中国科学院南京分院
Nanjing Branch of Chinese Academy of Sciences

人工智能原理与算法

3. Logistic回归

夏睿

2023.2.24

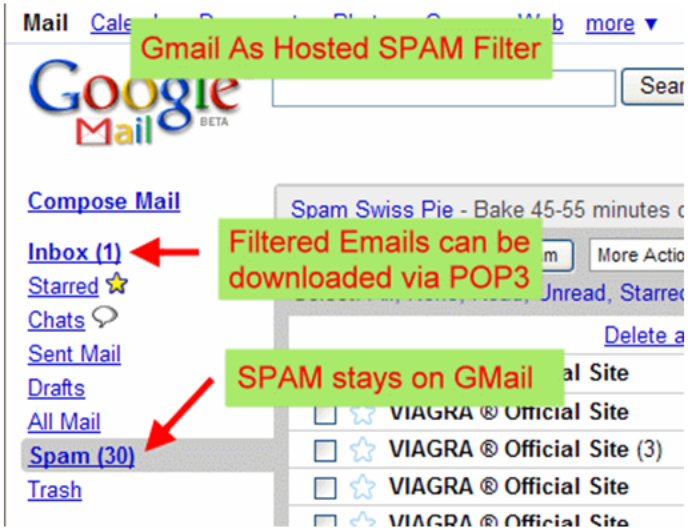
目录

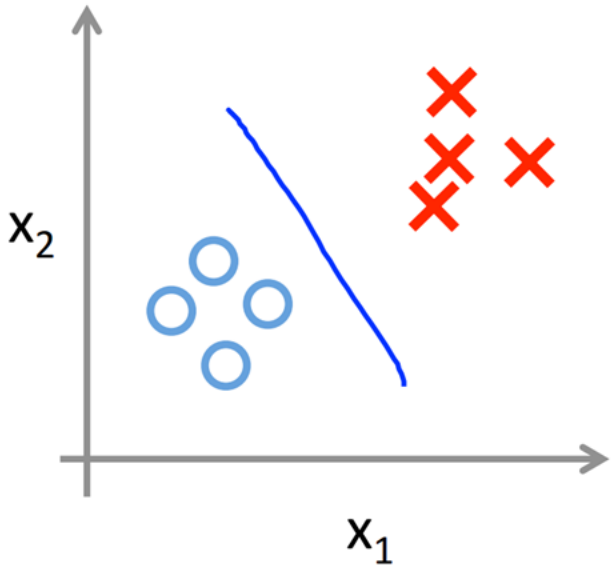
- Logistic回归
- Softmax回归
- 分类任务的性能评估

Logistic回归

Logistic回归

- 一种分类模型，尽管它被称作“回归”；
- 一种二分类模型；
- 一种线性分类模型：它具有线性决策边界（超平面），尽管它用一个非线性激活函数（Sigmoid函数）来计算后验概率。



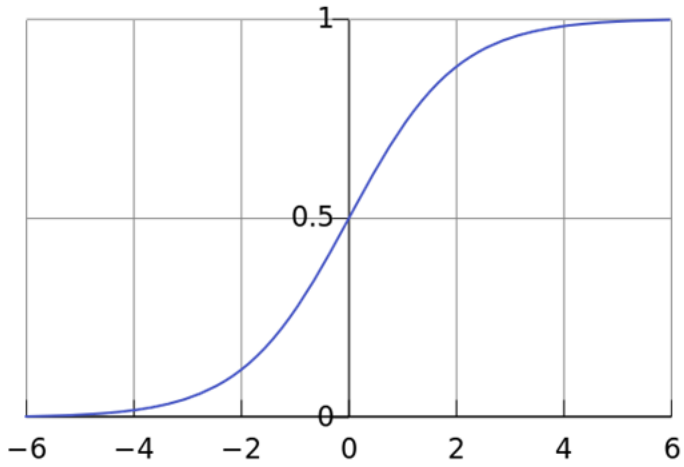


模型假设

- Sigmoid 函数

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

$$\frac{d\delta(z)}{dz} = \frac{e^{-z}}{(1+e^{-z})^2} = \delta(z)\,(1-\delta(z))$$



Sigmoid · (1 − Sigmoid)

- 模型假设

$$p(y=1|\boldsymbol{x};\boldsymbol{\theta}) = h(\boldsymbol{x}) = \delta\left(\boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{x}\right) = \frac{1}{1+e^{-\boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{x}}}$$

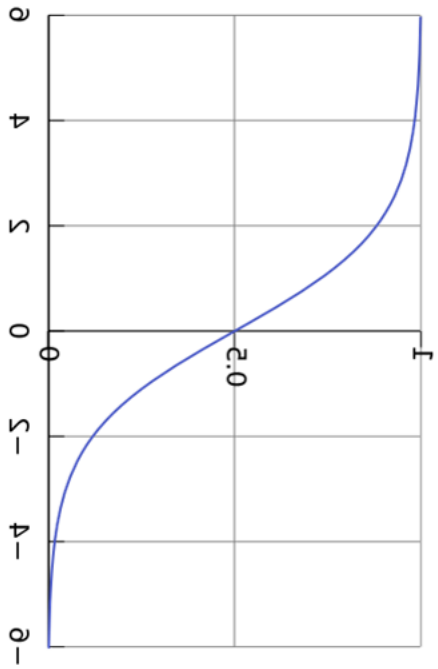
$$p(y=0\,|\boldsymbol{x};\boldsymbol{\theta}) = 1 - h(\boldsymbol{x})$$

