模式识别 U5 Logistic 回归

课堂内容

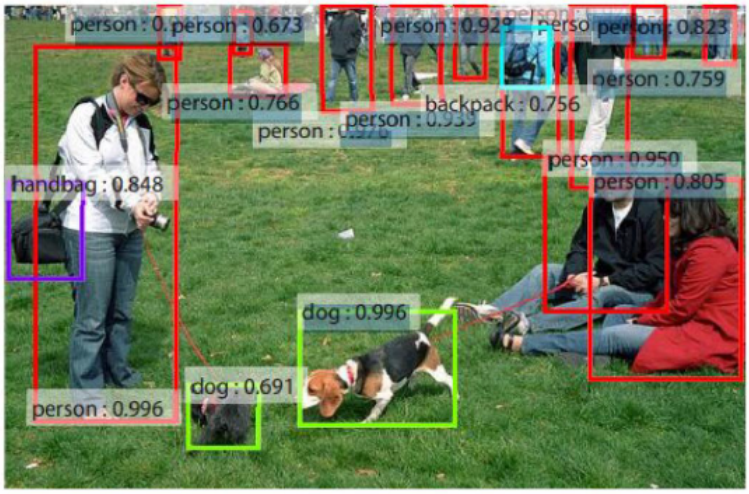
- 5.1 Logistic回归问题
- 5.2 Logistic回归损失
- 5.3 Logistic回归算法
- 5.4 二元分类线性模型讨论

5.1 逻辑斯蒂回归问题

逻辑斯蒂回归是一种“软分类”策略，即考虑分类中的概率性或称模糊性

逻辑回归假设数据服从伯努利分布，通过**极大似然函数**的方法，运用**梯度下降**来求解参数，来达到将**数据二分类**的目的

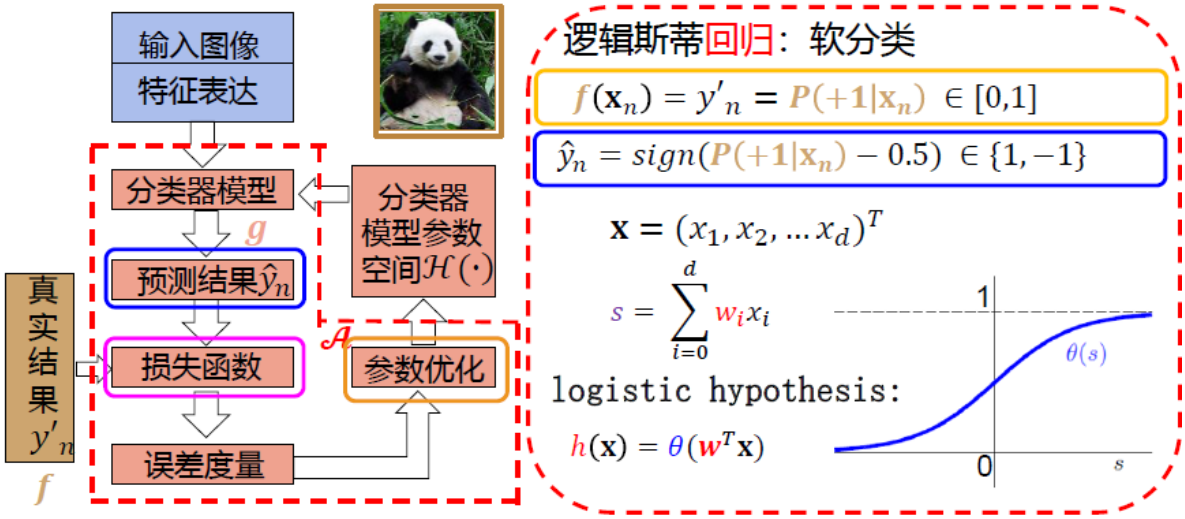
逻辑斯蒂回归应用示例



感知器算法：硬分类

逻辑斯蒂回归：软分类

(*"Soft" binary classification*)



