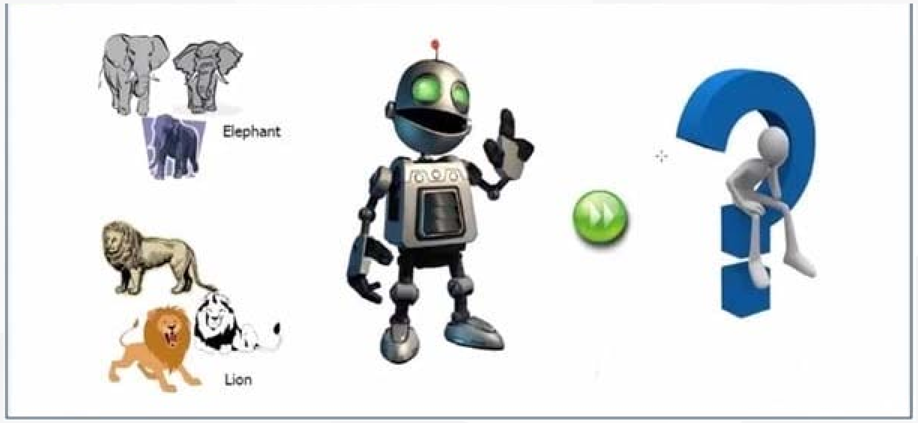
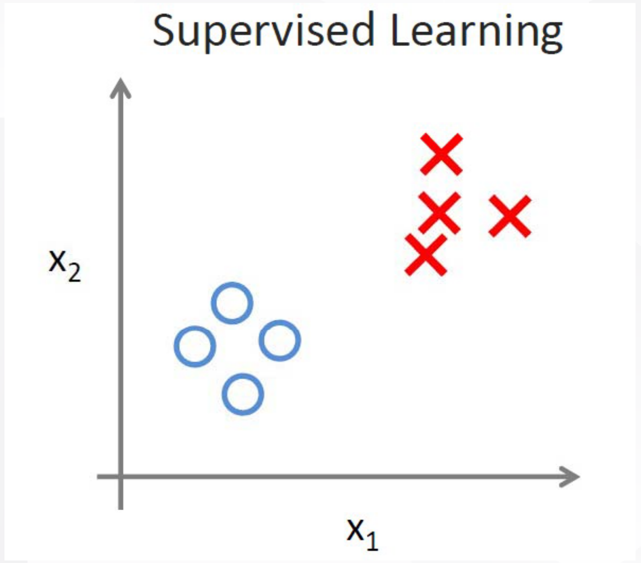
第六章: 有监督学习方法



什么是有监督学习



■有监督学习：从**有标记**的训练数据中学习推断函数。

■有监督学习算法分析训练数据，产生推断函数。推断函数能够对新的样本进行预测。

■ 最优的情形：算法能够准确地对**没见过的样本**进行正确地分类。

■**目标函数（target function）**：$Y=f(X)$ 或 $P(Y|X)$

有监督学习方法一: 产生式模型 Generative Model

- 首先对联合分布的进行推断:

$$p(x, y) = p(y)\,p(x \mid y)$$

- 然后使用贝叶斯定理来计算条件分布$p(y|x)$

$$p(y \mid x) = \frac{p(x, y)}{p(x)} = \frac{p(y)\,p(x \mid y)}{\int p(y)\,p(x \mid y)dy}$$

- 利用条件概率密度来预测。

- 例子:确定某人所说语言的类别（英语，法语，汉语）
  - 产生式模型的做法首先学习每种语言，然后确定所说的语主属于哪种语言。

有监督学习方法二: 判别式模型Discriminative Model

■直接估计条件概率分布$P(y|x)$或条件概率密度函数$p(y|x)$。

■根据估计的函数确定输出 。

■例子:确定某人所说语言的类别（英语，法语，汉语）？

•不学习任何语言的情况下确定语言差异——这是一项容易得多的任务!

有监督学习方法三: 判别函数

- 寻找一个函数$f(x)$， 将每个输入直接映射到目标输出。

  例如： 学习二值 分类器时，$f(.)$是一个二值函数
  - $f=1$ 表示第一类C_1 , $f=0$ 或 (-1) 表示第二类C_2
  - 例如: $f(\boldsymbol{x})=\text{sign}(d(\boldsymbol{x}))$
- 概率不起直接作用
  - 不能直接获取后验概率。
  - f 通常旨在近似条件分布$p(y|x)$。

回归方法

回归

- 输入: N i.i.d 训练样本 $(\boldsymbol{x}^i, y^i) \in X \times R,\ i = 1, \cdots, N$
- 目标函数: $f \in \mathcal{F}$
- 损失函数: $L(f; x, y) = (f(x) - y)^2$
- 期望风险: $\int (f(x) - y)^2 dP(x, y)$
- 如果f是关于x的线性函数，最优化问题为: $\min_{\boldsymbol{w}} J(\boldsymbol{w}) = \sum_{i=1}^{N} (\boldsymbol{w}^T \boldsymbol{x}^i - y^i)^2$

$$\frac{\partial J(\boldsymbol{w})}{\partial w_j} = 2\sum_{i=1}^{N} x_j^i (\boldsymbol{w}^T \boldsymbol{x}^i - y^i) \xrightarrow{\text{梯度等于0}} \boldsymbol{w}^* = (\boldsymbol{X}^T \boldsymbol{X})^{-1} \boldsymbol{X}^T \boldsymbol{y}$$

$$\boldsymbol{X}^T = (\boldsymbol{x}^1, \boldsymbol{x}^2, \cdots, \boldsymbol{x}^N),\ \boldsymbol{y}^T = (y^1, y^2, \cdots, y^N).$$

最小二乘法（Least Mean Squares Algorithm)

■最优化问题:

$$\min_{w} J(\boldsymbol{w}) = \sum_{i=1}^{N} (\boldsymbol{w}^T \boldsymbol{x}^i - y^i)^2$$

■梯度下降:

$$\frac{\partial J(\boldsymbol{w})}{\partial w_j} = 2\sum_{i=1}^{N} x_j^i (\boldsymbol{w}^T \boldsymbol{x}^i - y^i)$$

■更新规则:

$$\boxed{w_j = w_j - 2\alpha \sum_{i=1}^{N} (\boldsymbol{w}^T \boldsymbol{x}^i - y^i) x_j^i,\ \alpha > 0}$$

批梯度下降，BGD
Batch Gradient Descent

■优点:

●一次迭代是对所有样本进行计算，此时利用矩阵进行操作，实现了并行。

●由全数据集确定的方向能够更好地代表样本总体，从而更准确地朝向极值所在的方向。当目标函数为凸函数时，BGD一定能够得到全局最优。

■缺点:

●当样本数目 *N*很大时，每迭代一步都需要对所有样本计算，训练过程会很慢。

最小二乘法（Least Mean Squares Algorithm)

■最优化问题: $\min_{w} J(\boldsymbol{w}) = \sum_{i=1}^{N} (\boldsymbol{w}^T \boldsymbol{x}^i - y^i)^2$

■梯度下降: $\dfrac{\partial J(\boldsymbol{w})}{\partial w_j} = 2\sum_{i=1}^{N} x_j^i (\boldsymbol{w}^T \boldsymbol{x}^i - y^i)$

■更新规则:

$$\boxed{w_j = w_j - 2\alpha \sum_{i=1}^{N} (\boldsymbol{w}^T \boldsymbol{x}^i - y^i) x_j^i, \ \alpha > 0}$$

批梯度下降，**BGD**
Batch Gradient Descent

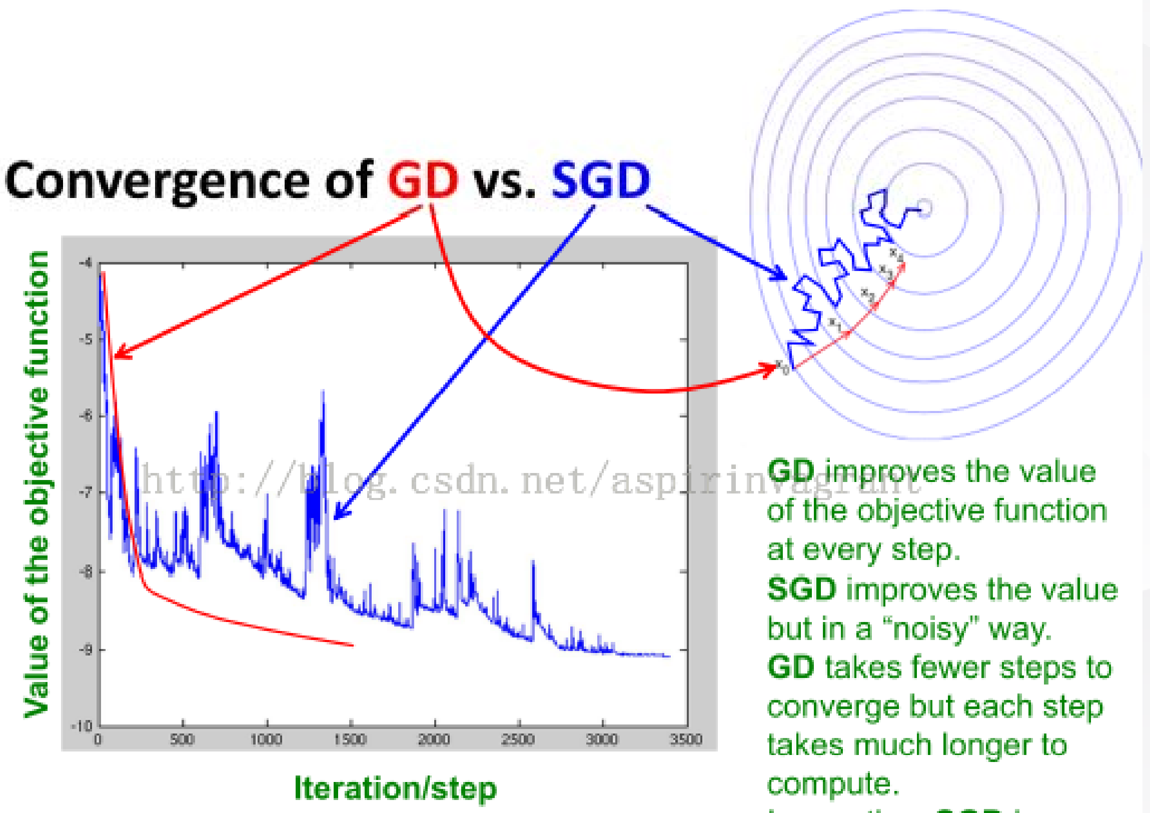
$$\boxed{w_j = w_j - 2\alpha (\boldsymbol{w}^T \boldsymbol{x}^i - y^i) x_j^i, \ \alpha > 0}$$

