# Softmax回归-梯度推导

## OutLine

- Softmax梯度推导
- 正则化
- 【实践】Softmax实践

## 基本符号

- m个训练样本： $\{(x_1,y_1),(x_2,y_2),\ldots,(x_m,y_m)\}$
- 每一个样本，包含n维特征，第i个样本表示： $x_i=\left\{x_i^{(1)},x_i^{(2)},\ldots,x_i^{(n)}\right\}$
- 每个样本的标签y有D种选择，第i个样本的标签集合： $y_i=\left\{y_i^{(1)},y_i^{(2)},\ldots,y_i^{(D)}\right\}$
- 以“数字识别”为例，样本维度28*28=784维度，n=784，输出标签维度D=10
  - 对于每种标签label，存在D=10个对应的权重向量： $\theta=\{\theta_1,\theta_2,\ldots,\theta_D\}$
  - 有D=10组模型

## Softmax回顾：

- 第i个样本，属于d的标签的概率为：$p(y_i=d|x_i,\theta)=\frac{e^{\theta_d^T x_i}}{\sum_{k=1}^{D}e^{\theta_k^T x_i}}$
  - 所以，第d模型训练的参数为$\theta_d$，输入样本在这个模型上的得分为 $\theta_d^T x_i$
  - 于是，每个模型的最终得分为：$\theta_1^T x_i,\theta_2^T x_i,\ldots,\theta_d^T x_i,\ldots,\theta_D^T x_i$
  - 最终：归一化后，每个得分为：

$$\frac{e^{\theta_1^T x_i}}{\sum_{k=1}^{D}e^{\theta_k^T x_i}},\frac{e^{\theta_2^T x_i}}{\sum_{k=1}^{D}e^{\theta_k^T x_i}},\ldots,\frac{e^{\theta_d^T x_i}}{\sum_{k=1}^{D}e^{\theta_k^T x_i}},\ldots,\frac{e^{\theta_D^T x_i}}{\sum_{k=1}^{D}e^{\theta_k^T x_i}}$$

向量形式

$$\begin{Bmatrix} p(y_i=1|x_i,\theta_1) \\ p(y_i=2|x_i,\theta_2) \\ \ldots\ldots \\ p(y_i=d|x_i,\theta_d) \\ \ldots \\ p(y_i=D|x_i,\theta_D) \end{Bmatrix}=\begin{Bmatrix} \frac{e^{\theta_1^T x_i}}{\sum_{k=1}^{D}e^{\theta_k^T x_i}} \\ \frac{e^{\theta_2^T x_i}}{\sum_{k=1}^{D}e^{\theta_k^T x_i}} \\ \ldots. \\ \frac{e^{\theta_d^T x_i}}{\sum_{k=1}^{D}e^{\theta_k^T x_i}} \\ \ldots. \\ \frac{e^{\theta_D^T x_i}}{\sum_{k=1}^{D}e^{\theta_k^T x_i}} \end{Bmatrix}$$

## Loss function

- 逻辑回归的loss function表示为：

$$loss = -\frac{1}{m}(\sum_{i=1}^{m} y_i log(h(x_i)) + (1-y_i)log(1-h(x_i)))$$

简化 $\Rightarrow$

$$loss = -\frac{1}{m}\sum_{i=1}^{m}\sum_{j=0}^{1} I(y_i = j)log(h_j(x_i))$$

- 将此，推导到多分类的softmax loss，存在D个标签，总样本数m个：

$$J(\theta) = -\frac{1}{m}[\sum_{i=1}^{m}\sum_{d=1}^{D} I(y_i = j)log(\frac{e^{\theta_d^T x_i}}{\sum_{k=1}^{D} e^{\theta_k^T x_i}})]$$

## Loss最小化

$$J(\theta)=-\frac{1}{m}[\sum_{i=1}^{m}\sum_{d=1}^{D}I(y_i=j)log(\frac{e^{\theta_d^T x_i}}{\sum_{k=1}^{D}e^{\theta_k^T x_i}})]$$