逻辑斯蒂函数

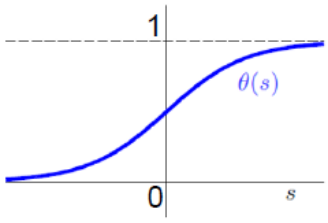
$$\theta(-\infty) = 0; \quad \theta(0) = \frac{1}{2}; \quad \theta(\infty) = 1$$

$$\theta(s) = \frac{e^s}{1+e^s} = \frac{1}{1+e^{-s}}$$

---- Sigmoid 函数：平滑 (*Smooth*)、单调 (*Monotonic*)

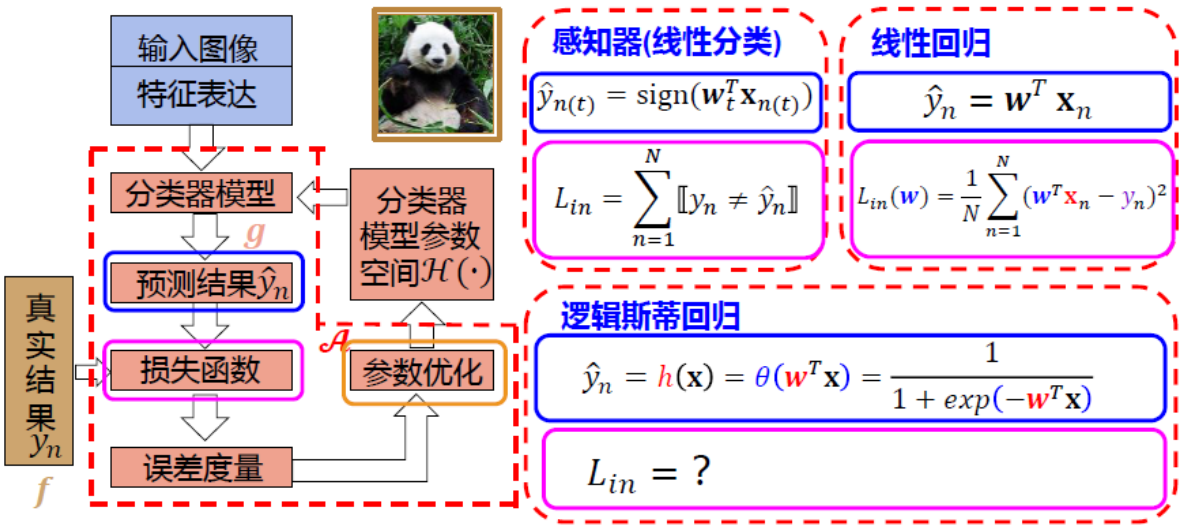
逻辑斯蒂回归用如下模型来估计$f(\mathbf{x}_n)$

$$h(\mathbf{x}) = \theta(\mathbf{w}^T\mathbf{x}) = \frac{1}{1+exp(-\mathbf{w}^T\mathbf{x})}$$



5.2 逻辑斯蒂回归损失

往期损失函数回顾



逻辑斯蒂回归可以使用平方损失函数作为损失函数吗?