随机梯度下降，**SGD**
Stochastic Gradient Descent

$$\boxed{\boldsymbol{w} = \boldsymbol{w} - 2\alpha \boldsymbol{X}^T \boldsymbol{b}, \ \text{where } b_i = \boldsymbol{w}^T \boldsymbol{x}^i - y^i, \ \boldsymbol{b} = (b_1, b_2, \cdots, b_N)^T}$$

SGD 与BGD/GD对比

- Convergence of GD vs. SGD

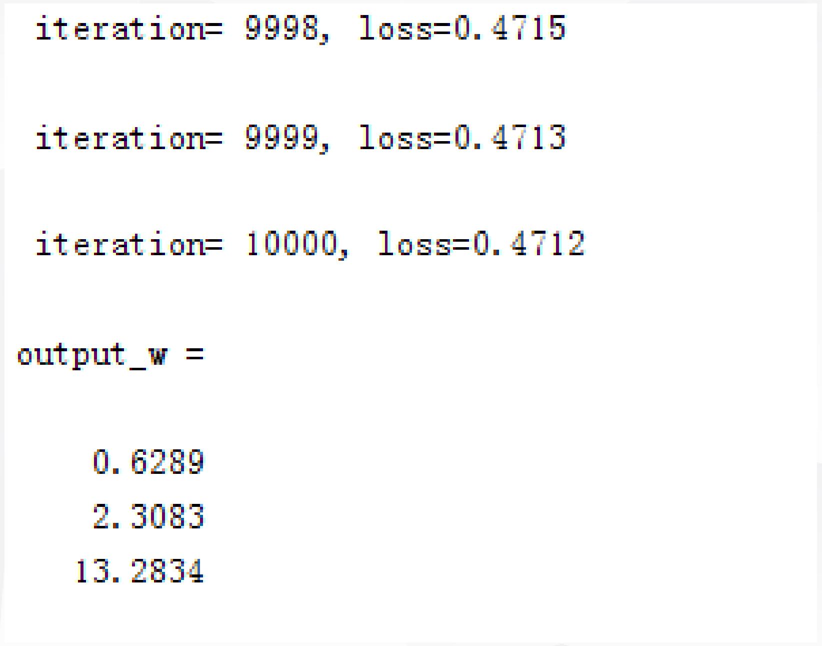


GD improves the value of the objective function at every step.
SGD improves the value but in a "noisy" way.
GD takes fewer steps to converge but each step takes much longer to compute.
In practice, **SGD** is much faster!

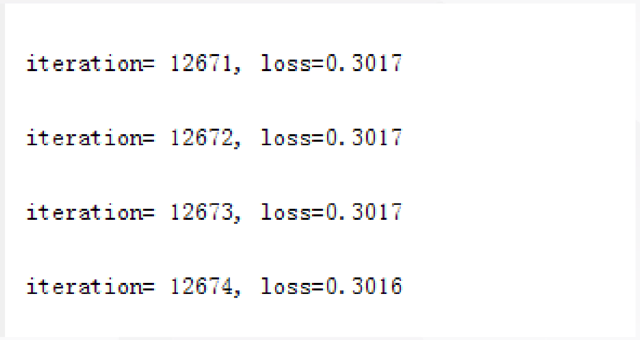
作业(选做)

■给定训练数据集 $X = \begin{pmatrix} 1 & 2 & 5 & 4 \\ 2 & 5 & 1 & 2 \end{pmatrix}, y = (19 \quad 26 \quad 19 \quad 20)^T$，令 step=0.001, w_0=[1 1]^T

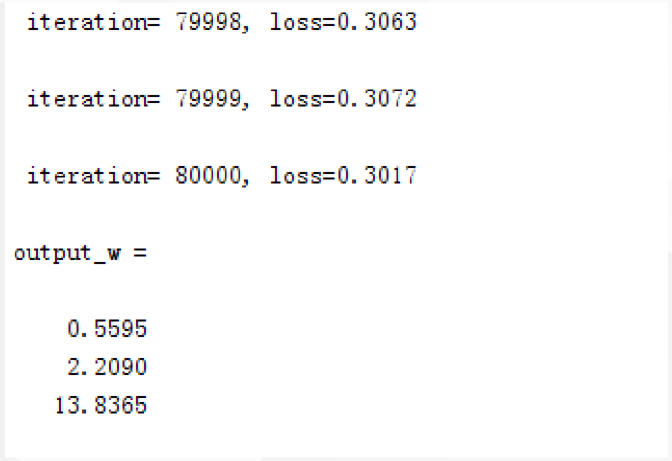
■编程实现 SGD和GD算法，求解 w



```
 iteration= 9998, loss=0.4715

 iteration= 9999, loss=0.4713

 iteration= 10000, loss=0.4712

output_w =

    0.6289
    2.3083
   13.2834
```



```
iteration= 12671, loss=0.3017

iteration= 12672, loss=0.3017

iteration= 12673, loss=0.3017

iteration= 12674, loss=0.3016
```



```
 iteration= 79998, loss=0.3063

 iteration= 79999, loss=0.3072

 iteration= 80000, loss=0.3017

output_w =

    0.5595
    2.2090
   13.8365
```

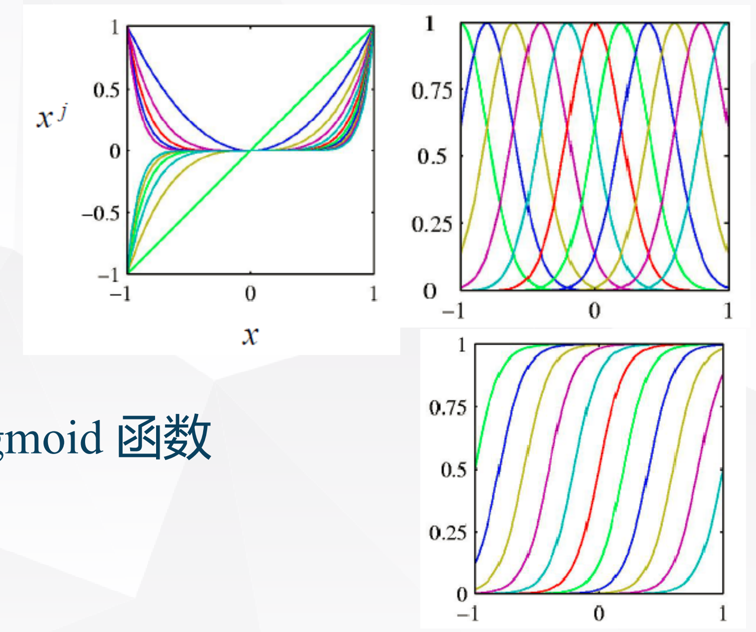
利用非线性基进行线性回归（广义线性回归）

■对非线性基进行线性组合: $f(\boldsymbol{w},\boldsymbol{x}) = w_0 + \sum_{j=1}^{K} w_j \phi_j(\boldsymbol{x})$

$$\Phi = (1, \phi_1, \cdots, \phi_K)$$

■非线性基函数

- $\phi(\boldsymbol{x}) = (1, x, x^2, \cdots, x^K)$ 多项式基函数
- $\phi_j(\boldsymbol{x}) = \exp\left(-\frac{(x-\mu_j)^2}{2s^2}\right)$ 高斯函数
- $\phi_j(\boldsymbol{x}) = \sigma\left(\frac{x-\mu_j}{s}\right), \sigma(a) = \frac{1}{1+\exp(-a)}$ Sigmoid 函数



广义线性回归的闭式解

- 最优化问题: $\min_{w} \; J(\boldsymbol{w}) = \sum_{i=1}^{N} (\boldsymbol{w}^T \phi(\boldsymbol{x}^i) - y^i)^2$
- 梯度: $\dfrac{\partial J(\boldsymbol{w})}{\partial w_j} = 2\sum_{i=1}^{N} \phi_j(\boldsymbol{x}^i)(\boldsymbol{w}^T \phi(\boldsymbol{x}^i) - y^i)$
- 闭式解: $\boldsymbol{w}^* = (\Phi^T \Phi)^{-1} \Phi^T \boldsymbol{y}$

其中 $\Phi = \begin{pmatrix} \phi_0(\boldsymbol{x}^1) & \cdots & \phi_K(\boldsymbol{x}^1) \\ \vdots & \vdots & \vdots \\ \phi_0(\boldsymbol{x}^N) & \cdots & \phi_K(\boldsymbol{x}^N) \end{pmatrix}, \boldsymbol{y} = (y^1, \cdots, y^N)^T$

- 也可用 SGD求解。

解释: MLE

- 在高斯噪声模型下，最小化平方误差与最大似然的解相同。

- 假设y是具有加性高斯噪声的确定函数f给出的标量，即 $y = f(\boldsymbol{x}, \boldsymbol{w}) + \varepsilon$，$\varepsilon$ 是均值为0，方差为 β^{-1} 的高斯噪声

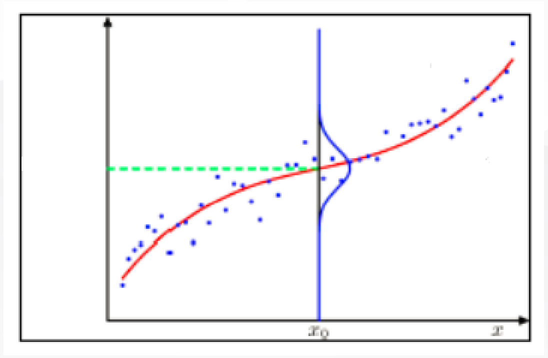
$$p(y \mid \boldsymbol{x}, \boldsymbol{w}, \beta) = N(y \mid f(\boldsymbol{x}, \boldsymbol{w}), \beta^{-1})$$

- 训练数据：

$$(\boldsymbol{x}^i, y^i), i = 1, 2, \cdots, N$$

- 似然函数：

$$\prod_{i=1}^{N} N(y^i \mid \boldsymbol{w}^T \boldsymbol{x}^i, \beta^{-1})$$



对数似然函数

■对数似然

$$\sum_{i=1}^{N} \ln N(y^i \mid \boldsymbol{w}^T \boldsymbol{x}^i, \beta^{-1}) = \frac{N}{2}\ln\beta - \frac{N}{2}\ln 2\pi - \frac{1}{2}\beta J(\boldsymbol{w})$$

