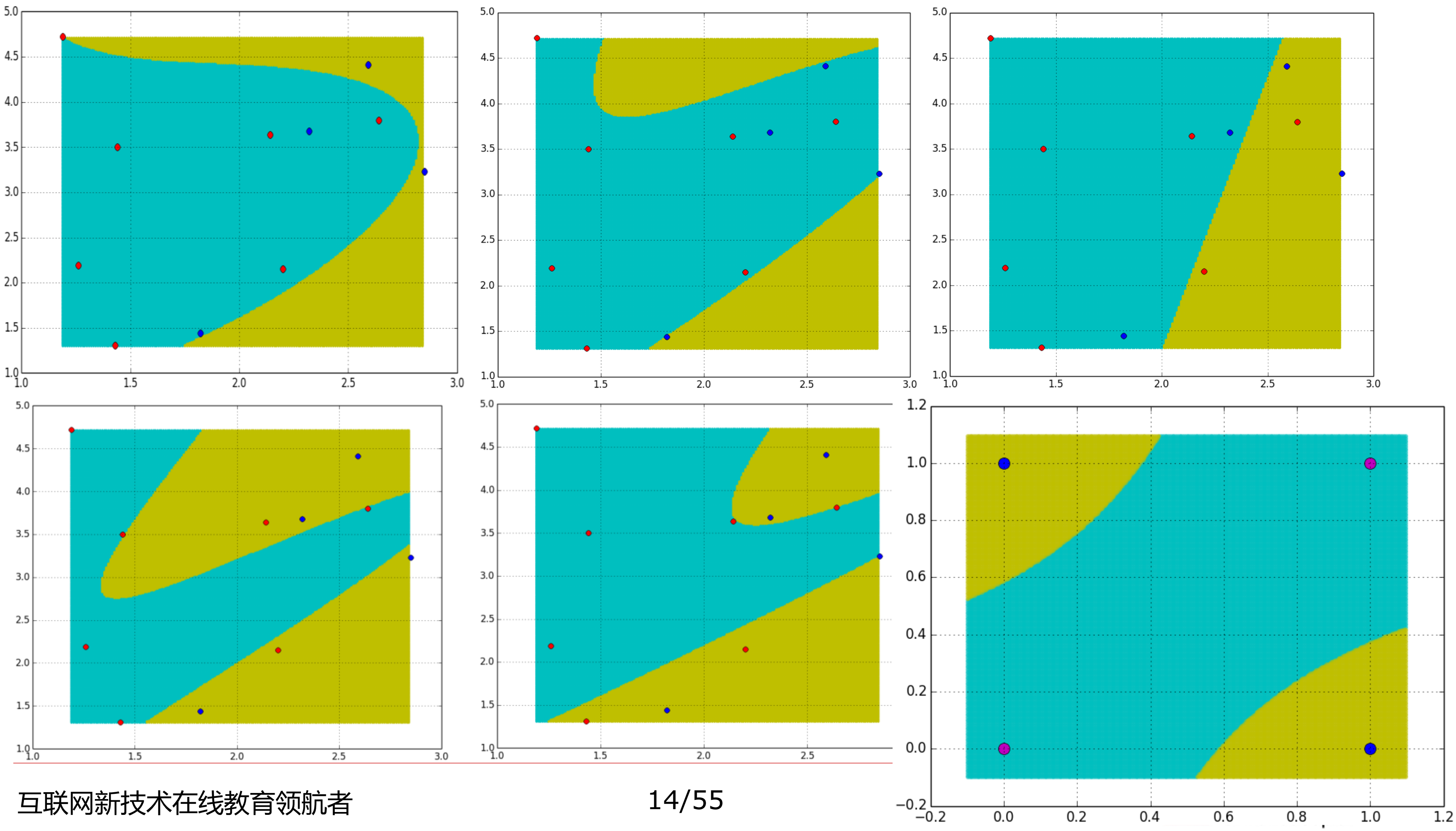
异或

x1	x2	y
0	0	0
0	1	1
1	0	1
1	1	0



数据升维：“选取特征”

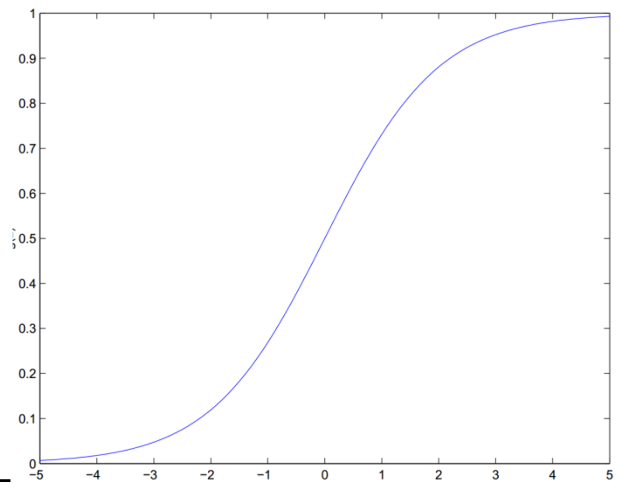
3	5	1
9	20	



广义线性模型Generalized Linear Model

- y不再只是正态分布，而是扩大为指数族中的任一分布；
- 变量x → g(x) → y
  - 连接函数g
- 连接函数g单调可导
  - 如Logistic回归中的 $g(z)=\dfrac{1}{1+e^{-z}}$
  - 拉伸变换：

$$g(z)=\frac{1}{1+e^{-\lambda z}}$$



广义线性模型

Common distributions with typical uses and canonical link functions

Distribution	Support of distribution	Typical uses	Link name	Link function	Mean function
Normal	real: $(-\infty, +\infty)$	Linear-response data	Identity	$\mathbf{X}\boldsymbol{\beta} = \mu$	$\mu = \mathbf{X}\boldsymbol{\beta}$
Exponential	real: $(0, +\infty)$	Exponential-response data, scale parameters	Inverse	$\mathbf{X}\boldsymbol{\beta} = \mu^{-1}$	$\mu = (\mathbf{X}\boldsymbol{\beta})^{-1}$
Gamma					
Inverse Gaussian	real: $(0, +\infty)$		Inverse squared	$\mathbf{X}\boldsymbol{\beta} = \mu^{-2}$	$\mu = (\mathbf{X}\boldsymbol{\beta})^{-1/2}$
Poisson	integer: $0, 1, 2, \ldots$	count of occurrences in fixed amount of time/space	Log	$\mathbf{X}\boldsymbol{\beta} = \ln(\mu)$	$\mu = \exp(\mathbf{X}\boldsymbol{\beta})$
Bernoulli	integer: $\{0, 1\}$	outcome of single yes/no occurrence	Logit	$\mathbf{X}\boldsymbol{\beta} = \ln\left(\frac{\mu}{1-\mu}\right)$	$\mu = \frac{\exp(\mathbf{X}\boldsymbol{\beta})}{1+\exp(\mathbf{X}\boldsymbol{\beta})} = \frac{1}{1+\exp(-\mathbf{X}\boldsymbol{\beta})}$
Binomial	integer: $0, 1, \ldots, N$	count of # of "yes" occurrences out of N yes/no occurrences			
Categorical	integer: $[0, K)$	outcome of single K-way occurrence			
	K-vector of integer: $[0, 1]$, where exactly one element in the vector has the value 1				
Multinomial	*K*-vector of integer: $[0, N]$	count of occurrences of different types (1 .. *K*) out of *N* total *K*-way occurrences			