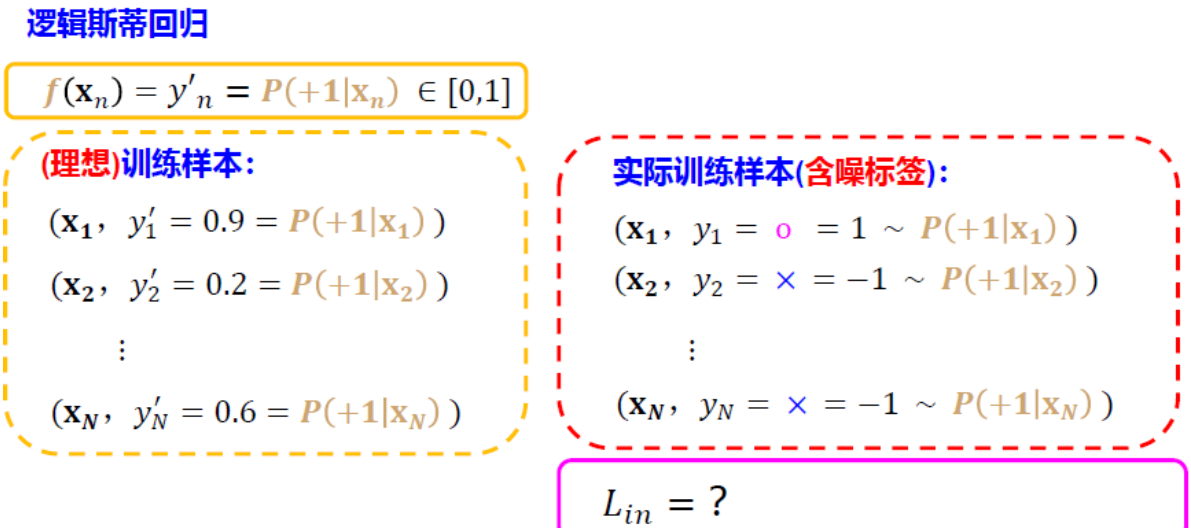
平方损失

数学推演:

$$h(\mathbf{x}) = \theta(\mathbf{w}^T\mathbf{x}) = \frac{1}{1+e^{-\mathbf{w}^T\mathbf{x}}}$$

$$\mathcal{L}_{in}(\mathbf{w}) = (\theta(\mathbf{w}^T\mathbf{x}) - y)^2$$

存在一个问题:



由实际训练样本标签带来的影响，我们改写损失函数形式为：

$$\mathcal{L}_{in}(\mathbf{w}) = (\theta(y\mathbf{w}^T\mathbf{x}) - 1)^2$$

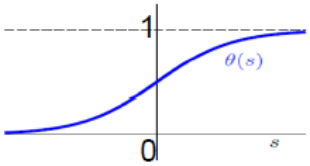
由此进行梯度推演：

$$\begin{aligned}
\frac{\partial \mathcal{L}_{in}(\mathbf{w},\mathbf{x},y)}{\partial w_i} &= 2[\theta(y\mathbf{w}^T\mathbf{x}) - 1]\frac{\partial \theta(\mathbf{w})}{\partial w_i} \\
&= 2[\theta(y\mathbf{w}^T\mathbf{x}) - 1]\frac{yx_i e^{-y\mathbf{w}^T\mathbf{x}}}{(1+e^{-y\mathbf{w}^T\mathbf{x}})^2} \qquad \theta(y\mathbf{w}^T\mathbf{x}) > 0(\text{正确分类})\ \ \nabla\mathcal{L} \to 0 \\
&= 2[\theta(y\mathbf{w}^T\mathbf{x}) - 1]yx_i\frac{1}{(1+e^{-y\mathbf{w}^T\mathbf{x}})}\frac{e^{-y\mathbf{w}^T\mathbf{x}}}{(1+e^{-y\mathbf{w}^T\mathbf{x}})} \\
&= 2[\theta(y\mathbf{w}^T\mathbf{x}) - 1]yx_i\theta(y\mathbf{w}^T\mathbf{x})[1-\theta(y\mathbf{w}^T\mathbf{x})] \\
&\qquad \theta(y\mathbf{w}^T\mathbf{x}) < 0(\text{错误分类})\ \ \nabla\mathcal{L} \to 0
\end{aligned}$$

逻辑斯蒂回归可以使用平方误差作为损失函数吗?