其中 $J(\boldsymbol{w}) = \sum_{i=1}^{N} (\boldsymbol{w}^T \boldsymbol{x}^i - y^i)^2$ sum of square error

■结论：最大化似然相当于最小化平方误差之和。

最小二乘法实际上是在假设误差项满足高斯分布且独立同分布情况下，使似然性最大化。

正则化的 LMS 与 MAP

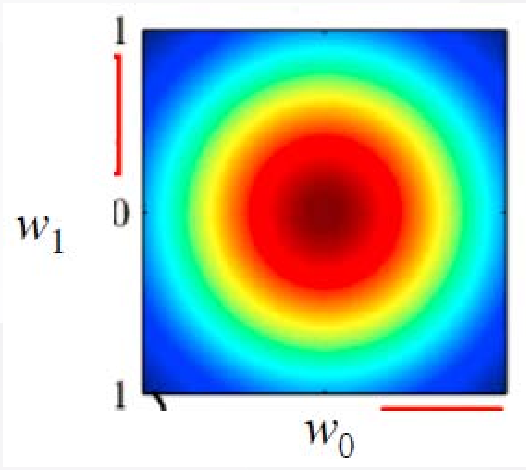
■正则化的LMS，最优化问题为： $\min_{w} \sum_{i=1}^{N} (\boldsymbol{w}^T \boldsymbol{x}^i - y^i)^2 + \lambda \boldsymbol{w}^T \boldsymbol{w}$

■闭式解为： $\boldsymbol{w}^* = \left(\Phi^T \Phi + \lambda \boldsymbol{I}\right)^{-1} \Phi^T \boldsymbol{y}$

■似然函数 为： $\prod_{i=1}^{N} N(y^i \mid \boldsymbol{w}^T \boldsymbol{x}^i, \beta^{-1})$

■参数的先验: $p(\boldsymbol{w}) = N(\boldsymbol{0}, \lambda^{-1} \boldsymbol{I})$ 多变量高斯

- w的各分量间的协相关系数为0，各分量方差为 λ^{-1}



参数的后验概率

■贝叶斯定理： $p(\boldsymbol{w} \mid \boldsymbol{y}) = p(\boldsymbol{y} \mid \boldsymbol{w})\, p(\boldsymbol{w}) / p(\boldsymbol{y})$

●似然函数： $p(\boldsymbol{y} \mid \boldsymbol{X}, \boldsymbol{w}, \beta) = \prod_{i=1}^{N} N(y^i \mid \boldsymbol{w}^T \boldsymbol{x}^i, \beta^{-1})$

●先验： $p(\boldsymbol{w}) = N(\boldsymbol{0}, \lambda^{-1}\boldsymbol{I})$

■后验概率依然是高斯分布，对后验取对数：

$$\boxed{\ln(p(\boldsymbol{w} \mid \boldsymbol{y})) = -\beta \sum_{i=1}^{N} (y^i - \boldsymbol{w}^T \boldsymbol{x}^i)^2 - \lambda \boldsymbol{w}^T \boldsymbol{w} + constant}$$

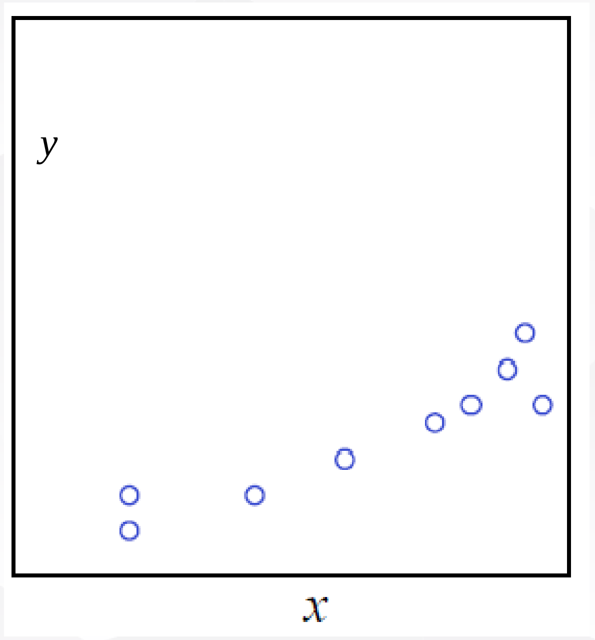
■最大化后验等同于最小化带有正则项的平方和误差。

$$\boxed{\min_{\boldsymbol{w}} \sum_{i=1}^{N} (\boldsymbol{w}^T \boldsymbol{x}^i - y^i)^2 + \alpha \boldsymbol{w}^T \boldsymbol{w},\ \ \alpha = \frac{\lambda}{\beta}}$$

例子: 直线拟合

- 输入自变量x，目标变量 y
- 目标：利用直线拟合数据
- 线性回归的目标：利用给定的样本，来学习函数： $f(\boldsymbol{w}, x) = w_0 + w_1 x$

$$\boldsymbol{w} = (w_0, w_1)^T$$



产生数据

■利用 w_0=-0.3, w_1=0.5的函数 $f(x,\mathbf{w}) = w_0 + w_1 x$ 产生数据

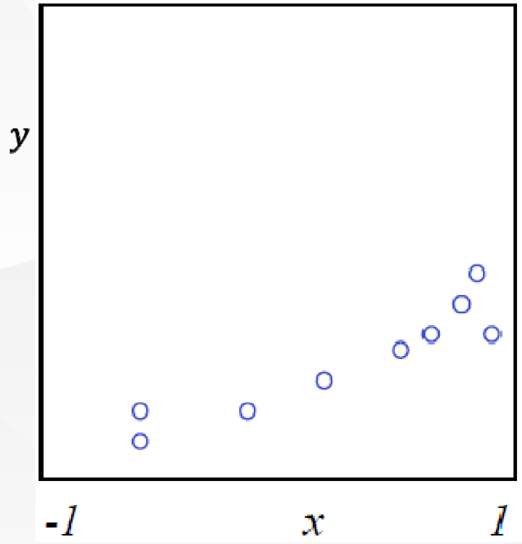
● 首先从均匀分布U(-1,1)中产生x^i, 然后利用函数$f(x^i,\mathbf{w})$计算相应的输出 。

●添加 $\sigma = 5$ 的高斯噪声来得到目标变量 y^i,

$$\beta = \left(\frac{1}{0.2}\right)^2 = 25$$

●对于 $\mathbf{w}$ 的先验，我们选择如下正态分布 $\alpha = 2$

$$p(\mathbf{w}|\alpha) = N(0, \alpha^{-1}\mathbf{I})$$

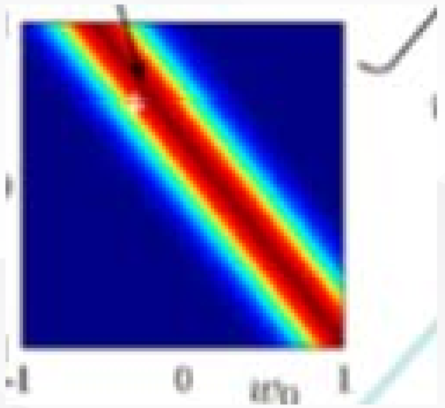


具体推导

$$p(y_n \mid x_n, \boldsymbol{w}) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(w_0 + w_1 x_n - y_n)^2}{2\sigma^2} \right)$$

$$w_0 + w_1 x_n = y_n$$

在权向量空间，等高线是什么形状？



$$p(y_1 \mid x_1, \boldsymbol{w}, \beta) = \frac{\sqrt{\beta}}{\sqrt{2\pi}} \exp\left(-\frac{\beta(w_0 + w_1 x_1 - y_1)^2}{2} \right)$$

$$p(\boldsymbol{w} \mid y_1) = \frac{p(y_1 \mid x_1, \boldsymbol{w})\, p(\boldsymbol{w})}{p(y_1)} \propto p(y_1 \mid x_1, \boldsymbol{w}, \beta)\, p(\boldsymbol{w})$$

$$= \frac{\sqrt{\beta}}{\sqrt{2\pi}} \exp\left(-\frac{\beta(w_0 + w_1 x_1 - y_1)^2}{2} \right) \frac{\sqrt{\alpha}}{\sqrt{2\pi}} \exp\left(-\frac{\alpha(w_0^2 + w_1^2)}{2} \right)$$

$$= 数 * \exp\left(-\frac{(\boldsymbol{w}\text{-}\hat{\boldsymbol{\mu}})^T \hat{\boldsymbol{\Sigma}}^{-1} (\boldsymbol{w}\text{-}\hat{\boldsymbol{\mu}})^T}{2} \right)$$

$$\hat{\boldsymbol{\Sigma}}^{-1} = \begin{pmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{12} & \Sigma_{22} \end{pmatrix}, \hat{\boldsymbol{\mu}} = (\mu_0, \mu_1)^T$$

$$\Sigma_{11} = \beta + \alpha，\ \Sigma_{22} = \beta(x_1)^2 + \alpha, \Sigma_{12} = \beta x_1$$

$$\mu_0 = \frac{\Sigma_{12}\beta x_1 y_1 - \Sigma_{22}\beta y_1}{(\Sigma_{12})^2 - \Sigma_{11}\Sigma_{22}}, \mu_1 = \frac{\Sigma_{12}\beta x_1 - \Sigma_{11}\beta y_1 x_1}{(\Sigma_{12})^2 - \Sigma_{11}\Sigma_{22}}$$

Sequential Bayesian Learning

$$p(y_n \mid x_n, \boldsymbol{w}) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(w_0 + w_1 x_n - y_n)^2}{2\sigma^2} \right)$$