Softmax回归

- K分类，第k类的参数为$\vec{\theta}_k$，组成二维矩阵$\theta_{k\times n}$
- 概率：$p(c=k\mid x;\theta)=\dfrac{\exp\left(\theta_k^T x\right)}{\sum_{l=1}^{K}\exp\left(\theta_l^T x\right)},\quad k=1,2,\cdots,K$
- 似然函数：$L(\theta)=\prod_{i=1}^{m}\prod_{k=1}^{K}p\left(c=k\mid x^{(i)};\theta\right)^{y_k^{(i)}}=\prod_{i=1}^{m}\prod_{k=1}^{K}\left(\dfrac{\exp\left(\theta_k^T x^{(i)}\right)}{\sum_{l=1}^{K}\exp\left(\theta_l^T x^{(i)}\right)}\right)^{y_k^{(i)}}$
- 对数似然：$J_m(\theta)=\ln L(\theta)=\sum_{i=1}^{m}\sum_{k=1}^{K}y_k^{(i)}\cdot\left(\theta_k^T x^{(i)}-\ln\sum_{l=1}^{K}\exp\left(\theta_l^T x^{(i)}\right)\right)$
- 随机梯度：$J(\theta)=\sum_{k=1}^{K}y_k\cdot\left(\theta_k^T x-\ln\sum_{l=1}^{K}\exp\left(\theta_l^T x\right)\right)$

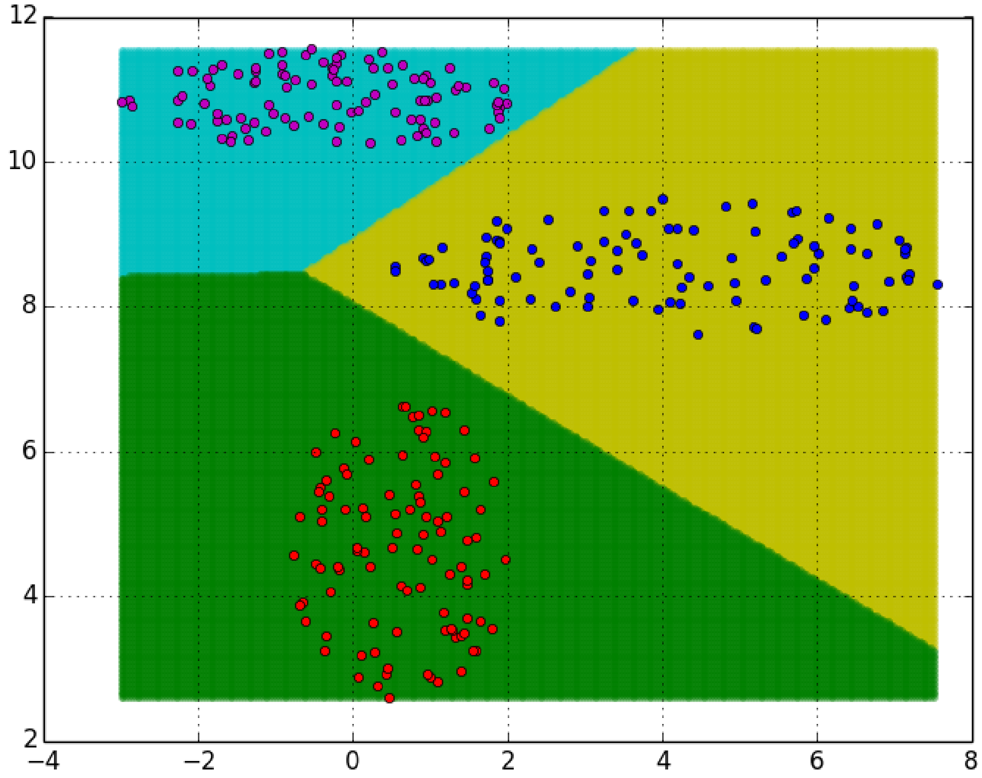
$$\frac{\partial J(\theta)}{\partial\theta_k}=\left(y_k-p\left(y_k\mid x;\theta\right)\right)\cdot x$$

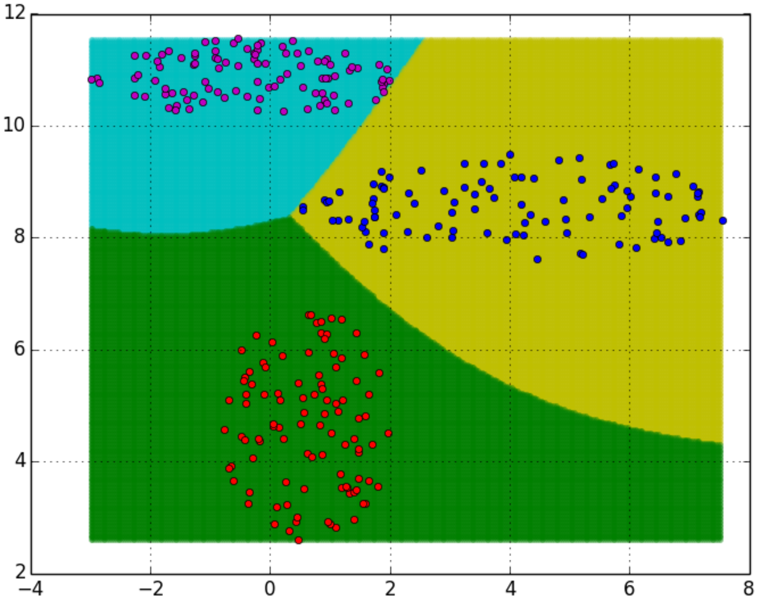
Code

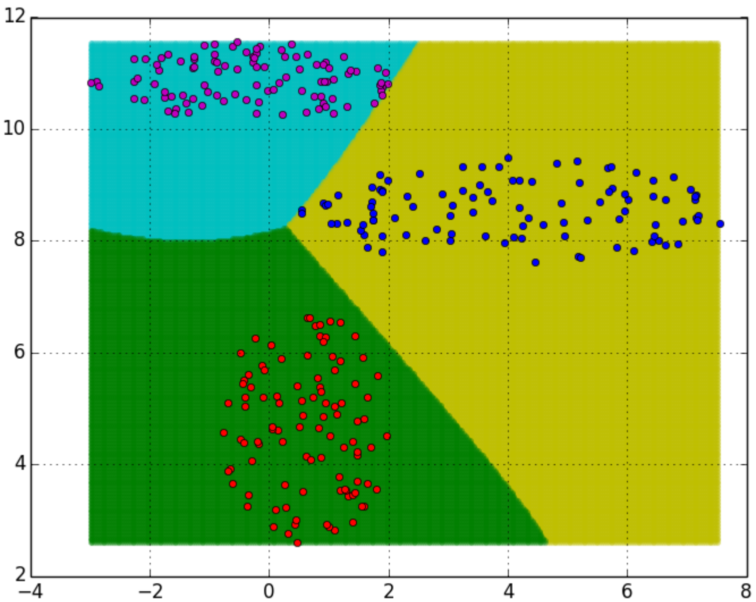
```python
def soft_max(data, K, alpha, lamda):
    n = len(data[0]) - 1    # 样本维度
    w = np.zeros((K, n))    # 当前权值：每个类别有各自的权值
    wNew = np.zeros((K, n))    # 临时权值：迭代过程中的权值
    for times in range(1000):
        for d in data:
            x = d[:-1]               # 输入向量
            for k in range(K):       # K：类别数目
                y = 0                # 期望输出(标量)
                if int(d[-1] + 0.5) == k:
                    y = 1
                p = predict(w, x, k)
                g = (y - p) * x      # 梯度(n维列向量)
                wNew[k] = w[k] + (alpha * g + lamda * w[k])
            w = wNew.copy()
    return w
```