$$L_{in}(\boldsymbol{w}) = (\theta(y\boldsymbol{w}^T\mathbf{x}) - 1)^2$$

$$\frac{\partial L_{in}(\boldsymbol{w},\mathbf{x},y)}{\partial w_i} = 2(\theta(y\boldsymbol{w}^T\mathbf{x}) - 1)\underbrace{\theta(y\boldsymbol{w}^T\mathbf{x})(1\text{-}\theta(y\boldsymbol{w}^T\mathbf{x}))}_{\frac{\partial\theta(z)}{\partial z}}yx_i$$



$$if\ \ (y\boldsymbol{w}^T\mathbf{x})) > 0 \quad \nabla L_{in}(\boldsymbol{w},\mathbf{x},y) = 0$$

$$if\ \ (y\boldsymbol{w}^T\mathbf{x})) < 0 \quad \nabla L_{in}(\boldsymbol{w},\mathbf{x},y) = 0$$

交叉熵损失

基本概念

相对熵又称为KL散度 (Kullback–Leibler divergence)，用来描述两个概率分布的差异性。假设有对同一变量x的$q(x)$和$p(x)$两个概率分布,那么两者之间的相对熵可由以下定义:

$$D_{KL}(p\|q) = \sum_{i=1}^{N} p(x_i)\log\left(\frac{p(x_i)}{q(x_i)}\right)$$

对于实际应用，$p(x)$是目标分布，$q(x)$是预测的匹配分布。

2、交叉熵

- 简单概念
  交叉熵是信息熵论中的概念，它原本是用来估算平均编码长度的。在深度学习中，可以看作通过概率分布$q(x)$表示概率分布$p(x)$的困难程度。其表达式为：

$$H(p,q)=\sum_{i=1}^{n}p(x_i)\log\frac{1}{q(x_i)}=-\sum_{i=1}^{n}p(x_i)\log q(x_i)$$

- 简单性质
  交叉熵刻画的是两个概率分布的距离，也就是说交叉熵值越小（相对熵的值越小），两个概率分布越接近。
  因为上面说到，当$q(x)$=$p(x)$时，$D_{KL}(p\|q)$才有最小值，即下式取等号。

$$D_{KL}(p\|q)=-H(p)+H(p,q)\geq 0\Rightarrow H(p,q)\geq H(p)$$

下面将给出两个具体样例来直观地说明通过交叉熵可以判断预测答案和真实答案之间的距离。假设有个三分类问题，某个正确答案和一个经过$softmax$回归后的预测答案如下：

	x_1	x_2	x_3
$p(x_i)$	1	0	0
$q_1(x_i)$	0.5	0.4	0.1
$q_2(x_i)$	0.8	0.1	0.1

那么$p(x)$与$q_1(x)$的交叉熵为：（log计算取10为底，即lg）

$$\mathrm{H}((1,0,0),(0.5,0.4,0.1))=-(1\times\log 0.5+0\times\log 0.4+0\times\log 0.1)\approx 0.3$$

$p(x)$与$q_2(x)$的交叉熵为：

$$\mathrm{H}((1,0,0),(0.8,0.1,0.1))=-(1\times\log 0.8+0\times\log 0.1+0\times\log 0.1)\approx 0.1$$

从直观上可以看到第二个预测的结果要优于第一个，并且通过计算交叉熵，结果也是一致的。

逻辑斯蒂回归的最佳解：

$$g=\underset{\boldsymbol{w}}{argmin}\frac{1}{N}\sum_{n=1}^{N}-ln\theta(y_n\boldsymbol{w}^T\mathbf{x}_n)$$

$$\theta(y_n\boldsymbol{w}^T\mathbf{x}_n)=\frac{1}{1+\exp(-y_n\boldsymbol{w}^T\mathbf{x}_n)}$$

$$g=\underset{\boldsymbol{w}}{argmin}\frac{1}{N}\sum_{n=1}^{N}\ln(1+exp(-y_n\boldsymbol{w}^T\mathbf{x}_n))$$