- J(θ)对θ求导，是一个矩阵，表示为：$$\frac{\partial J(\theta)}{\partial \theta}=\left\{\frac{\partial J(\theta)}{\partial \theta_1},\frac{\partial J(\theta)}{\partial \theta_2},\ldots,\frac{\partial J(\theta)}{\partial \theta_d},\ldots,\frac{\partial J(\theta)}{\partial \theta_D}\right\}$$
- 此外，$\theta_d$是一个与输入样本维度一致的n维向量，表示为：$$\frac{\partial J(\theta)}{\partial \theta_d}=\left\{\frac{\partial J(\theta)}{\partial \theta_d^{(1)}},\frac{\partial J(\theta)}{\partial \theta_d^{(2)}},\ldots,\frac{\partial J(\theta)}{\partial \theta_d^{(D)}}\right\}$$
- 由与D个标签，所以对应D个θ，每个θ维度为n，所以最终θ形式为：

$$\theta=\{\theta_1,\theta_2,\ldots,\theta_d,\ldots,\theta_D\}=\left\{\begin{matrix}\theta_1^{(1)},\theta_2^{(1)},\ldots,\theta_d^{(1)},\ldots,\theta_D^{(1)}\\ \theta_1^{(2)},\theta_2^{(2)},\ldots,\theta_d^{(2)},\ldots,\theta_D^{(2)}\\ \ldots\\ \theta_1^{(d)},\theta_2^{(d)},\ldots,\theta_d^{(d)},\ldots,\theta_D^{(d)}\\ \ldots\\ \theta_1^{(N)},\theta_2^{(N)},\ldots,\theta_d^{(N)},\ldots,\theta_D^{(N)}\end{matrix}\right\}$$

- 所以：$\frac{\partial J(\theta)}{\partial \theta}$是一个矩阵

  $\frac{\partial J(\theta)}{\partial \theta_d}$是一个向量

## 推 导

- J(θ)展开为：

$$J(\theta)=-\frac{1}{m}\sum_{i=1}^{m}[I(y_i=1)log(\frac{e^{\theta_1^Tx_i}}{\sum_{k=1}^{D}e^{\theta_k^Tx_i}})+I(y_i=2)log(\frac{e^{\theta_2^Tx_i}}{\sum_{k=1}^{D}e^{\theta_k^Tx_i}})+\ldots+I(y_i=d)log(\frac{e^{\theta_d^Tx_i}}{\sum_{k=1}^{D}e^{\theta_k^Tx_i}})+\ldots+I(y_i=D)log(\frac{e^{\theta_D^Tx_i}}{\sum_{k=1}^{D}e^{\theta_k^Tx_i}})]$$

- 求$\frac{\partial J(\theta)}{\partial \theta_c}$，其中c属于{1,2,3,..,d,...,D}，所以分两种情况：
  - d=c
  - d!=c
  - 接下来，我们先仅考虑，取p(yi=d|xi,θd)对于$\theta_d$的导数

$$p(y_i=d|x_i,\theta)=\frac{e^{\theta_d^Tx_i}}{\sum_{k=1}^{D}e^{\theta_k^Tx_i}}$$

## 推 导

- 第一种情况：d=c

$$\frac{\partial p(y_i=d|x_i,\theta_d)}{\partial\theta_c}=\frac{\partial p(y_i=d|x_i,\theta_d)}{\partial\theta_d}=\frac{\partial\left(\frac{e^{\theta_d^T x_i}}{\sum_{k=1}^{D}e^{\theta_k^T x_i}}\right)}{\partial\theta_d}$$

$$=\frac{x_i e^{\theta_d^T x_i}\sum_{k=1}^{D}e^{\theta_k^T x_i}-e^{\theta_d^T x_i}x_i e^{\theta_d^T x_i}}{(\sum_{k=1}^{D}e^{\theta_k^T x_i})^2}$$