Softmax分类

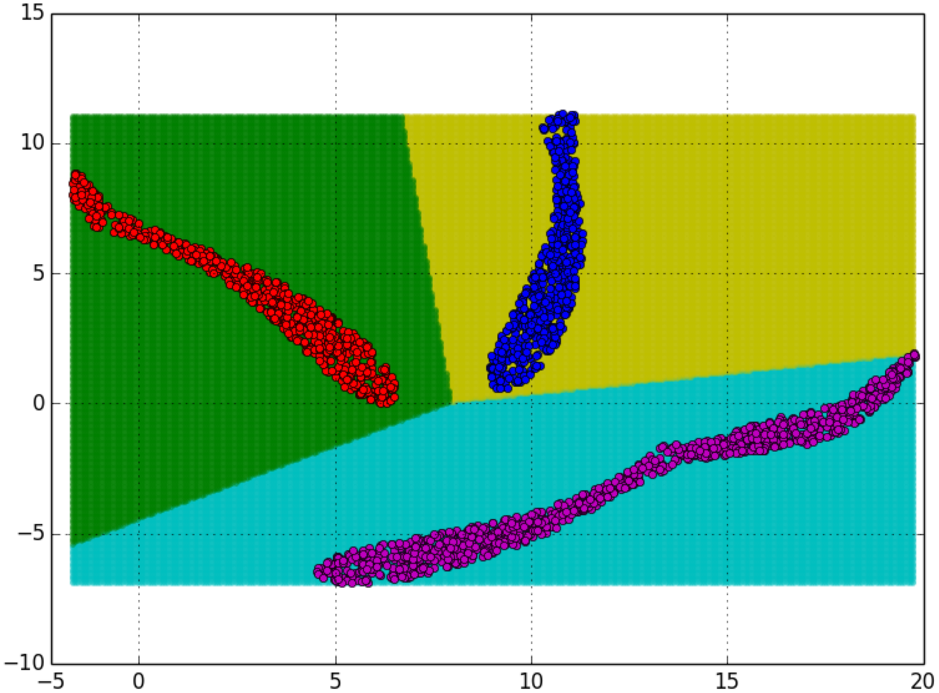
1 | 2/3



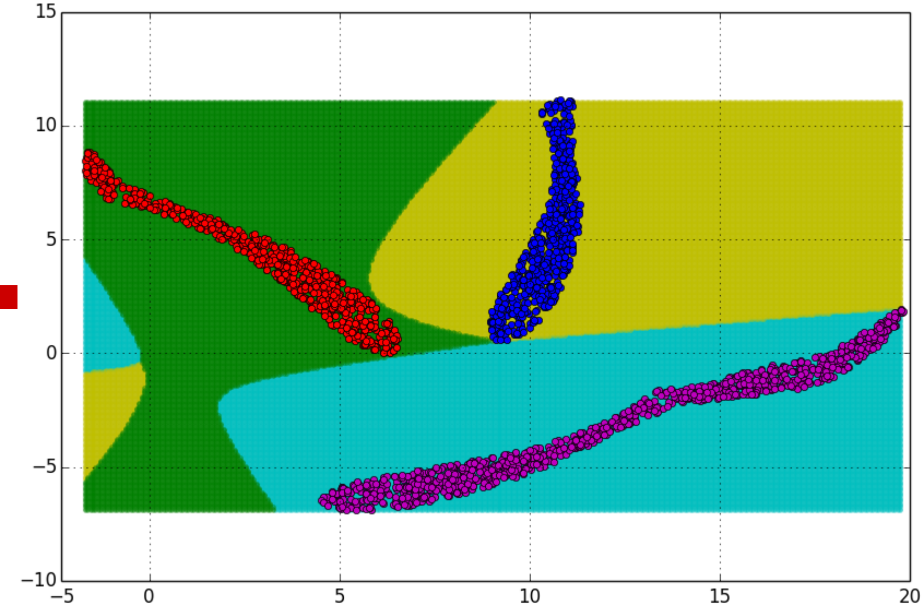




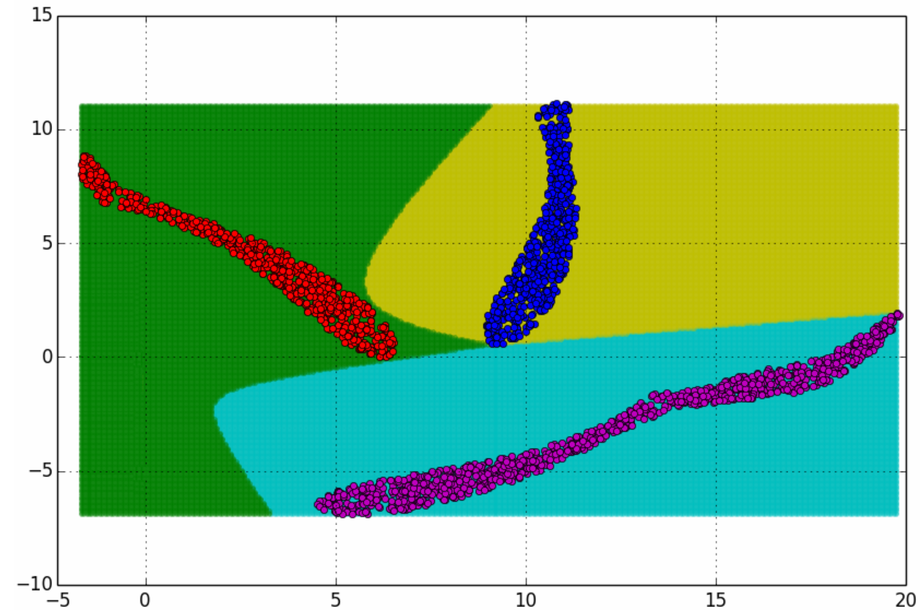
特征选择



15
10
5
0
−5
−10
−5
0
5
10
15
20



15
10
5
0
−5
−10
−5
0
5
10
15
20



15
10
5
0
−5
−10
−5
0
5
10
15
20

骰子问题

- 普通的一个骰子的某一次投掷，出现点5的概率是多大?
  - 等概率：各点的概率都是1/6
  - 对于“一无所知”的骰子，假定所有点数等概率出现是“最安全”的做法。
- 对给定的某个骰子，经过N次投掷后发现，点数的均值为2.71828，请问：再投一次出现点5的概率有多大?

带约束的优化问题

- 令6个面朝上的概率为(p1,p2…p6)，用向量**p**表示。
- 目标函数：$H(\vec{p}) = -\sum_{i=1}^{6} p_i \ln p_i$
- 约束条件：$\sum_{i=1}^{6} p_i = 1 \quad \sum_{i=1}^{6} i \cdot p_i = 2.71828$
- Lagrange函数：

$$L(\vec{p}, \lambda_1, \lambda_2) = -\sum_{i=1}^{6} p_i \ln p_i + \lambda_1 \left(1 - \sum_{i=1}^{6} p_i\right) + \lambda_2 \left(2.71828 - \sum_{i=1}^{6} i \cdot p_i\right)$$

- 求解：

$$\frac{\partial L}{\partial p_i} = -\ln p_i - 1 - \lambda_1 - i \cdot \lambda_2 \overset{\text{令}}{=\!=} 0$$