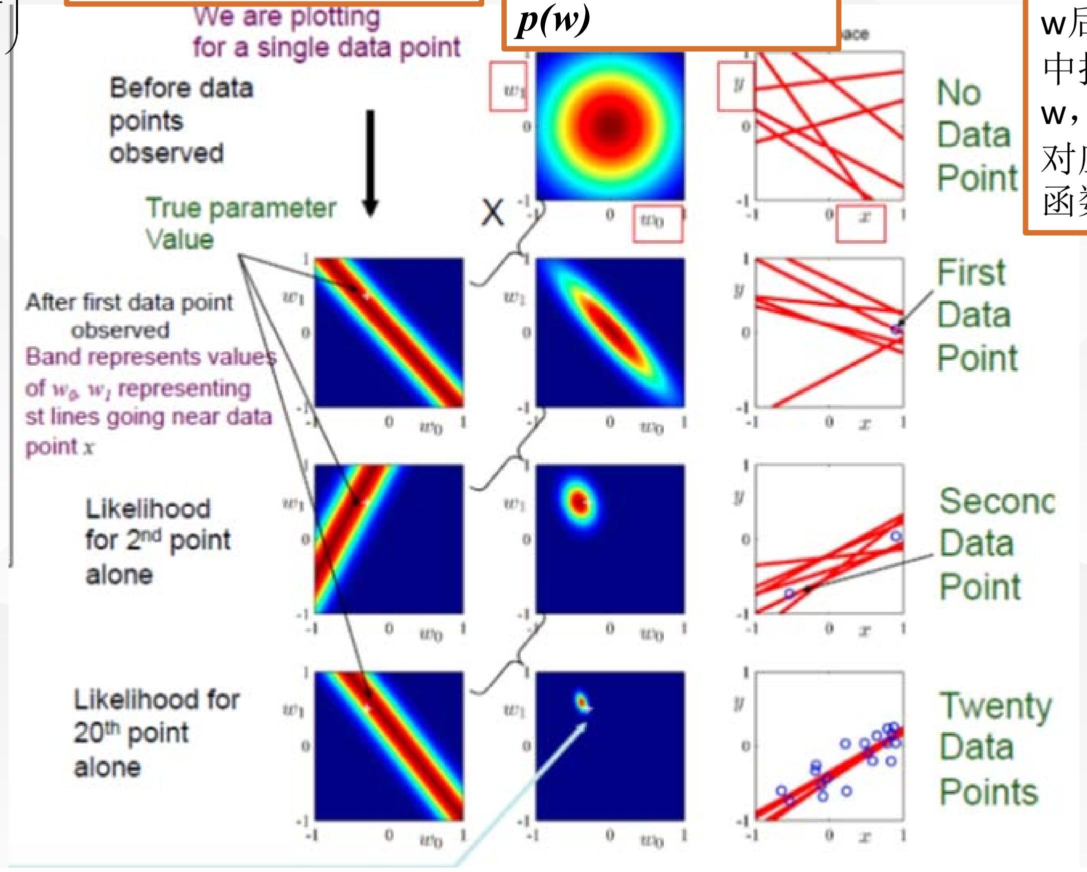
- 由于只有两个参数
  - •我们可以在参数空间画出参数的参验及后验分布
- 对后验概率进行更新。

当有无限的数据点时，参数的后验分布是以其真实值为中心的Delta分布。

似然函数　*p(y|x,w)*（关于 ***w***的函数 ）

给定 ***p(y|x,w)***， ***w*** 的先验/后验 ***p(w)***

六个样本(从w后验概率中抽取六个w，得到的对应的回归函数f(*x*,***w***)



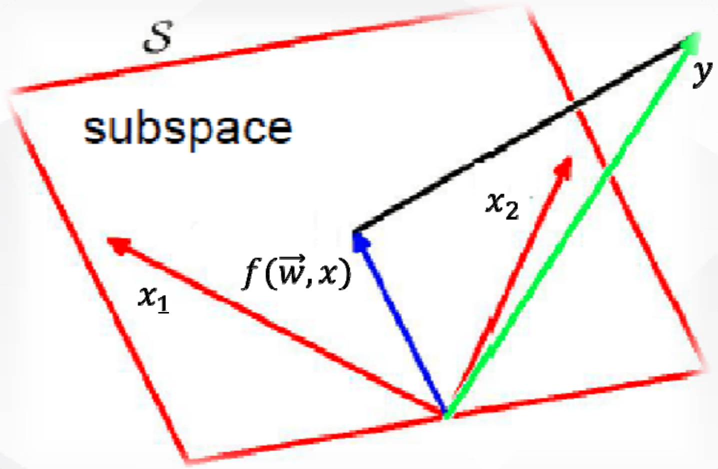
MLE 与MAP

- MLE: $\hat{\theta}_{MLE} = \arg\max_{\theta} P(D \mid \theta)$
- MAP: $\hat{\theta}_{MAP} = \arg\max_{\theta} P(\theta \mid D) = \arg\max_{\theta} P(D \mid \theta) P(\theta)$
- MLE 可以看作是使用**哪种分布作为先验**的 MAP ?
- MLE是频率学派的想法，而MAP是贝叶斯学派的。
- 二者之间的比较与 ERM 与 SRM 的比较类似。为什么?
- 更多的数据会使MLE拟合得更好，但是容易 过拟合。

LMS-几何解释

- $\boldsymbol{y}=\left(y^1,y^2,\cdots,y^N\right)$ 在N 维空间
- 如果 $\boldsymbol{w}$ 的维度(m)小于N, 那么样本 x_js 在m维的子空间 S
- $\boldsymbol{X}=\left(\boldsymbol{x}^1,\boldsymbol{x}^2,\cdots,\boldsymbol{x}^N\right)$, 那么 $f(\boldsymbol{w},\boldsymbol{X})$ 是一个N维向量
- 最优的$\boldsymbol{w}$ 使得 $f(\boldsymbol{w},\boldsymbol{X})$ 为子空间 S 中与y最近的向量。

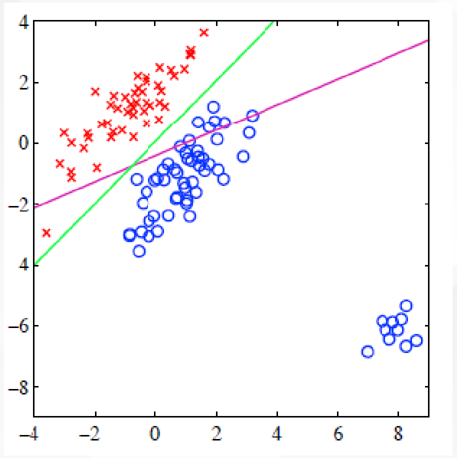
相当于$\boldsymbol{y}$在子空间S上的投影。



分类方法

分类问题

- 输入: N i.i.d 训练样本 $(\boldsymbol{x}^i, y^i) \in X \times C,\ i = 1, \cdots, N$
- 目标函数： $f \in \mathcal{F}$
- 损失函数： $L(f; x, y) = I_{\{f(\boldsymbol{x}) \neq y\}}$
- 期望风险（损失）： $\int I_{\{f(\boldsymbol{x}) \neq y\}} dP(\boldsymbol{x}, y) = P(f(\boldsymbol{x}) \neq y)$



为什么不直接应用回归的损失，这样就可以用回归问题的求解方法。

异常值问题！

判别函数法

- 二类问题: y=1 or -1
- 求解 $\boldsymbol{w}$, $f(\boldsymbol{x},\boldsymbol{w}) = y$ ，例如:$f(\boldsymbol{x},\boldsymbol{w}) = \text{sgn}(\boldsymbol{w}^T\boldsymbol{x}) = \begin{cases} 1, \boldsymbol{w}^T\boldsymbol{x} > 0 \\ -1, \boldsymbol{w}^T\boldsymbol{x} < 0 \end{cases}$
- 使用 y=1 或 -1, 所有数据需要满足:

$$\boldsymbol{w}^T\boldsymbol{x}y > 0$$

- 对于每一个错分的样本，**一些方法**试图最小化如下的式子：

$$\hat{E}_p(\boldsymbol{w}) = -\sum_{i \in \boxed{T_M}} \boldsymbol{w}^T\boldsymbol{x}^i y^i$$

错分样本的集合

判别式模型: Logistic 回归

■ 估计后验概率 $p(y|\boldsymbol{x})$

$$P(y=1\,|\,\boldsymbol{x}) = f(\boldsymbol{x},\boldsymbol{w}) = g(\boldsymbol{w}^T\boldsymbol{x}) = \frac{1}{1+\exp(-\boldsymbol{w}^T\boldsymbol{x})}$$

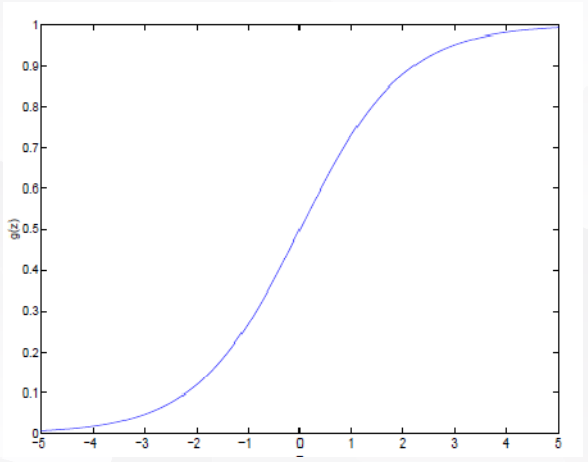
Logistic function

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

Sigmoid function

■ $g(z)$的性质

- $z \to \infty$ 时$g(z) \to 1$
- $z \to -\infty$ *时*, $g(z) \to 0$
- $g(z)$的0-1之间
- $g'(z)=g(z)(1-g(z))$



求解：最大似然估计（Maximum Likelihood Estimator)

- 概率分布：

$$\boxed{P(y \mid \boldsymbol{x}, \boldsymbol{w}) = (f(\boldsymbol{x}, \boldsymbol{w}))^{y}(1 - f(\boldsymbol{x}, \boldsymbol{w}))^{1-y}}$$

Bernoulli

- 似然:

$$\boxed{L(\boldsymbol{w}) = \prod_{i=1}^{N} P(y^i \mid \boldsymbol{x}^i, \boldsymbol{w}) = \prod_{i=1}^{N} \left(f(\boldsymbol{x}^i, \boldsymbol{w})\right)^{y^i} \left(1 - f(\boldsymbol{x}^i, \boldsymbol{w})\right)^{1-y^i}}$$

- 最大化log 似然:

$$\boxed{l(\boldsymbol{w}) = \log L(\boldsymbol{w}) = \sum_{i=1}^{N} \left(y^i \log f(\boldsymbol{x}^i, \boldsymbol{w}) + (1 - y^i) \log\left(1 - f(\boldsymbol{x}^i, \boldsymbol{w})\right)\right)}$$

- 梯度:

$$\frac{\partial l(\boldsymbol{w})}{\partial w_j} = (y^i - f(\boldsymbol{x}^i, \boldsymbol{w}))x_j^i, \forall (\boldsymbol{x}^i, y^i)$$