$$=x_i\frac{e^{\theta_d^T x_i}}{\sum_{k=1}^{D}e^{\theta_k^T x_i}}\frac{\sum_{k=1}^{D}e^{\theta_k^T x_i}-e^{\theta_d^T x_i}}{\sum_{k=1}^{D}e^{\theta_k^T x_i}}$$

$$=x_i p(y_i=d|x_i,\theta_d)[1-p(y_i=d|x_i\theta_d)]$$

## 推 导

- 第二种情况：d!=c

$$\frac{\partial p(y_i=d|x_i,\theta_d)}{\partial \theta_c} = e^{\theta_d^T x_i} \frac{-x_i e^{\theta_c^T x_i}}{(\sum_{k=1}^{D} e^{\theta_k^T x_i})^2}$$

$$= -x_i \frac{e^{\theta_d^T x_i}}{\sum_{k=1}^{D} e^{\theta_k^T x_i}} \frac{e^{\theta_c^T x_i}}{\sum_{k=1}^{N} e^{\theta_k^T x_i}}$$

$$= -x_i p(y_i=d|x_i,\theta_d) p(y_i=c|x_i,\theta_c)$$

## 推 导

d=c

$$x_i p(y_i=d|x_i,\theta_d)[1-p(y_i=d|x_i\theta_d)]$$

d!=c

$$-x_i p(y_i=d|x_i,\theta_d)p(y_i=c|x_i,\theta_c)$$

- 将两种情况的结果综合到 $\frac{\partial J(\theta)}{\partial \theta_c}$

$$\frac{\partial J(\theta)}{\partial \theta_c} = -\frac{1}{m}\sum_{i=1}^{m}[\sum_{j\neq c} I(y_i=j)\frac{1}{p(y_i=j|x_i,\theta_j))}(-x_i)(p(y_i=j|x_i,\theta_j)p(y_i=c|x_i,\theta_c))+$$

$$I(y_i=c)\frac{1}{p(y_i=c|x_i,\theta_c)}x_i p(y_i=c|x_i,\theta_c)(1-p(y_i=c|x_i,\theta_c))]$$

$$= -\frac{1}{m}\sum_{i=1}^{m}x_i[-\sum_{y_i\neq c} I(y_i=j)p(y_i=c|x_i,\theta_c)+I(y_i=c)(1-p(y_i=c|x_i,\theta_c))]$$

$$= -\frac{1}{m}\sum_{i=1}^{m}x_i[I(y_i=c)-P(y_i=c|x_i,\theta_c)\sum_{i=1}^{D}I(y_i=j)]$$

$$= -\frac{1}{m}\sum_{i=1}^{m}x_i[I(y_i=c)-P(y_i=c|x_i,\theta_c)]$$

- 由于 $\sum_{j=1}^{D} I(y_i=j)=1$
- 最终： $\frac{\partial J(\theta)}{\partial \theta_c} = -\frac{1}{m}\sum_{i=1}^{m}x_i[I(y_i=c)-P(y_i=c|x_i,\theta_c)]$

L2正则

$$J(\theta) = -\frac{1}{m}[\sum_{i=1}^{m}\sum_{d=1}^{D}I(y_i=j)log(\frac{e^{\theta_d^T x_i}}{\sum_{k=1}^{D}e^{\theta_k^T x_i}})] + \frac{\lambda}{2}\sum_{d=1}^{D}||\theta_d||_2^2$$

$$\frac{\partial J(\theta)}{\partial \theta_c} = -\frac{1}{m}\sum_{i=1}^{m}x_i[I(y_i=c)-P(y_i=c|x_i,\theta_c)]+\lambda\theta_c$$