交叉熵损失
(*Cross-Entropy Loss*)

$$L_{in}=\frac{1}{N}\sum_{n=1}^{N}\ln(1+exp(-y_n\boldsymbol{w}^T\mathbf{x}_n))$$

交叉熵介绍：

Distribution p(x)
$p(x=1)=f(x_n)$
$p(x=0)=1-f(x_n)$

cross entropy

Distribution q(x)
$q(x=1)=\theta(x_n)$
$q(x=0)=1-\theta(x_n)$

$$H(p,q)=-\sum_x p(x)\ln(q(x))$$

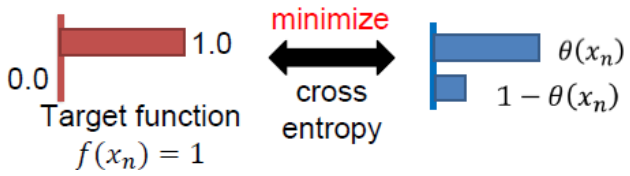
$$s=w^Tx$$

$$\theta(s)=\frac{1}{1+\exp(-s)}$$

$$H(f(x_n),\theta(x_n)))=\sum_n -[f(x_n)ln\theta(x_n)+(1-f(x_n))ln(1-ln\theta(x_n))]$$

$$H(f(x_n),\theta(x_n)))=\sum_n -[ln\theta(x_n)]$$

$$H(f(x_n),\theta(x_n)))=\sum_n [\ln(1+\exp(-ys)]$$



交叉熵损失梯度：

$$L_{in}=\ln(1+exp(-y_n\boldsymbol{w}^T\mathbf{x}_n))$$

$$\frac{\partial L_{in}(\boldsymbol{w},\mathbf{x},y)}{\partial w_i}=\frac{\partial \ln(\blacksquare)}{\partial \blacksquare}\frac{\partial(1+\exp(\bullet))}{\partial \bullet}\frac{\partial(-y\boldsymbol{w}^T\mathbf{x})}{\partial w_i}$$

$$=\frac{1}{\blacksquare}\exp(\bullet)(-yx_i)=\frac{\exp(\bullet)}{1+\exp(\bullet)}(-yx_i)$$

$$\nabla L_{in}(\boldsymbol{w},\mathbf{x},y)=\theta(-y\boldsymbol{w}^T\mathbf{x})(-y\mathbf{x})$$

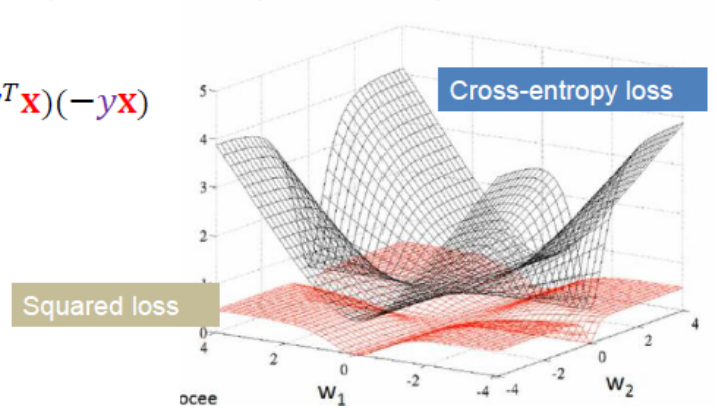
交叉熵损失与平方损失的梯度对比：

平方损失的梯度：

$$\nabla L_{in}(\boldsymbol{w},\mathbf{x},y)=2(\theta(y\boldsymbol{w}^T\mathbf{x})-1)\theta(y\boldsymbol{w}^T\mathbf{x})(1\text{-}\theta(y\boldsymbol{w}^T\mathbf{x}))y\mathbf{x}$$

交叉熵损失的梯度：

$$\nabla L_{in}(\boldsymbol{w},\mathbf{x},y)=\theta(-y\boldsymbol{w}^T\mathbf{x})(-y\mathbf{x})$$