- SGD:

$$\boxed{w_j = w_j + \alpha (y^i - f(\boldsymbol{x}^i, \boldsymbol{w}))x_j^i}$$

补充材料

- $g'(z)=g(z)(1-g(z))$

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

$$\boxed{L(\boldsymbol{w})=\prod_{i=1}^{N}P(y^i\mid \boldsymbol{x}^i,\boldsymbol{w})=\prod_{i=1}^{N}\left(f(\boldsymbol{x}^i,\boldsymbol{w})\right)^{y^i}\left(1-f(\boldsymbol{x}^i,\boldsymbol{w})\right)^{1-y^i}}$$

$$P(y=1\mid \boldsymbol{x})=f(\boldsymbol{x},\boldsymbol{w})=g(\boldsymbol{w}^T\boldsymbol{x})=\frac{1}{1+\exp(-\boldsymbol{w}^T\boldsymbol{x})}$$

$$\boxed{\begin{aligned}
&l(\boldsymbol{w})=\log L(\boldsymbol{w})=\sum_{i=1}^{N}\left(y^i\ln f(\boldsymbol{x}^i,\boldsymbol{w})+(1-y^i)\ln\left(1-f(\boldsymbol{x}^i,\boldsymbol{w})\right)\right)\\
&\frac{\partial l(\boldsymbol{w})}{\partial w_j}=\sum_{i=1}^{N}\left(y^i\frac{1}{f(\boldsymbol{x}^i,\boldsymbol{w})}\frac{\partial f(\boldsymbol{x}^i,\boldsymbol{w})}{\partial w_j}+(1-y^i)\frac{1}{\left(1-f(\boldsymbol{x}^i,\boldsymbol{w})\right)}\frac{-\partial f(\boldsymbol{x}^i,\boldsymbol{w})}{\partial w_j}\right)\\
&\frac{\partial f(\boldsymbol{x}^i,\boldsymbol{w})}{\partial w_j}=f(\boldsymbol{x}^i,\boldsymbol{w})(1-f(\boldsymbol{x}^i,\boldsymbol{w}))x_j^i\\
&\frac{\partial l(\boldsymbol{w})}{\partial w_j}=\sum_{i=1}^{N}\left(y^i(1-f(\boldsymbol{x}^i,\boldsymbol{w}))x_j^i+(1-y^i)\left(-f(\boldsymbol{x}^i,\boldsymbol{w})\right)x_j^i\right)=\sum_{i=1}^{N}\left(y^i-f(\boldsymbol{x}^i,\boldsymbol{w})\right)x_j^i
\end{aligned}}$$

多类 Logistic回归

- Softmax 函数取代logistic sigmoid:

$$P(C_k \mid \boldsymbol{w}, \boldsymbol{x}) = \frac{\exp(\boldsymbol{w}_k^T \boldsymbol{x})}{\sum_{j=1}^{K} \exp(\boldsymbol{w}_j^T \boldsymbol{x})}$$

  这里 $\boldsymbol{w}_1, \boldsymbol{w}_2, \cdots, \boldsymbol{w}_K$ 是待学习的参数。

- y可以看作是取K值之一的离散变量
  - 将y表示为K维的向量
  - 如果 y =3, 那么$\boldsymbol{y}$=(0,0,1,0,…,0)（第三个元素为1，其它元素为0）
  - K维向量满足：

$$\sum_{i=1}^{K} P(y_i \mid \boldsymbol{w}, \boldsymbol{x}) = 1$$

One hot 表示
独热表示

- 概率分布:

$$\mu_i = P(y_i = 1 \mid \boldsymbol{w}, \boldsymbol{x})$$

$$P(\boldsymbol{y} \mid \boldsymbol{\mu}) = \prod_{i=1}^{K} \mu_i^{y_i}$$

Generalized Bernoulli
广义伯努里分布

似然函数

$$P(\boldsymbol{y}^1,\boldsymbol{y}^2,\cdots,\boldsymbol{y}^N \mid \boldsymbol{w}_1,\boldsymbol{w}_2,\cdots,\boldsymbol{w}_K,\boldsymbol{x}^1,\cdots,\boldsymbol{x}^N)$$
$$=\prod_{i=1}^{N}\prod_{k=1}^{K}P(C_k \mid \boldsymbol{w}_k,\boldsymbol{x}^i)=\prod_{i=1}^{N}\prod_{k=1}^{K}\mu_{ik}^{y_k^i}$$

■其中

$$\mu_{ik}=\frac{\exp(\boldsymbol{w}_k^T\boldsymbol{x}^i)}{\sum_{j=1}^{K}\exp(\boldsymbol{w}_j^T\boldsymbol{x}^i)}$$

优化多类LR

■最优化问题

$$\min E = -\ln P(\boldsymbol{y}^1, \cdots, \boldsymbol{y}^N \mid \boldsymbol{w}_1, \cdots, \boldsymbol{w}_k, \boldsymbol{x}^1, \cdots, \boldsymbol{x}^N)$$

$$\Rightarrow \boxed{\min - \sum_{i=1}^{N} \sum_{k=1}^{K} y_k^i \ln \mu_{ik}}$$

Cross Entropy Loss Function
交叉熵损失函数

■梯度:

$$\boxed{\nabla_{\boldsymbol{w}_j} E = \sum_{i=1}^{N} (\mu_{ij} - y_j^i) \boldsymbol{x}^i}$$

■N : 样本数量

补充材料

$$E = -\sum_{i=1}^{N}\sum_{k=1}^{K} y_k^i \ln \mu_{ik}$$

$$\mu_{ik} = \frac{\exp(\boldsymbol{w}_k^T \boldsymbol{x}^i)}{\sum_{j=1}^{K}\exp(\boldsymbol{w}_j^T \boldsymbol{x}^i)}$$

$$\ln \mu_{ik} = \ln \exp(\boldsymbol{w}_k^T \boldsymbol{x}^i) - \ln \sum_{j=1}^{K}\exp(\boldsymbol{w}_j^T \boldsymbol{x}^i)$$

$$if\ \ k = j, \frac{\partial \ln \mu_{ij}}{\partial \boldsymbol{w}_j} = \boldsymbol{x}^i - \frac{\boldsymbol{x}^i \exp(\boldsymbol{w}_j^T \boldsymbol{x}^i)}{\sum_{j=1}^{K}\exp(\boldsymbol{w}_j^T \boldsymbol{x}^i)} = \boldsymbol{x}^i - \boldsymbol{x}^i \mu_{ij}$$

$$if\ \ k \neq j, \frac{\partial \ln \mu_{ik}}{\partial \boldsymbol{w}_j} = \frac{\boldsymbol{x}^i \exp(\boldsymbol{w}_j^T \boldsymbol{x}^i)}{\sum_{j=1}^{K}\exp(\boldsymbol{w}_j^T \boldsymbol{x}^i)} = -\boldsymbol{x}^i \mu_{ij}$$

$$\nabla_{\boldsymbol{w}_j} E = -\sum_{i=1}^{N}\left(\sum_{k=1}^{K}(-y_k^i \boldsymbol{x}^i \mu_{ij}) + y_j^i \boldsymbol{x}^i\right) = \sum_{i=1}^{N}\left(\sum_{k=1}^{K}(y_k^i \boldsymbol{x}^i \mu_{ij}) - y_j^i \boldsymbol{x}^i\right)$$

$$= \sum_{i=1}^{N}\left(\boldsymbol{x}^i \mu_{ij} \sum_{k=1}^{K}(y_k^i) - y_j^i \boldsymbol{x}^i\right)$$

由于$\sum_{k=1}^{K}(y_k^i) = 1$

所以上式$= \sum_{i=1}^{N}\left(\mu_{ij} - y_j^i\right)\boldsymbol{x}^i$

多类 Logistic 回归(另一种形式)

- 只学习K-1的向量 $\boldsymbol{w}_1, \boldsymbol{w}_2, \cdots, \boldsymbol{w}_{K-1}$

$$P(C_k \mid \boldsymbol{w}, \boldsymbol{x}) = \frac{\exp(\boldsymbol{w}_k^T \boldsymbol{x})}{1 + \sum_{j=1}^{K-1} \exp(\boldsymbol{w}_j^T \boldsymbol{x})} \qquad P(C_K \mid \boldsymbol{w}, \boldsymbol{x}) = \frac{1}{1 + \sum_{j=1}^{K-1} \exp(\boldsymbol{w}_j^T \boldsymbol{x})}$$

- 交叉熵损失:

$$\boxed{\min - \sum_{i=1}^{N} \sum_{k=1}^{K} y_k^i \ln \mu_{ik}}$$

$$\boxed{\mu_{ik} = \frac{\exp(\boldsymbol{w}_k^T \boldsymbol{x}^i)}{1 + \sum_{j=1}^{K-1} \exp(\boldsymbol{w}_j^T \boldsymbol{x}^i)}, \mu_{iK} = \frac{1}{1 + \sum_{j=1}^{K-1} \exp(\boldsymbol{w}_j^T \boldsymbol{x}^i)}}$$

- 梯度:

$$\boxed{\nabla_{\boldsymbol{w}_j} E = \sum_{i=1}^{N} (\mu_{ij} - y_j^i) \boldsymbol{x}^i}$$