$$\Rightarrow p_i = e^{-1-\lambda_1 - i \cdot \lambda_2}$$

$$\Rightarrow \lambda_1 = -0.0787, \; \lambda_2 = 0.2808$$

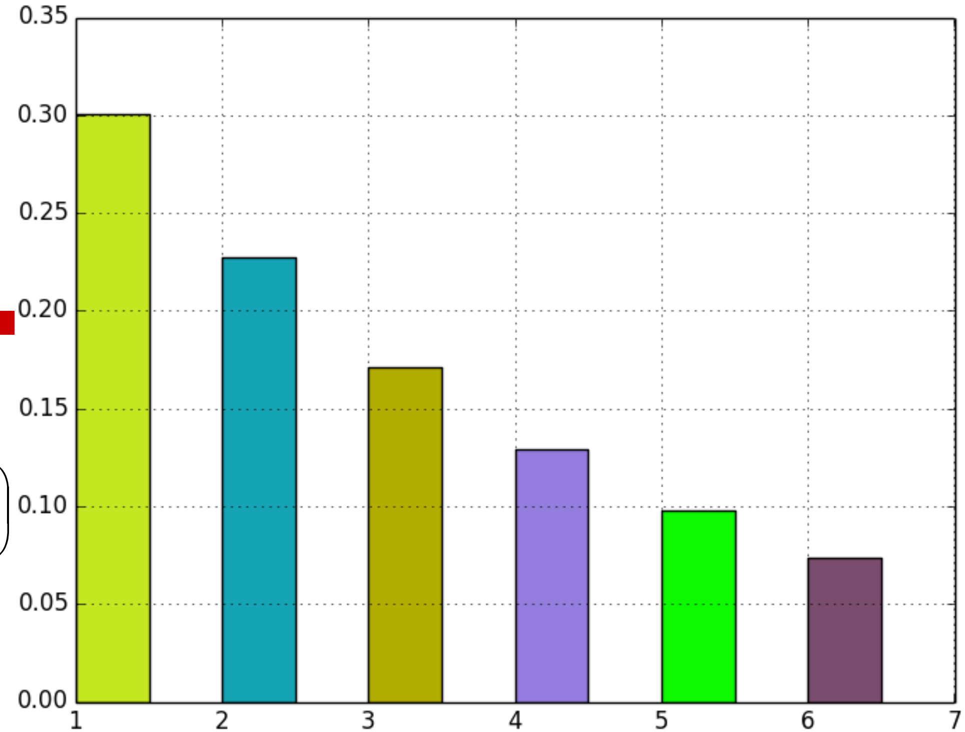
使用梯度下降计算Lagrange乘子

- 根据pi的解：$p_i = e^{-1-\lambda_1 - i\cdot\lambda_2}\ ,i = 1,2,\cdots 5,6$
- 构造目标函数并计算梯度：

$$J(\lambda) = \left(\sum_{i=1}^{6} p_i - 1\right)^2 + \left(\sum_{i=1}^{6} i\cdot p_i - E\right)^2$$

$$\Rightarrow \begin{cases} \dfrac{\partial J(\lambda)}{\partial \lambda_1} = -2\left(\sum_{i=1}^{6} p_i - 1\right)\cdot\left(\sum_{i=1}^{6} p_i\right) - 2\left(\sum_{i=1}^{6} i\cdot p_i - E\right)\cdot\left(\sum_{i=1}^{6} i\cdot p_i\right) \\ \dfrac{\partial J(\lambda)}{\partial \lambda_2} = -2\left(\sum_{i=1}^{6} p_i - 1\right)\cdot\left(\sum_{i=1}^{6} i\cdot p_i\right) - 2\left(\sum_{i=1}^{6} i\cdot p_i - E\right)\cdot\left(\sum_{i=1}^{6} i^2\cdot p_i\right) \end{cases}$$

预测结果

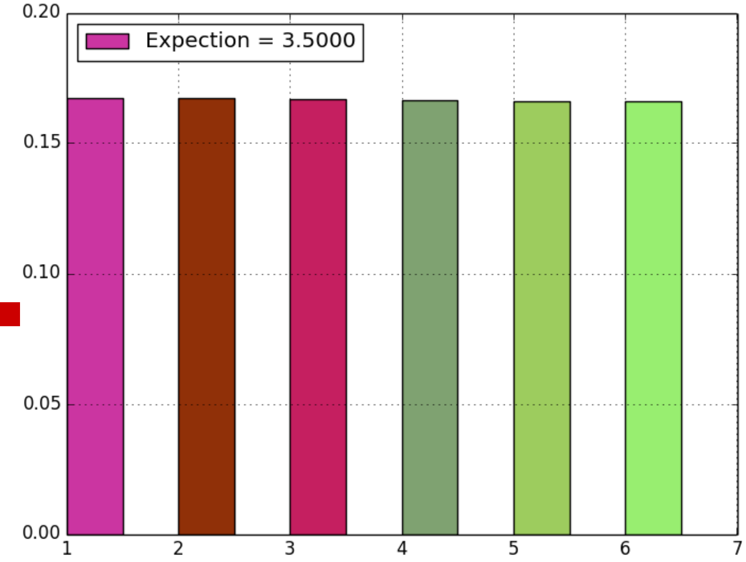


0.301 0.227 0.171 0.129 0.098 0.074

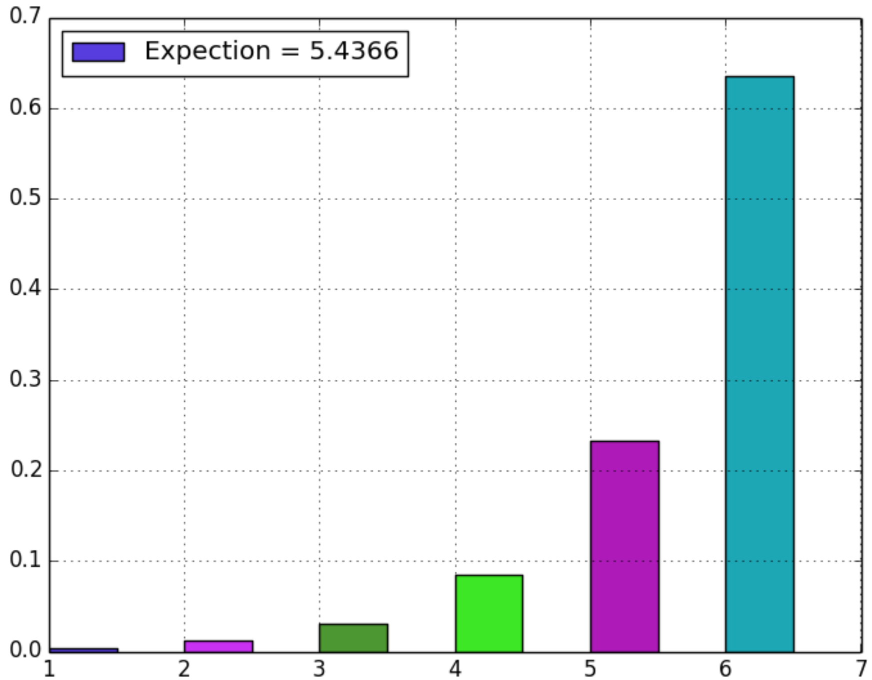
$$
\begin{cases}
\dfrac{\partial J(\lambda)}{\partial \lambda_1} = -2\left(\sum_{i=1}^{6} p_i - 1\right)\cdot\left(\sum_{i=1}^{6} p_i\right) - 2\left(\sum_{i=1}^{6} i\cdot p_i - E\right)\cdot\left(\sum_{i=1}^{6} i\cdot p_i\right) \\
\dfrac{\partial J(\lambda)}{\partial \lambda_2} = -2\left(\sum_{i=1}^{6} p_i - 1\right)\cdot\left(\sum_{i=1}^{6} i\cdot p_i\right) - 2\left(\sum_{i=1}^{6} i\cdot p_i - E\right)\cdot\left(\sum_{i=1}^{6} i^2\cdot p_i\right)
\end{cases}
$$

```python
if __name__ == "__main__":
    theta1, theta2 = 0, 0
    i = np.arange(1, 7)
    i2 = i ** 2
    alpha = 0.01
    beta = 0.1
    a = np.zeros(6)
    for times in range(5000):
        a = -np.arange(1, 7) * theta2 - theta1 - 1
        a = np.exp(a)
        f1 = np.dot(a, i)
        f2 = np.sum(a)
        theta1 += alpha * (f1-np.e) * f1 + beta * (f2 - 1) * f2
        theta2 += alpha * (f1-np.e) * np.dot(a, i2) + beta * (f2 - 1) * f1
    show(a/a.sum())
```