5.3 逻辑斯蒂回归算法

逻辑斯蒂回归

$$\hat{y}_n = h(\mathbf{x}) = \theta(\boldsymbol{w}^T\mathbf{x}) = \frac{1}{1+exp(-\boldsymbol{w}^T\mathbf{x})}$$

$$L_{in} = \frac{1}{N}\sum_{n=1}^{N}\ln(1+exp(-y_n\boldsymbol{w}^T\mathbf{x}_n))$$

$$\nabla L_{in}(\boldsymbol{w},\mathbf{x},y) = \frac{1}{N}\sum_{n=1}^{N}\theta(-y_n\boldsymbol{w}^T\mathbf{x}_n)(-y_n\mathbf{x}_n)$$

梯度下降法实现逻辑斯蒂回归

- 初始化权向量$\boldsymbol{w}_0$
- $\boldsymbol{for}\ t = 0,1,2,\ldots$（$t$ 代表迭代次数）

① 计算梯度：$\nabla L_{in}(\boldsymbol{w}_t) = \frac{1}{N}\sum_{n=1}^{N}\theta(-y_n\boldsymbol{w}_t^T\mathbf{x}_n)(-y_n\mathbf{x}_n)$

② 对权向量$\boldsymbol{w}_t$进行更新：$\boldsymbol{w}_{t+1} \leftarrow \boldsymbol{w}_t - \eta\nabla L_{in}(\boldsymbol{w}_t)$

…直到$\nabla L_{in}(\boldsymbol{w}) = \mathbf{0}$，或者迭代足够多次数

返回最终的$\boldsymbol{w}_{t+1}$作为学到的g

梯度下降法实现逻辑斯蒂回归

- 初始化权向量$\boldsymbol{w}_0$ **Stochastic Gradient Descent(SGD)**
- $\boldsymbol{for}\ t = 0,1,2,\ldots$（$t$ 代表迭代次数）

① 计算梯度：$\nabla L_{in}(\boldsymbol{w}_t) = \frac{1}{B}\sum_{n=1}^{B}\theta(-y_n\boldsymbol{w}_t^T\mathbf{x}_n)(-y_n\mathbf{x}_n)$

② 对权向量$\boldsymbol{w}_t$进行更新：$\boldsymbol{w}_{t+1} \leftarrow \boldsymbol{w}_t - \eta\nabla L_{in}(\boldsymbol{w}_t)$

…直到$\nabla L_{in}(\boldsymbol{w}) = \mathbf{0}$，或者迭代足够多次数

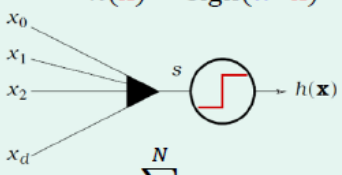
返回最终的$\boldsymbol{w}_{t+1}$作为学到的g

5.4 二元分类线性模型讨论

三种线性模型比较

线性分类(感知器):

$h(\mathbf{x}) = \text{sign}(\boldsymbol{w}^T\mathbf{x})$

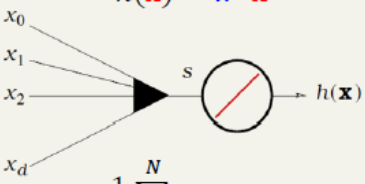


$L_{in} = \sum_{n=1}^{N} [\![y_n \neq \hat{y}_n]\!]$

NP难问题

线性回归:

$h(\mathbf{x}) = \boldsymbol{w}^T\mathbf{x}$

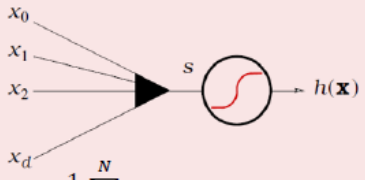


$L_{in} = \frac{1}{N}\sum_{n=1}^{N}(\boldsymbol{w}^T\mathbf{x}_n - y_n)^2$

凸函数、易优化、解析解

逻辑斯蒂回归:

$h(\mathbf{x}) = \theta(\boldsymbol{w}^T\mathbf{x})$



$L_{in} = \frac{1}{N}\sum_{n=1}^{N}\ln(1 + exp(-y_n\boldsymbol{w}^T\mathbf{x}_n))$