生成式模型: 高斯判别分析Gaussian Discriminative Analysis

- 伯努利分布：单个二进制变量 $x \in \{0,1\}$ 的分布由单个连续参数 $\beta \in [0,1]$ 控制。

$$P(x \mid \beta) = \beta^{x}(1-\beta)^{1-x}$$

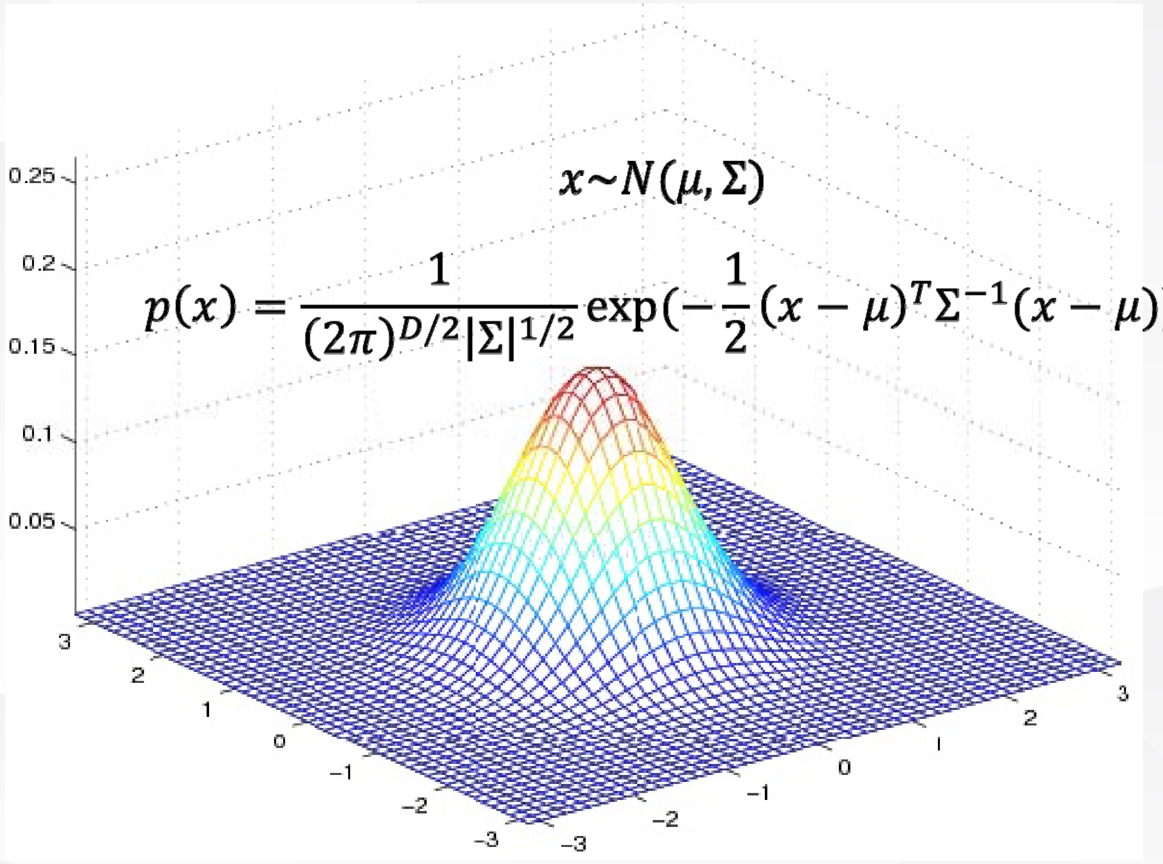
- 二项式分布: 给出N个服从伯努利分布的样本中观察到m次x=1的的概率:

$$P(m \mid N, \beta) = \binom{N}{m} \beta^{m}(1-\beta)^{N-m}$$

- 多项式分布（Multinomial Distribution）是二项式分布的推广。变量可以取*K*个状态，第k个状态被观测到了 m_k 次的概率为：

$$P(m_1, \cdots, m_K \mid N, \beta) = \binom{N}{m_1, \cdots, m_K} \prod_{k=1}^{K} \beta_k^{m_k}$$

多变量正态分布



$$x \sim N(\mu, \Sigma)$$

$$p(x) = \frac{1}{(2\pi)^{D/2}|\Sigma|^{1/2}} \exp(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu))$$

高斯判别分析（GDA）

■GDA 使用多变量正态分布对 $p(x|y)$ 进行建模

$$\boxed{y \sim Bernoulli(\beta);\ x \mid y=0 \sim N(\boldsymbol{\mu}_0, \boldsymbol{\Sigma}); x \mid y=1 \sim N(\boldsymbol{\mu}_1, \boldsymbol{\Sigma});}$$

■Log 似然:

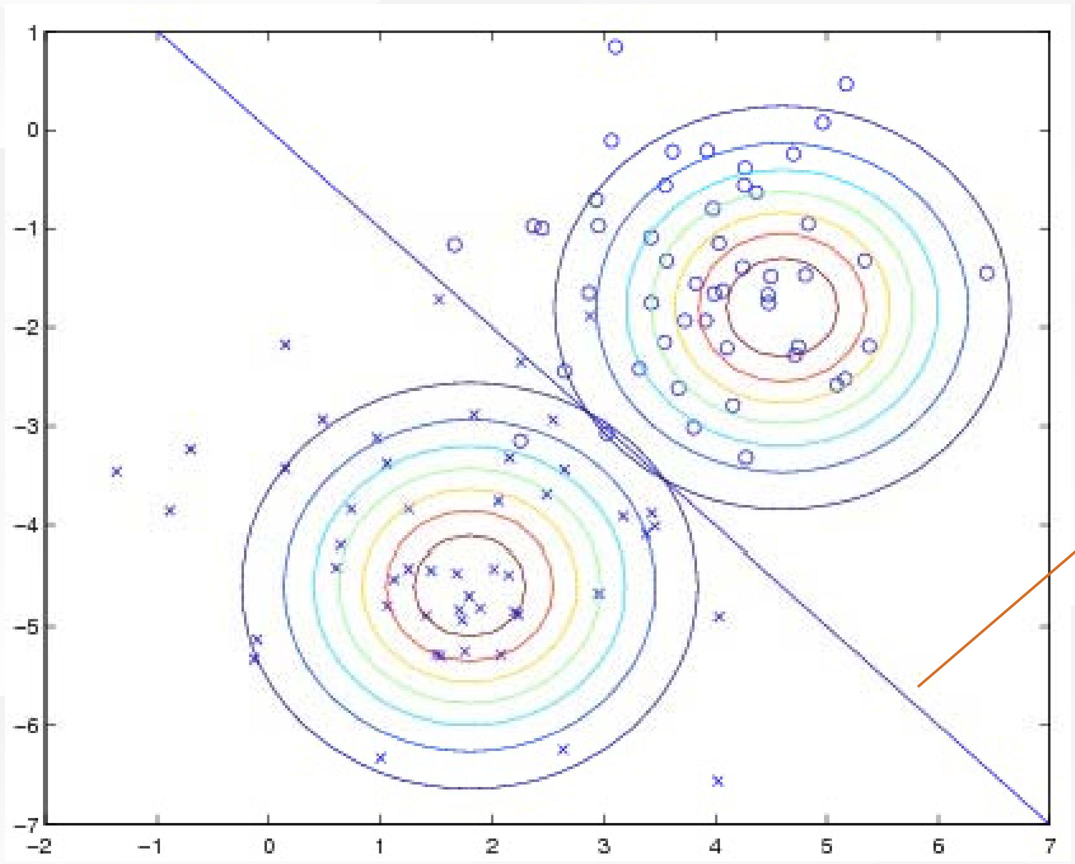
$$L(\beta, \boldsymbol{\mu}_0, \boldsymbol{\mu}_1, \boldsymbol{\Sigma}) = \log \prod_{i=1}^{N} p(\boldsymbol{x}^i, \boldsymbol{y}^i; \beta, \boldsymbol{\mu}_0, \boldsymbol{\mu}_1, \boldsymbol{\Sigma})$$

$$= \log \prod_{i=1}^{N} p(\boldsymbol{y}^i; \beta) p(\boldsymbol{x}^i \mid \boldsymbol{y}^i; \beta, \boldsymbol{\mu}_0, \boldsymbol{\mu}_1, \boldsymbol{\Sigma})$$

■MLE:

$$\boxed{\beta = \frac{1}{N}\sum_{i=1}^{N} I_{\{y^i=1\}}; \mu_k = \frac{\sum_{i=1}^{N} I_{\{y^i=k\}} x_i}{\sum_{i=1}^{N} I_{\{y^i=k\}}}, k=\{0,1\}; \Sigma = \frac{1}{N}\sum_{i=1}^{N} (\boldsymbol{x}^i - \boldsymbol{\mu}_{y^i})(\boldsymbol{x}^i - \boldsymbol{\mu}_{y^i})^T}$$

GDA示例



$P(y = 1|x) = 0.5$

GDA 与 LR

- ■有很强的模型假设。当假设是正确时，处理数据的效率更高。

- ■假设很弱，因此对偏离假设的情况更具鲁棒性.

- ■实际中, 谁更常用?

- ■在GDA中，特征向量 x 中的元素是连续的实数

  如果特征向量 x 中的元素是离散的，怎么处理？

生成式模型: 朴素贝叶斯（Naïve Bayes)

■假设给定y时，特征分量 $\boldsymbol{x}_j^{'}s$ 相互独立（**Naïve Bayes**).

$$p(x_1,\cdots,x_D \mid y) = \prod_{i=1}^{D} p(x_i \mid y)$$

■给定训练数据 $(\boldsymbol{x}^i, y^i),\ i = 1,\cdots,N$ ，对数似然为:

$$L = \sum_{i=1}^{N}\sum_{j=1}^{D}\{\log p(x_j^i \mid y^i) + \log p(y^i)\}$$

参数估计与 预测

■MLE

$$p(x_j=1\mid y=1)=\frac{\sum_{i=1}^{N} I_{\{x_j^i=1\wedge y^i=1\}}}{\sum_{i=1}^{N} I_{\{y^i=1\}}},\ p(x_j=1\mid y=0)=\frac{\sum_{i=1}^{N} I_{\{x_j^i=1\wedge y^i=0\}}}{\sum_{i=1}^{N} I_{\{y^i=0\}}}$$

$$p(y=1)=\frac{\sum_{i=1}^{N} I_{\{y^i=1\}}}{N}$$

Can be generalized to multi-classification problem!