## OutLine

- Softmax理论
- 正则化
- 【实践】Softmax实践

## 正则化（Regularization）

- 机器学习中几乎都可以看到损失函数后会添加一个额外项，通常有两类
  - L1正则：L1范数
  - L2正则：L2范数
- L1正则化和L2正则化上可以看做是损失函数的惩罚项
  - 所谓『惩罚』是指对损失函数中的某些参数做一些限制。
- 对于回归模型：
  - 使用L1正则化的模型建叫做Lasso回归
  - 使用L2正则化的模型叫做Ridge回归（岭回归）

## 正则化（Regularization）

- 举例:
  - L1正则:

$$\min_{w} \frac{1}{2n_{samples}} ||Xw - y||_2^2 + \boxed{\alpha ||w||_1}$$

  - L2正则:

$$\min_{w} ||Xw - y||_2{}^2 + \boxed{\alpha ||w||_2{}^2}$$

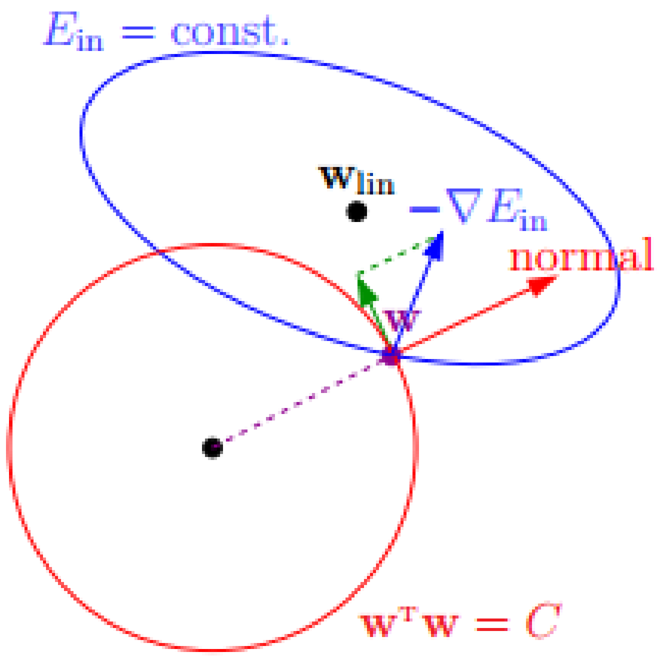
## 正则化（Regularization）

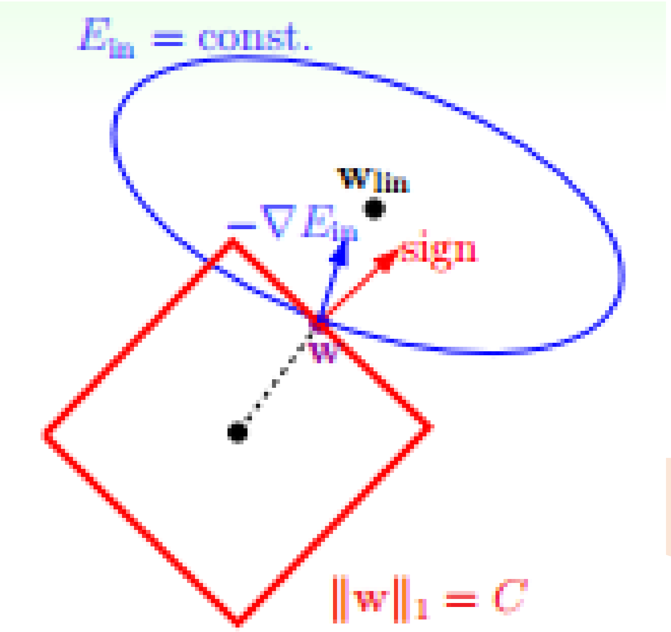
- L1正则：权值向量w中各个元素的绝对值之和，通常表示为$\|w\|_1$
- L2正则：权值向量w中各个元素的平方和然后再求平方根，通常表示为$\|w\|_2$

- 通常正则化项前添加一个系数，由用户指定

- 优点：
  - L1正则化可以产生稀疏权值矩阵，即产生一个稀疏模型，可以用于特征选择
  - L2正则化可以防止模型过拟合（overfitting）；一定程度上，L1也可以防止过拟合