凸函数、平滑、梯度下降

三种线性模型用于二元分类时(即: $y \in \{+1, -1\}$)损失函数比较

样本特征向量 $\mathbf{x}$ 与模型的权向量 $\boldsymbol{w}$ 的内积用 s 表示: $s = \boldsymbol{w}^T\mathbf{x}$

线性分类(感知器):

$h(\mathbf{x}) = \text{sign}(s)$

$L_{in} = [\![h(\mathbf{x}) \neq y]\!]$

$L_{0/1}(s, y) = [\![\text{sign}(s) \neq y]\!]$
$= [\![\text{sign}(ys) \neq 1]\!]$

线性回归:

$h(\mathbf{x}) = s$

$L_{in} = (h(\mathbf{x}) - y)^2$

$L_{sqr}(s, y) = (s - y)^2$
$= (ys - 1)^2$

逻辑斯蒂回归:

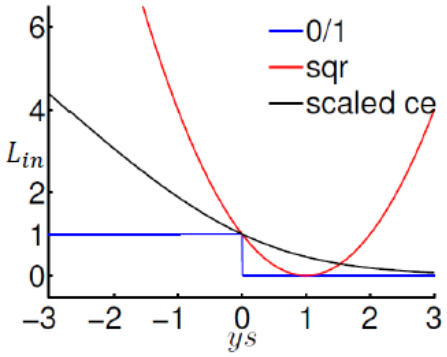
$h(\mathbf{x}) = \theta(s)$

$L_{in} = -\ln(h(y\mathbf{x}))$

$L_{ce} = \ln(1 + exp(-ys))$

三种线性模型用于二元分类时(即: $y \in \{+1, -1\}$)损失函数比较

样本特征向量 $\mathbf{x}$ 与模型的权向量 $\boldsymbol{w}$ 的内积用 s 表示: $s = \boldsymbol{w}^T\mathbf{x}$



0/1 $L_{0/1}(s, y) = [\![sign(ys) \neq 1]\!]$

sqr $L_{sqr}(s, y) = (ys - 1)^2$

ce $L_{ce}(s, y) = ln(1 + \exp(-ys))$

Scaled ce $L_{sce}(s, y) = log_2(1 + \exp(-ys))$

$$L_{0/1}(s,y) \le L_{sqr}(s,y)$$
$$L_{0/1}(s,y) \le L_{sce}(s,y)$$
$$L_{0/1}(s,y) \le L_{ce}(s,y)$$

训练或测试时，只要做到 $L_{sqr}(s,y)$或者 $L_{ce}(s,y)$ 很小，$L_{0/1}(s,y)$也会很小

线性回归与逻辑斯蒂回归可用于线性分类

① 在标签为{+1，-1}的训练样本集 $\mathcal{D}$ 上运行线性回归/逻辑斯蒂回归算法，得到$\boldsymbol{w}^*$

② 返回分类结果：$g(\mathbf{x}) = sign(\boldsymbol{w}^{*T}\mathbf{x})$

线性分类(感知器)：

优点：样本线性可分时，算法收敛有理论保障

不足：样本非线性可分时NP难问题，可用Pocket算法实现

线性回归：

优点：凸函数，最容易优化，有解析解

不足：当$|ys|$很大时，$L_{0/1}(s,y)$的上界过于宽松

逻辑斯蒂回归：

优点：凸函数，易于优化

不足：当$ys \ll 0$时，$L_{0/1}(s,y)$的上界过于宽松

小结

5.1 逻辑斯蒂回归问题

模型的理论输出为概率值，分类面假设空间模型用Sigmoid函数

5.2 逻辑斯蒂回归损失

用交叉熵(cross-entropy)作为损失函数

5.3 逻辑斯蒂回归算法

用梯度下降法迭代实现参数更新

5.4 二元分类线性模型讨论

三个线性模型的特点及用途