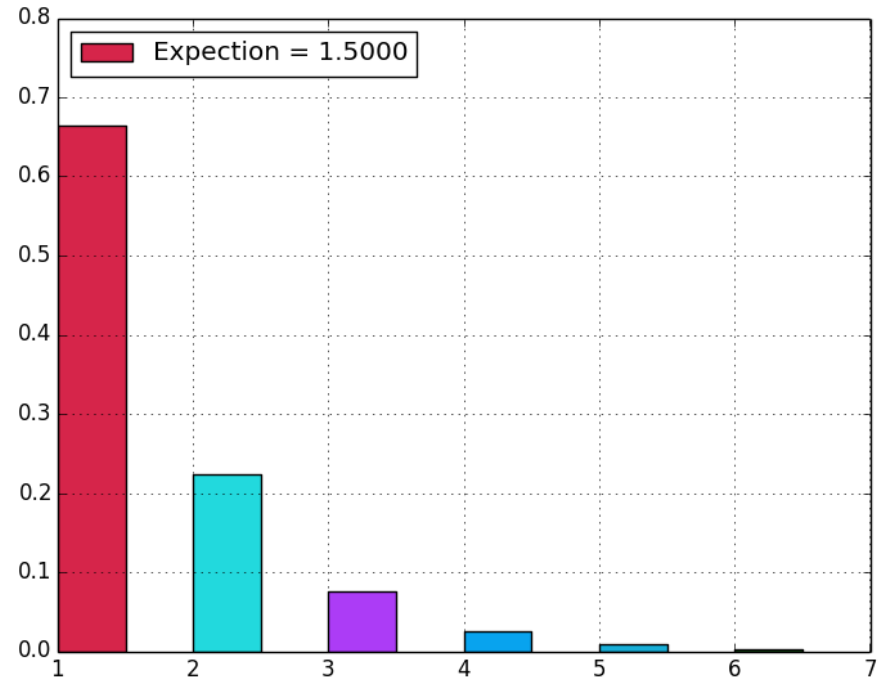
目标函数的有效性



Expection = 3.5000
0.20
0.15
0.10
0.05
0.00
1
2
3
4
5
6
7



Expection = 5.4366
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
1
2
3
4
5
6
7



Expection = 1.5000
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
1
2
3
4
5
6
7

定义信息量

- 原则：
  - 某事件发生的概率小，则该事件的信息量大。
  - 如果两个事件X和Y独立，即p(xy)=p(x)p(y)，假定X和Y的信息量分别为h(X)和h(Y)，则二者同时发生的信息量应该为h(XY)=h(X)+h(Y)。
- 定义随机变量X的概率分布为p(x)，从而定义X信息量：$h(x) = -\log_2 p(x)$
- 思考：事件X的信息量的期望如何计算呢？

熵

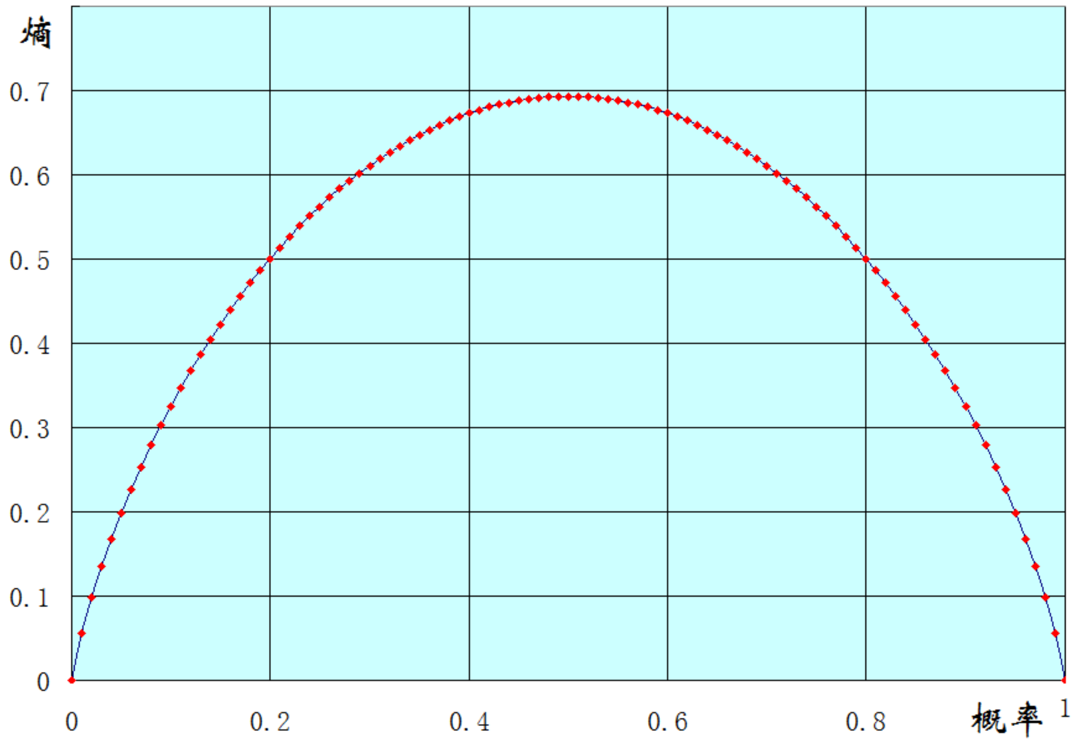
- 对随机事件的信息量求期望，得熵的定义：

$$H(X) = -\sum_{x \in X} p(x) \ln p(x)$$

  - 注：经典熵的定义，底数是2，单位是bit
  - 本例中，为分析方便使用底数e
  - 若底数是e，单位是nat(奈特)

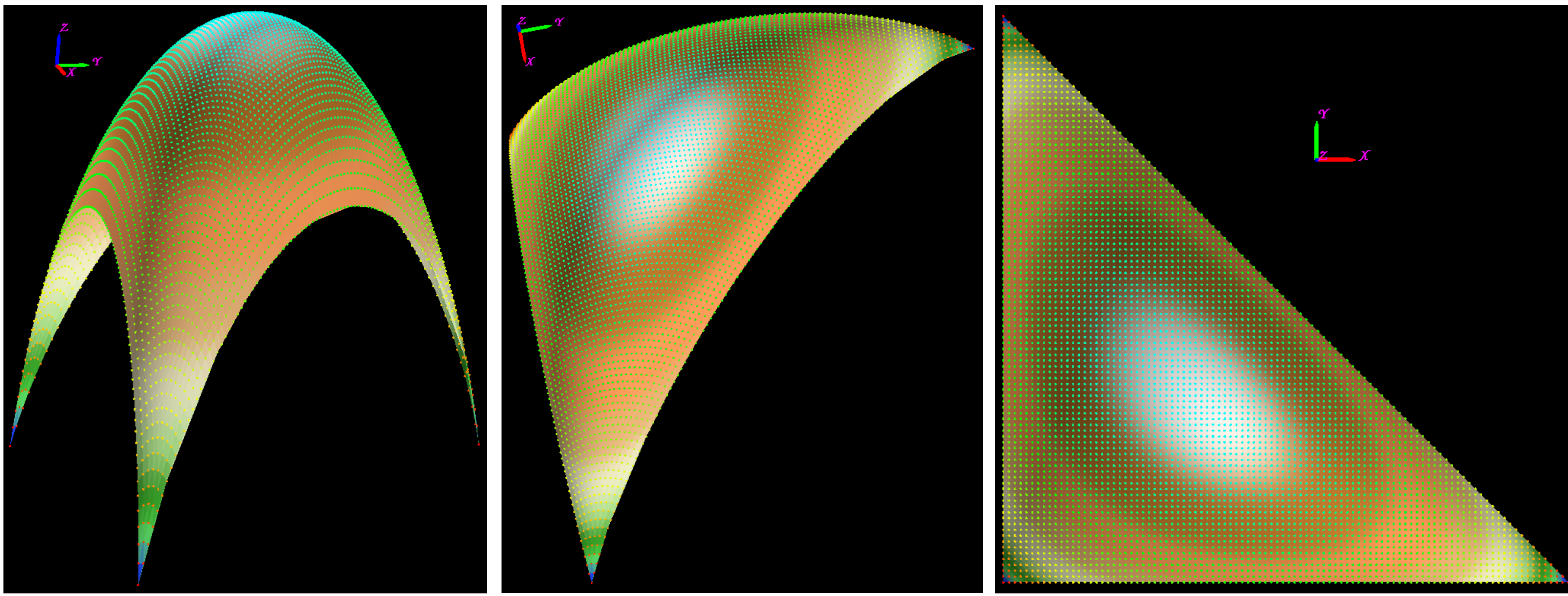
两点分布的熵

- 两点分布的熵 $H(X) = -\sum_{x \in X} p(x) \ln p(x) = -p \ln p - (1-p) \ln (1-p)$



继续思考：三点分布呢？

$$H(X) = -\sum_{x \in X} p(x) \ln p(x) = -p_1 \ln p_1 - p_2 \ln p_2 - (1 - p_1 - p_2) \ln (1 - p_1 - p_2)$$