■预测

$$p(y=1\mid \boldsymbol{x})=\frac{p(\boldsymbol{x}\mid y=1)\,p(y=1)}{p(\boldsymbol{x})}$$

$$=\frac{\prod_{j=1}^{D} p(x_j|y=1)p(y=1)}{\prod_{j=1}^{D} p(x_j|y=1)p(y=1)+\prod_{j=1}^{D} p(x_j|y=0)p(y=0)}$$

平滑

- 给定训练数据 $\{\boldsymbol{x}^1,\cdots,\boldsymbol{x}^N\}$， 利用最大似然估计可以估计变量$x$取每个值的概率：

$$p(x=j)=\frac{\sum_{i=1}^{N} I_{\{\boldsymbol{x}^i=j\}}}{N},\ j=1,\ldots,K$$

- 如果训练集中某个类别的数据没有涵盖x的第 m个取值的话，就无法估计相应的条件概率，从而导致模型可能会在测试集上的分类产生误差。

- Laplace 平滑，在各个估计中加入平滑项:

$$p(x=j)=\frac{\sum_{i=1}^{N} I_{\{\boldsymbol{x}^i=j\}}+1}{N+K},\ j=1,\ldots,K$$

NB 与LR

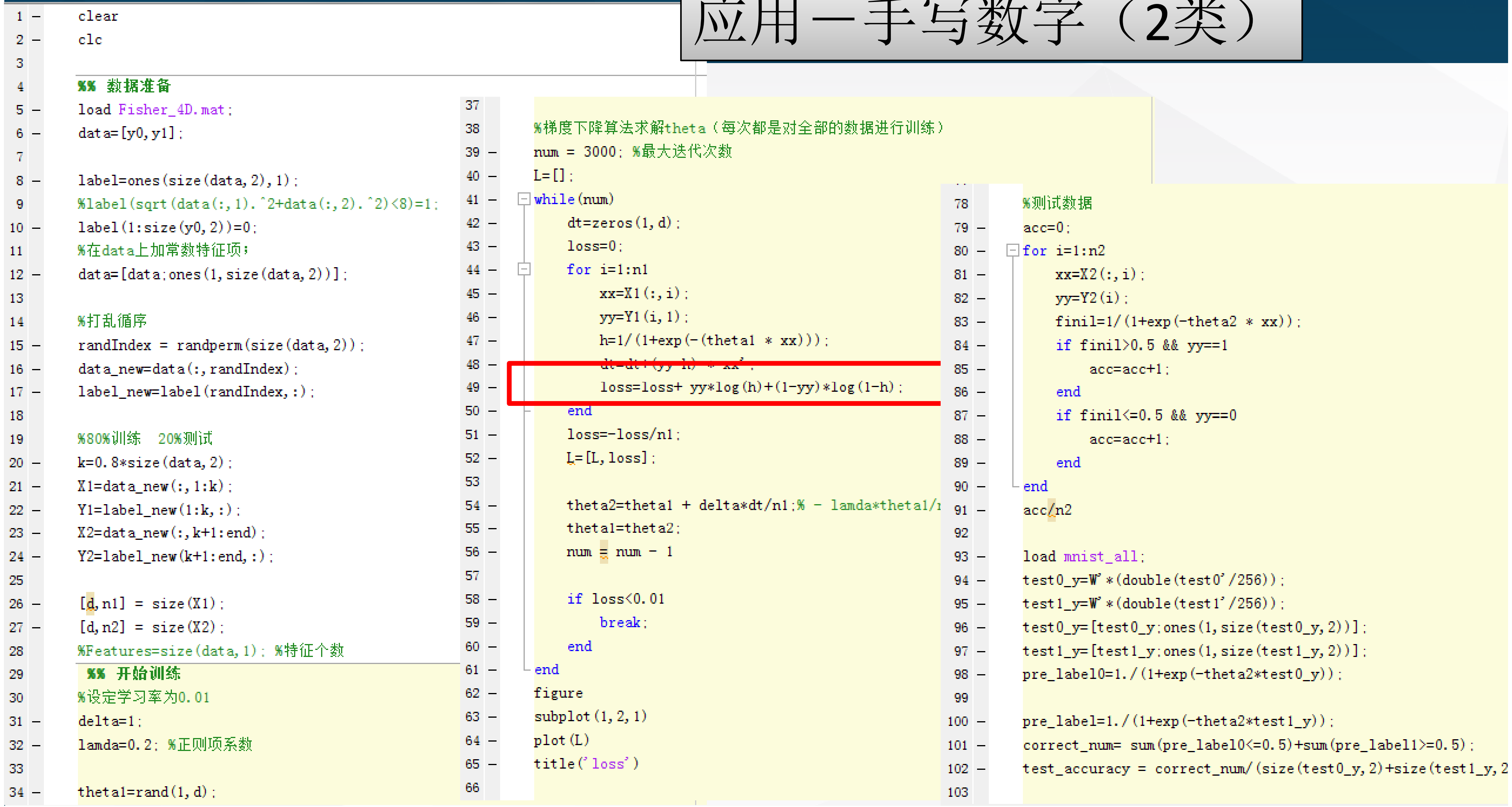
- 渐近比较（training data→ infinity)
  - 当模型假设是正确时
    - NB 和 LR 产生相似的分类器
  - 当模型假设不正确时
    - 因为LR不假设有条件独立性，因此它的偏差较小。
    - 因此，LR的预期表现优于NB。
- 非渐近比较 (Ng. and Jordan 2002)
  - 参数估计的收敛性

    需要多少训练数据才能获得一个好的估计?

    - NB ： O($\log D$)
    - LR： O(D)

    NB收敛速度更快，以拟合其渐近估计量

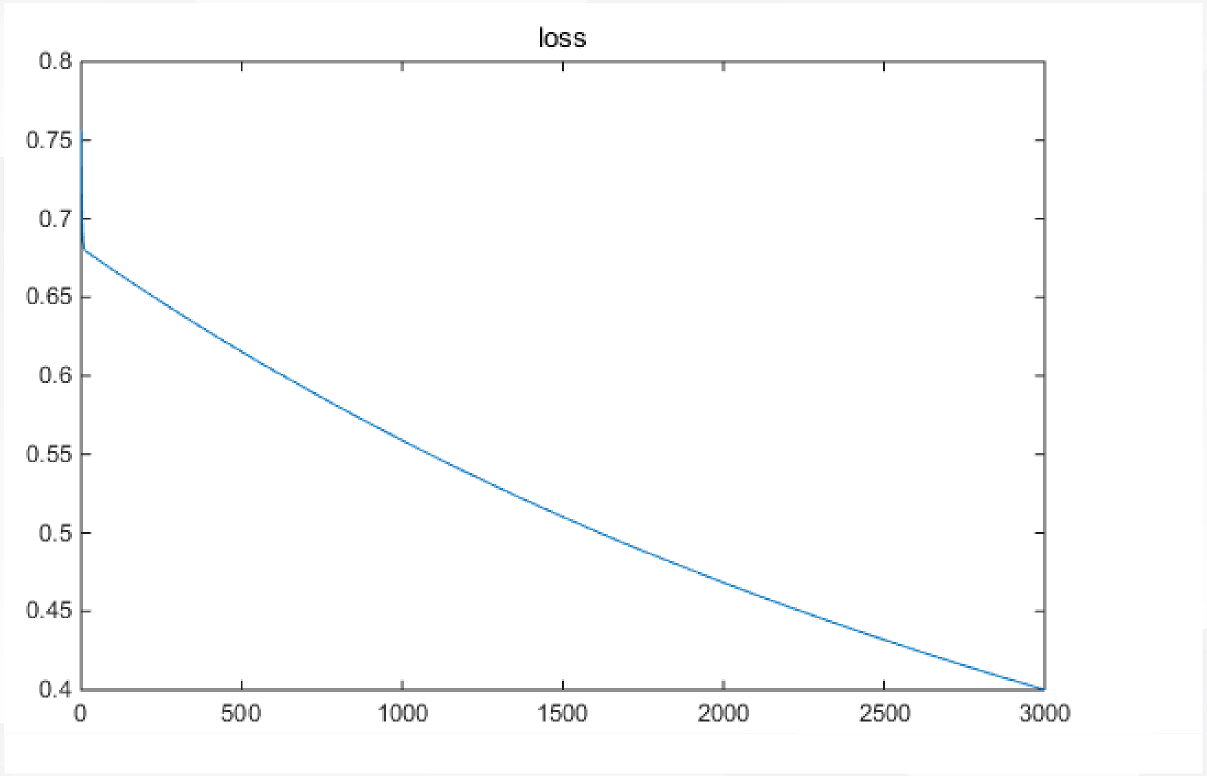
应用—手写数字（2类）



```
clear
clc

%% 数据准备
load Fisher_4D.mat;
data=[y0,y1];

label=ones(size(data,2),1);
%label(sqrt(data(:,1).^2+data(:,2).^2)<8)=1;
label(1:size(y0,2))=0;
%在data上加常数特征项;
data=[data;ones(1,size(data,2))];

%打乱循序
randIndex = randperm(size(data,2));
data_new=data(:,randIndex);
label_new=label(randIndex,:);

%80%训练   20%测试
k=0.8*size(data,2);
X1=data_new(:,1:k);
Y1=label_new(1:k,:);
X2=data_new(:,k+1:end);
Y2=label_new(k+1:end,:);

[d,n1] = size(X1);
[d,n2] = size(X2);
%Features=size(data,1); %特征个数
 %% 开始训练
%设定学习率为0.01
delta=1;
lamda=0.2; %正则项系数

theta1=rand(1,d);
```

```

%梯度下降算法求解theta（每次都是对全部的数据进行训练）
num = 3000; %最大迭代次数
L=[];
while(num)
    dt=zeros(1,d);
    loss=0;
    for i=1:n1
        xx=X1(:,i);
        yy=Y1(i,1);
        h=1/(1+exp(-(theta1 * xx)));
        dt=dt+(yy-h) * xx';
        loss=loss+ yy*log(h)+(1-yy)*log(1-h);
    end
    loss=-loss/n1;
    L=[L,loss];

    theta2=theta1 + delta*dt/n1;% - lamda*theta1/r
    theta1=theta2;
    num = num - 1

    if loss<0.01
        break;
    end
end
figure
subplot(1,2,1)
plot(L)
title('loss')