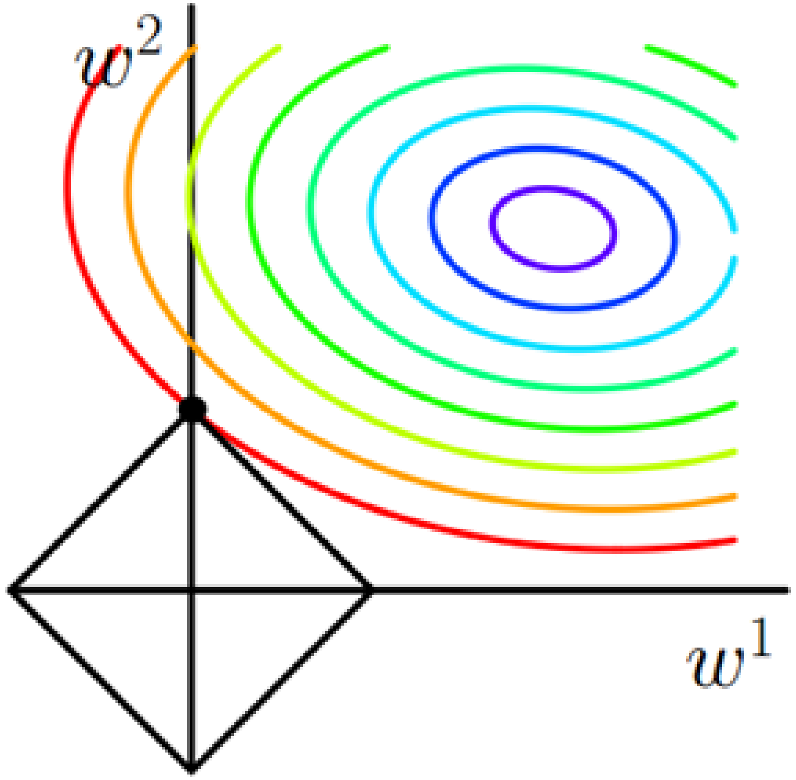
## 直 观 理 解


$E_{\text{in}} = \text{const.}$
$\mathbf{w}_{\text{lin}}$
$-\nabla E_{\text{in}}$
normal
$\mathbf{w}$
$\mathbf{w}^T\mathbf{w} = C$