公式推导 $N \to \infty \Rightarrow \ln N! \to N(\ln N - 1)$

$$H = \frac{1}{N}\ln\frac{N!}{\prod_{i=1}^{k} n_i!} = \frac{1}{N}\ln(N!) - \frac{1}{N}\sum_{i=1}^{k}\ln(n_i!)$$

$$\to (\ln N - 1) - \frac{1}{N}\sum_{i=1}^{k} n_i(\ln n_i - 1)$$

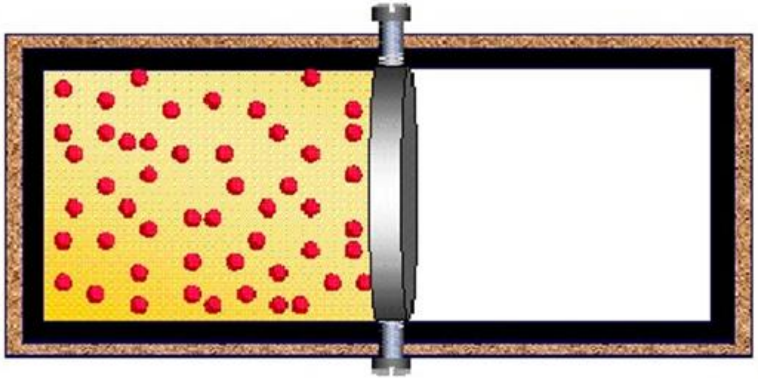
$$= \ln N - \frac{1}{N}\sum_{i=1}^{k} n_i \ln n_i = -\frac{1}{N}\left(\left(\sum_{i=1}^{k} n_i \ln n_i\right) - N\ln N\right)$$

$$= -\frac{1}{N}\sum_{i=1}^{k}(n_i \ln n_i - n_i \ln N) = -\frac{1}{N}\sum_{i=1}^{k}\left(n_i \ln\frac{n_i}{N}\right)$$

$$= -\sum_{i=1}^{k}\left(\frac{n_i}{N}\ln\frac{n_i}{N}\right) \to -\sum_{i=1}^{k}(p_i \ln p_i)$$

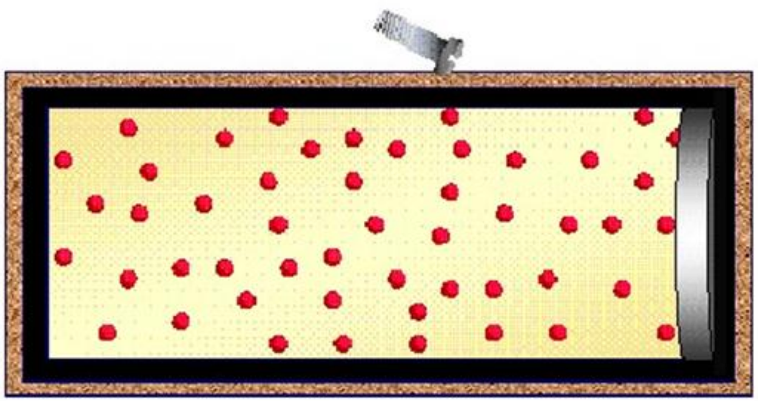
自封闭系统的运动总是倒向均匀分布

- 密封箱子中间放一隔板
- 隔板左边空间注入烟，右边真空



去掉隔板会怎样？

- 左边的烟就会自然（自发）地向右边扩散，最后均匀地占满整个箱体



均匀分布的信息熵

- 以离散分布为例：假定某离散分布可取N个值，概率都是1/N，计算该概率分布的熵。
- 解：概率分布律 $p_i = \frac{1}{N},\quad i = 1,2,\cdots,N$
- 计算熵：

$$H(p) = -\sum_{i=1}^{N} p_i \ln p_i = -\sum_{i=1}^{N} \frac{1}{N} \ln \frac{1}{N}$$

$$= \sum_{i=1}^{N} \frac{1}{N} \ln N = \ln N$$

- 思考：连续均匀分布的熵如何计算？

最大熵的理解 $0 \le H(X) \le \log|X|$

- 熵是随机变量**不确定性**的度量，不确定性越大，熵值越大；
  - 若随机变量退化成定值，熵最小：为0
  - 若随机分布为均匀分布，熵最大。
- 以上是无条件的最大熵分布，若有条件呢?
  - 最大熵模型
- 思考：若只给定**期望**和**方差**的前提下，最大熵的分布形式是什么？

引理：根据函数形式判断概率分布

- 正态分布的概率密度函数

$$p(x)=\frac{1}{\sqrt{2\pi}\sigma}e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

- 对数正态分布

$$\ln p(x)=\ln\frac{1}{\sqrt{2\pi}}-\ln\sigma-\frac{(x-\mu)^2}{2\sigma^2}=\alpha\cdot x^2+\beta\cdot x+\gamma$$

- 该分布的对数是关于随机变量X的二次函数
  - 根据计算过程的可逆性，若某对数分布能够写成随机变量二次形式，则该分布必然是正态分布。

举例

- Gamma分布的定义

$$f(x;\alpha,\beta)=\frac{\beta^{\alpha}}{\Gamma(\alpha)}\cdot x^{\alpha-1}e^{-\beta x}\quad,\quad x\geq 0(\text{常系数}\alpha,\beta>0)$$

- 对数形式

$$\ln f(x;\alpha,\beta)=\alpha\ln\beta+(\alpha-1)\ln x-\beta x-\ln\Gamma(\alpha)=A\cdot x+B\cdot\ln x+C$$

  - 若某连续分布的对数能够写成随机变量一次项和对数项的和，则该分布是Gamma分布。

- 注：
  - Gamma函数：$\Gamma(\alpha)=\int_0^{\infty}t^{\alpha-1}e^{-t}dt\qquad\Gamma(n)=(n-1)!$
  - Gamma分布的期望为：$E(X)=\frac{\alpha}{\beta}$

给定方差的最大熵分布

- 建立目标函数

$$\arg\max_{p(x)} H(X) = -\sum_x p(x)\ln p(x) \qquad s.t. \begin{cases} E(X) = \mu \\ Var(X) = \sigma^2 \end{cases}$$

- 使用方差公式化简约束条件

$$Var(X) = E(X^2) - E^2(X)$$
$$\Rightarrow E(X^2) = E^2(X) + Var(X) = \mu^2 + \sigma^2$$

- 显然，此问题为带约束的极值问题。
  - Lagrange乘子法

建立Lagrange函数，求驻点

$$\arg\max_{p(x)} H(X) = -\sum_x p(x)\ln p(x) \quad s.t. \begin{cases} E(X) = \mu \\ E(X^2) = \mu^2 + \sigma^2 \end{cases}$$

$$L(p) = -\sum_x p(x)\ln p(x) + \lambda_1 (E(X) - \mu) + \lambda_2 \left(E(X^2) - \mu^2 - \sigma^2\right)$$

$$= -\sum_x p(x)\ln p(x) + \lambda_1\left(\sum_x xp(x) - \mu\right) + \lambda_2\left(\sum_x x^2 p(x) - \mu^2 - \sigma^2\right)$$

$$\Rightarrow \frac{\partial L}{\partial p} = -\ln p(x) - 1 + \lambda_1 x + \lambda_2 x^2 \stackrel{\Delta}{==} 0 \Rightarrow \ln p(x) = \lambda_2 x^2 + \lambda_1 x - 1$$