```

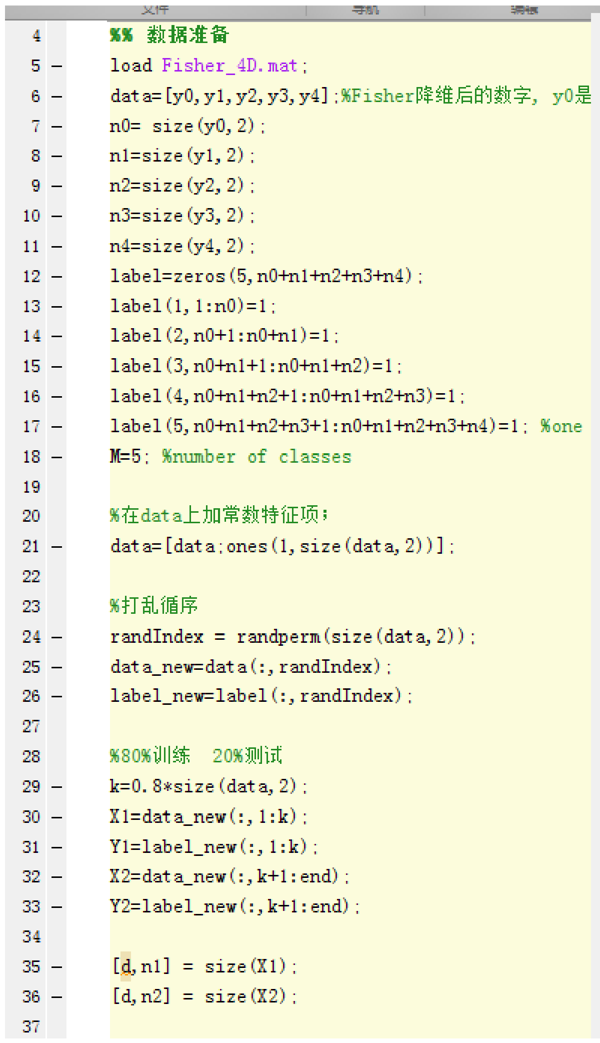
```
%测试数据
acc=0;
for i=1:n2
    xx=X2(:,i);
    yy=Y2(i);
    finil=1/(1+exp(-theta2 * xx));
    if finil>0.5 && yy==1
        acc=acc+1;
    end
    if finil<=0.5 && yy==0
        acc=acc+1;
    end
end
acc/n2

load mnist_all;
test0_y=W'*(double(test0'/256));
test1_y=W'*(double(test1'/256));
test0_y=[test0_y;ones(1,size(test0_y,2))];
test1_y=[test1_y;ones(1,size(test1_y,2))];
pre_label0=1./(1+exp(-theta2*test0_y));

pre_label=1./(1+exp(-theta2*test1_y));
correct_num= sum(pre_label0<=0.5)+sum(pre_label1>=0.5);
test_accuracy = correct_num/(size(test0_y,2)+size(test1_y,2

```



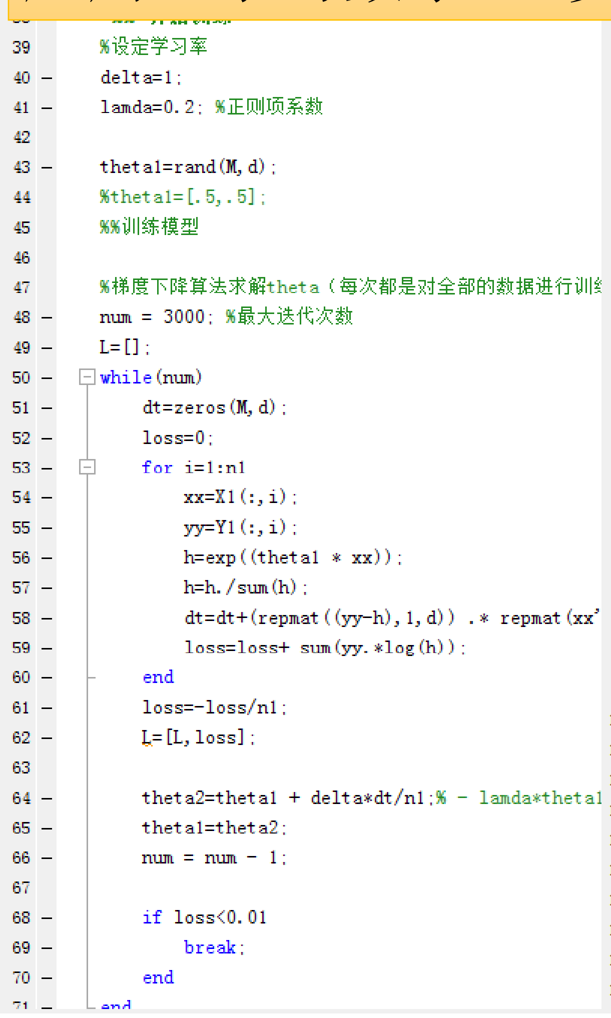
validation accuracy = 0.9957

test_accuracy = 0.9967

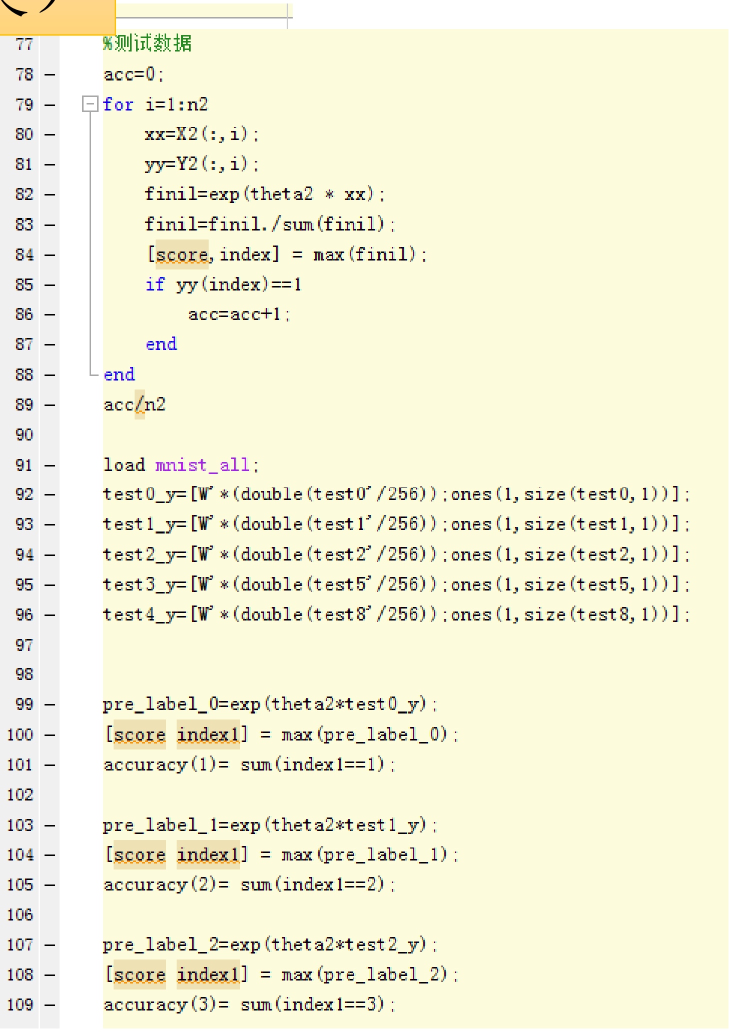
应用一手写数字（5类）



```matlab
%% 数据准备
load Fisher_4D.mat;
data=[y0,y1,y2,y3,y4];%Fisher降维后的数字, y0是
n0= size(y0,2);
n1=size(y1,2);
n2=size(y2,2);
n3=size(y3,2);
n4=size(y4,2);
label=zeros(5,n0+n1+n2+n3+n4);
label(1,1:n0)=1;
label(2,n0+1:n0+n1)=1;
label(3,n0+n1+1:n0+n1+n2)=1;
label(4,n0+n1+n2+1:n0+n1+n2+n3)=1;
label(5,n0+n1+n2+n3+1:n0+n1+n2+n3+n4)=1; %one
M=5; %number of classes

%在data上加常数特征项;
data=[data;ones(1,size(data,2))];

%打乱循序
randIndex = randperm(size(data,2));
data_new=data(:,randIndex);
label_new=label(:,randIndex);

%80%训练  20%测试
k=0.8*size(data,2);
X1=data_new(:,1:k);
Y1=label_new(:,1:k);
X2=data_new(:,k+1:end);
Y2=label_new(:,k+1:end);

[d,n1] = size(X1);
[d,n2] = size(X2);
```



```matlab
%设定学习率
delta=1;
lamda=0.2; %正则项系数

theta1=rand(M,d);
%theta1=[.5,.5];
%%训练模型

%梯度下降算法求解theta（每次都是对全部的数据进行训纟
num = 3000; %最大迭代次数
L=[];
while(num)
    dt=zeros(M,d);
    loss=0;
    for i=1:n1
        xx=X1(:,i);
        yy=Y1(:,i);
        h=exp((theta1 * xx));
        h=h./sum(h);
        dt=dt+(repmat((yy-h),1,d)) .* repmat(xx'
        loss=loss+ sum(yy.*log(h));
    end
    loss=-loss/n1;
    L=[L,loss];

    theta2=theta1 + delta*dt/n1;% - lamda*theta1
    theta1=theta2;
    num = num - 1;

    if loss<0.01
        break;
    end
end
```



```matlab
%测试数据
acc=0;
for i=1:n2
    xx=X2(:,i);
    yy=Y2(:,i);
    finil=exp(theta2 * xx);
    finil=finil./sum(finil);
    [score,index] = max(finil);
    if yy(index)==1
        acc=acc+1;
    end
end
acc/n2

load mnist_all;
test0_y=[W'*(double(test0'/256));ones(1,size(test0,1))];
test1_y=[W'*(double(test1'/256));ones(1,size(test1,1))];
test2_y=[W'*(double(test2'/256));ones(1,size(test2,1))];
test3_y=[W'*(double(test5'/256));ones(1,size(test5,1))];
test4_y=[W'*(double(test8'/256));ones(1,size(test8,1))];


pre_label_0=exp(theta2*test0_y);
[score index1] = max(pre_label_0);
accuracy(1)= sum(index1==1);

pre_label_1=exp(theta2*test1_y);
[score index1] = max(pre_label_1);
accuracy(2)= sum(index1==2);

pre_label_2=exp(theta2*test2_y);
[score index1] = max(pre_label_2);
accuracy(3)= sum(index1==3);
```

Multi-class logistic

多类情况3

3000次 test_accuracy = 0.8655， validation acc = 0.8645

3500次 test_accuracy = 0.8769， validation acc = 0.8791

5000次 test_accuracy = 0.8969， validation acc = 0.9013

7000次 test_accuracy = 0.9054， validation acc = 0.9139

9000次 test_accuracy = 0.9108， validation acc = 0.9172

11000次 test_accuracy = 0.9134， validation acc = 0.9211

应用一手写数字（5类）

LR 分类

- 利用 MLE估计参数
- 如果有参数的正则项，如何估计参数?

$$l(\boldsymbol{w}) = \log L(\boldsymbol{w}) = \sum_{i=1}^{N} y^i \log f(x^i, \boldsymbol{w}) + (1 - y^i) \log(1 - f(x^i, \boldsymbol{w})) + \lambda \|\boldsymbol{w}\|_2^2$$

Homework (necessary)

- 第六章作业——必做.pdf