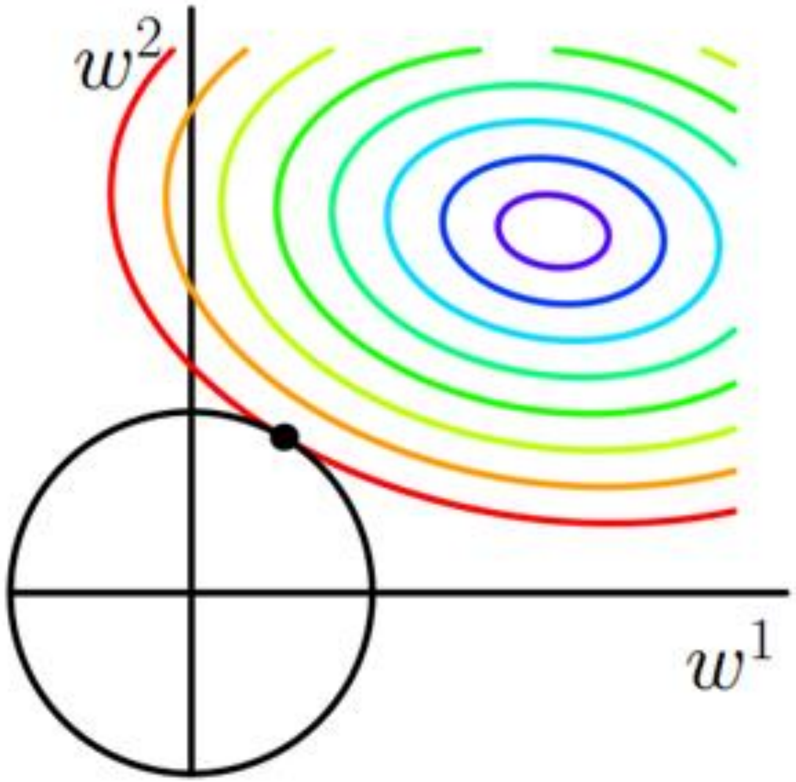

$E_{\text{in}} = \text{const.}$
$\mathbf{w}_{\text{lin}}$
$-\nabla E_{\text{in}}$
sign
$\mathbf{w}$
$\|\mathbf{w}\|_1 = C$

## 直 观 理 解

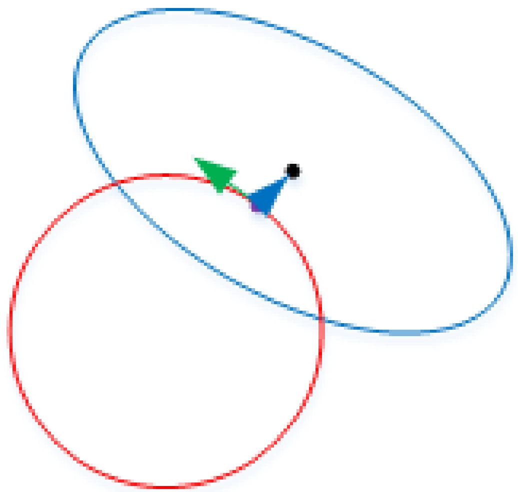
L1正则：




L2正则：

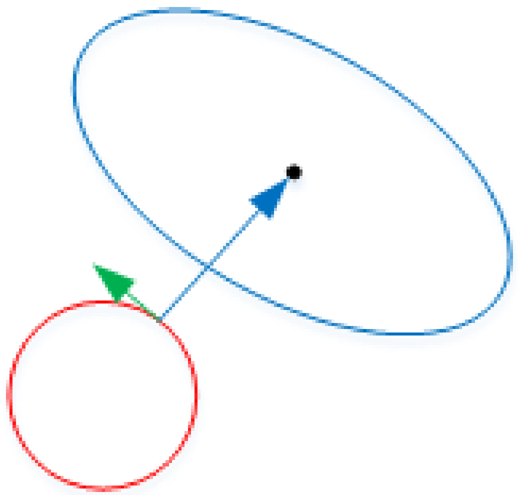


## 直 观 理 解



λ很小，C很大

正则化失效，容易造成过拟合



λ很大，C很小

容易造成欠拟合

## 过 拟 合

- 拟合过程中通常都倾向于让权值尽可能小，最后构造一个所有参数都比较小的模型。因为一般认为参数值小的模型比较简单，能适应不同的数据集，也在一定程度上避免了过拟合现象。
- 可以设想一下对于一个线性回归方程，若参数很大，那么只要数据偏移一点点，就会对结果造成很大的影响；但如果参数足够小，数据偏移得多一点也不会对结果造成什么影响，专业一点的说法是『抗扰动能力强』。

## 过 拟 合

- 为什么L2会控制过拟合？模型为什么可以获得很小的参数?
- 以回归为例：

损失函数：
$$J(\theta)=\frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)})-y^{(i)})$$

迭代公式：
$$\theta_j:=\theta_j-\alpha\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)})-y^{(i)})x_j^{(i)}$$

- 若添加L2后，迭代公式变为

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

可以看到，与未添加L2正则化的迭代公式相比
每一次迭代，θj都要先乘以一个小于1的因子，从而使得θj不断减小，因此总得来看，θ是不断减小的

OutLine

- Softmax理论
- 正则化
- 【实践】Softmax实践

# Q&A

@八斗数据