- P(x)的对数是关于随机变量x的二次形式，所以，该分布p(x)必然是**正态分布**！

联合熵和条件熵

- 两个随机变量X，Y的联合分布，可以形成联合熵Joint Entropy，用H(X,Y)表示
- H(X,Y) – H(X)
  - (X,Y)发生所包含的熵，减去X单独发生包含的熵：在X发生的前提下，Y发生“新”带来的熵
  - 该式子定义为X发生前提下，Y的熵：
    - 条件熵H(Y|X)

推导条件熵的定义式

$$
\begin{aligned}
& H(X,Y) - H(X) \\
&= -\sum_{x,y} p(x,y)\log p(x,y) + \sum_{x} p(x)\log p(x) \\
&= -\sum_{x,y} p(x,y)\log p(x,y) + \sum_{x}\left(\sum_{y} p(x,y)\right)\log p(x) \\
&= -\sum_{x,y} p(x,y)\log p(x,y) + \sum_{x,y} p(x,y)\log p(x) \\
&= -\sum_{x,y} p(x,y)\log \frac{p(x,y)}{p(x)} \\
&= -\sum_{x,y} p(x,y)\log p(y \mid x)
\end{aligned}
$$

根据条件熵的定义式，可以得到

$$H(X,Y)-H(X)=-\sum_{x,y} p(x,y)\log p(y|x)$$

$$=-\sum_{x}\sum_{y} p(x,y)\log p(y|x)$$

$$=-\sum_{x}\sum_{y} p(x)p(y|x)\log p(y|x)$$

$$=-\sum_{x} p(x)\sum_{y} p(y|x)\log p(y|x)$$

$$=\sum_{x} p(x)\left(-\sum_{y} p(y|x)\log p(y|x)\right)$$

$$=\sum_{x} p(x)H\left(Y|X=x\right)$$

相对熵

- 相对熵，又称互熵，交叉熵，鉴别信息，Kullback熵，Kullback-Leible散度等
- 设p(x)、q(x)是X中取值的两个概率分布，则p对q的相对熵是

$$D(p\|q)=\sum_{x}p(x)\log\frac{p(x)}{q(x)}=E_{p(x)}\log\frac{p(x)}{q(x)}$$

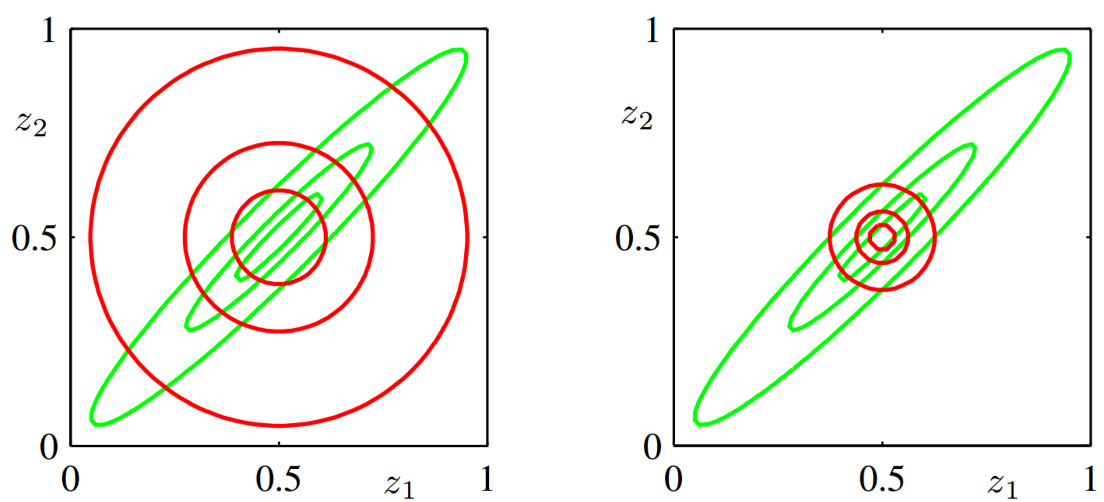
- 说明：
  - 相对熵可以度量两个随机变量的“距离”
    - 在“贝叶斯网络”、“变分推导”等章节会再次遇到
  - 一般的，$D(p\|q)\neq D(q\|p)$
  - $D(p\|q)\geq 0$、$D(q\|p)\geq 0$：凸函数中的Jensen不等式

思考

- 假定已知随机变量P，求相对简单的随机变量Q，使得Q尽量接近P
  - 方法：使用P和Q的K-L距离。
  - 难点：K-L距离是非对称的，两个随机变量应该谁在前谁在后呢？
- 假定使用KL(Q||P)，为了让距离最小，则要求在P为0的地方，Q尽量为0。会得到比较“窄”的分布曲线；
- 假定使用KL(P||Q)，为了让距离最小，则要求在P不为0的地方，Q也尽量不为0。会得到比较“宽”的分布曲线；

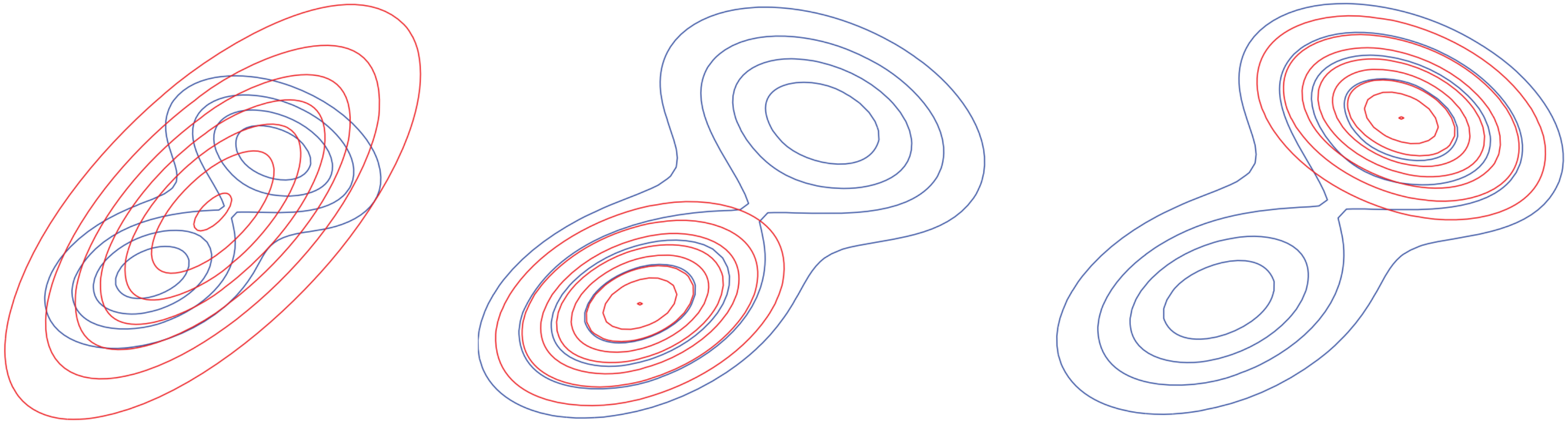
两个KL散度的区别

- 绿色曲线是真实分布p的等高线；红色曲线是使用近似$p(z_1,z_2)=p(z_1)p(z_2)$得到的等高线
  - 左：KL(p||q)：zero avoiding
  - 右：KL(q||p)：zero forcing



两个KL散度的区别

- 蓝色曲线是真实分布p的等高线；红色曲线是单模型近似分布q的等高线。
  - 左：KL(p||q)：q趋向于覆盖p
  - 中、右：KL(q||p)：q能够锁定某一个峰值



互信息

- 两个随机变量X，Y的互信息，定义为X，Y的联合分布和独立分布乘积的相对熵。

$$I(X,Y) = D\left(p(x,y) \| p(x)p(y)\right)$$
$$= \sum_{x,y} p(x,y) \log \frac{p(x,y)}{p(x)p(y)}$$

计算条件熵的定义式：H(Y)-I(X,Y)

$$
\begin{aligned}
&H(Y)-I(X,Y) \\
&=-\sum_{y} p(y)\log p(y)-\sum_{x,y} p(x,y)\log\frac{p(x,y)}{p(x)p(y)} \\
&=-\sum_{y}\left(\sum_{x} p(x,y)\right)\log p(y)-\sum_{x,y} p(x,y)\log\frac{p(x,y)}{p(x)p(y)} \\
&=-\sum_{x,y} p(x,y)\log p(y)-\sum_{x,y} p(x,y)\log\frac{p(x,y)}{p(x)p(y)} \\
&=-\sum_{x,y} p(x,y)\log\frac{p(x,y)}{p(x)} \\
&=-\sum_{x,y} p(x,y)\log p(y\,|\,x) \\
&=H(Y\,|\,X)
\end{aligned}
$$

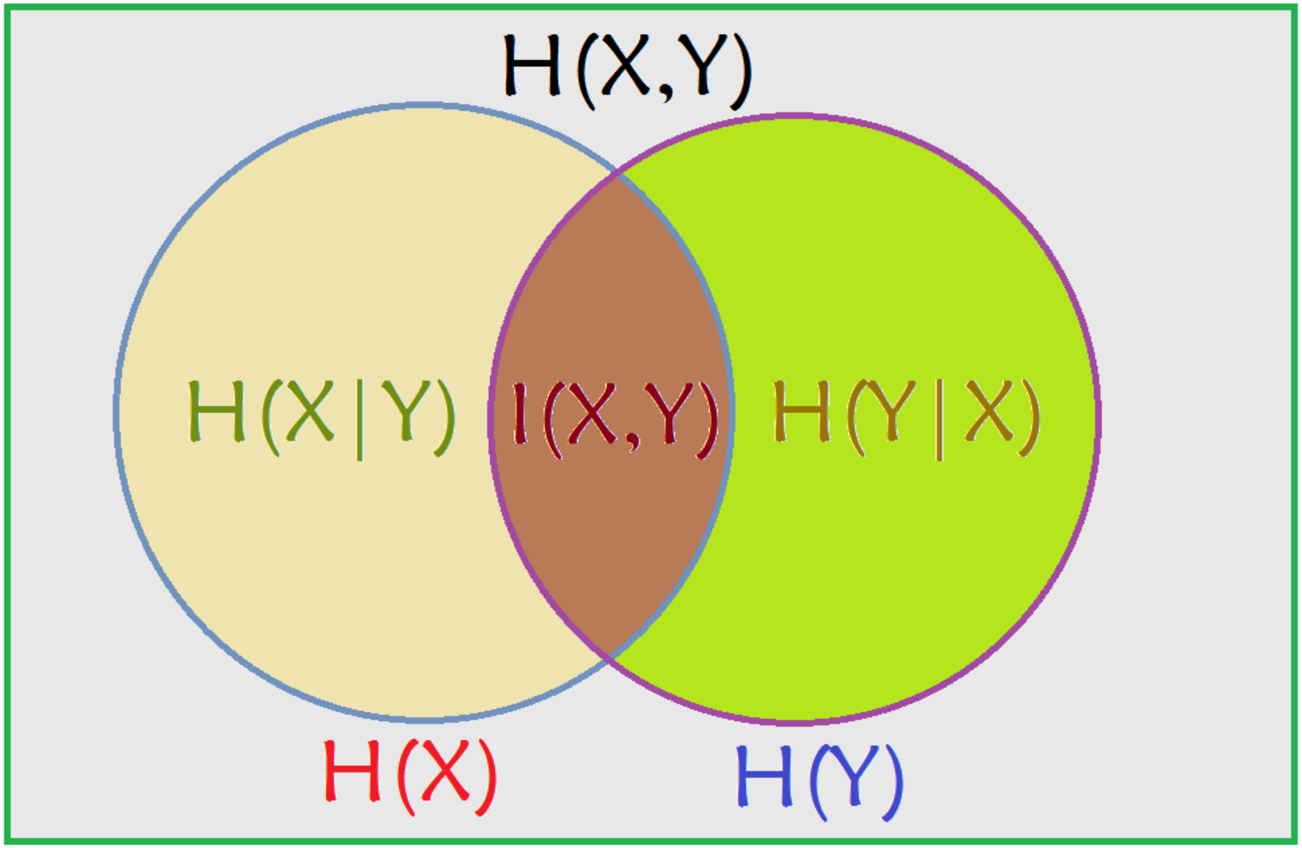
整理得到的等式

- $H(Y|X) = H(X,Y) - H(X)$
  - 条件熵定义
- $H(Y|X)= H(Y) - I(X,Y)$
  - 根据互信息定义展开得到
  - 有些文献将$I(X,Y)=H(Y) – H(Y|X)$作为互信息的定义式
- 对偶式
  - $H(X|Y)= H(X,Y) - H(Y)$
  - $H(X|Y) = H(X) - I(X,Y)$
- $I(X,Y)= H(X) + H(Y) - H(X,Y)$
  - 有些文献将该式作为互信息的定义式
- 试证明：$H(X|Y) \le H(X)$， $H(Y|X) \le H(Y)$

互信息：I(X,Y)=H(X)+H(Y)-H(X,Y)

$$
\begin{aligned}
I(X,Y) &= H(X)+H(Y)-H(X,Y) \\
&= \left(-\sum_x p(x)\log p(x)\right)+\left(-\sum_y p(y)\log p(y)\right)-\left(-\sum_{x,y} p(x,y)\log p(x,y)\right) \\
&= \left(-\sum_x \left(\sum_y p(x,y)\right)\log p(x)\right)+\left(-\sum_y \left(\sum_x p(x,y)\right)\log p(y)\right)+\sum_{x,y} p(x,y)\log p(x,y) \\
&= -\sum_{x,y} p(x,y)\log p(x)-\sum_{x,y} p(x,y)\log p(y)+\sum_{x,y} p(x,y)\log p(x,y) \\
&= \sum_{x,y} p(x,y)\left(\log p(x,y)-\log p(x)-\log p(y)\right) \\
&= \sum_{x,y} p(x,y)\left(\log \frac{p(x,y)}{p(x)p(y)}\right)
\end{aligned}
$$

强大的Venn图：帮助记忆



思考题：天平与假币

- 有12枚硬币，其中有1枚是假币，但不知道是重还是轻。现给定一架没有砝码的天平，问**至少**需要多少次称量才能找到这枚假币？
  - 答：3次。
  - 如何称量？如何证明？

总结和思考

- Logistic/Softmax回归是实践中解决分类问题的最重要方法。
  - 方法简单、容易实现、效果良好、易于解释
  - 不止是分类：推荐系统
- 思考
  - Logistic回归的目标函数，可否用相对熵解释？
  - 信息熵有什么用？

作业

- 推导Logistic/Softmax回归的梯度。
- 思考广义线性回归中link function的作用。
- 自己实现一个Softmax回归，并用于鸢尾花数据的分类。

参考文献

- Prof. Andrew Ng. *Machine Learning*. Stanford University
- 李航，统计学习方法，清华大学出版社，2012

我们在这里

- http://wenda.ChinaHadoop.cn
  - 视频/课程/社区
- 微博
  - @ChinaHadoop
  - @邹博_机器学习
- 微信公众号
  - 小象学院
  - 大数据分析挖掘

感谢大家！

恳请大家批评指正！