为何称作“Logistic回归”？

- Logit函数：Sigmoid的逆函数

$$p = \frac{1}{1+e^{-z}} \quad \Rightarrow \quad z = \log\frac{p}{1-p}$$

Logit 也称作log-odds （对数几率）



- 回顾线性回归

$$h = \boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{x}$$

- Logistic回归 = 针对Logit的线性回归

$$\text{logit} = \log\frac{p}{1-p} = \boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{x} \quad \Rightarrow \quad p = \frac{1}{1+e^{-\boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{x}}}$$

学习算法

- 似然函数（条件分布）

$$
\begin{aligned}
L(\boldsymbol{\theta}) &= \prod_{k=1}^{N} p\left(y^{(k)} \middle| \boldsymbol{x}^{(k)}; \boldsymbol{\theta}\right) \\
&= \prod_{k=1}^{N} \left(h\left(\boldsymbol{x}^{(k)}\right)\right)^{y^{(k)}} \left(1 - h\left(\boldsymbol{x}^{(k)}\right)\right)^{\left(1-y^{(k)}\right)} \\
&= \prod_{k=1}^{N} \left(\frac{1}{1+e^{-\boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{x}^{(k)}}}\right)^{y^{(k)}} \left(1 - \frac{1}{1+e^{-\boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{x}^{(k)}}}\right)^{\left(1-y^{(k)}\right)}
\end{aligned}
$$

- 最大似然估计

$$
\max_{\boldsymbol{\theta}} l(\boldsymbol{\theta}) = \sum_{k=1}^{N} y^{(k)} \log \frac{1}{1+e^{-\boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{x}^{(k)}}} + \left(1-y^{(k)}\right) \log \left(1 - \frac{1}{1+e^{-\boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{x}^{(k)}}}\right)
$$

无约束优化

- 无约束最优化问题

$$\max_{\boldsymbol{\theta}} \sum_{k=1}^{N} y^{(k)} \log h\left(\boldsymbol{x}^{(k)}\right) + \left(1 - y^{(k)}\right) \log\left(1 - h\left(\boldsymbol{x}^{(k)}\right)\right)$$

- 数值化最优化方法
  - 梯度下降
  - 随机梯度下降
  - 牛顿法
  - 拟牛顿法
  - 共轭梯度法
  - ……

梯度上升法

- 梯度计算

$$\frac{dl(\boldsymbol{\theta})}{d\boldsymbol{\theta}} = \frac{d\sum_{k=1}^{N} y^{(k)}\log h\left(\boldsymbol{x}^{(k)}\right) + \left(1-y^{(k)}\right)\log\left(1-h\left(\boldsymbol{x}^{(k)}\right)\right)}{d\boldsymbol{\theta}}$$

$$= \sum_{k=1}^{N}\left(y^{(k)}\frac{1}{h(\boldsymbol{x}^{(k)})} - \left(1-y^{(k)}\right)\frac{1}{1-h(\boldsymbol{x}^{(k)})}\right)\frac{d}{d\boldsymbol{\theta}}h\left(\boldsymbol{x}^{(k)}\right)$$

$$= \sum_{k=1}^{N}\left(y^{(k)}\frac{1}{h(\boldsymbol{x}^{(k)})} - \left(1-y^{(k)}\right)\frac{1}{1-h(\boldsymbol{x}^{(k)})}\right)h\left(\boldsymbol{x}^{(k)}\right)\left(1-h\left(\boldsymbol{x}^{(k)}\right)\right)\frac{d}{d\boldsymbol{\theta}}\boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{x}^{(k)}$$

$$= \sum_{k=1}^{N}\left(y^{(k)}\left(1-h\left(\boldsymbol{x}^{(k)}\right)\right) - \left(1-y^{(k)}\right)h\left(\boldsymbol{x}^{(k)}\right)\right)\boldsymbol{x}^{(k)}$$

$$= \sum_{k=1}^{N}\left(y^{(k)} - h\left(\boldsymbol{x}^{(k)}\right)\right)\boldsymbol{x}^{(k)}$$

– 误差 · 输入

- 梯度上升法

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha\sum_{k=1}^{N}\left(y^{(k)} - h\left(\boldsymbol{x}^{(k)}\right)\right)\boldsymbol{x}^{(k)}$$

随机梯度上升/下降法

- 随机选择一个训练样本

$$(\boldsymbol{x}, y)$$

- 计算梯度（分量）

$$\big(y - h_{\boldsymbol{\theta}}(\boldsymbol{x})\big)\boldsymbol{x}$$

- 更新权重

$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} + \alpha\big(y - h(\boldsymbol{x})\big)\boldsymbol{x}$$

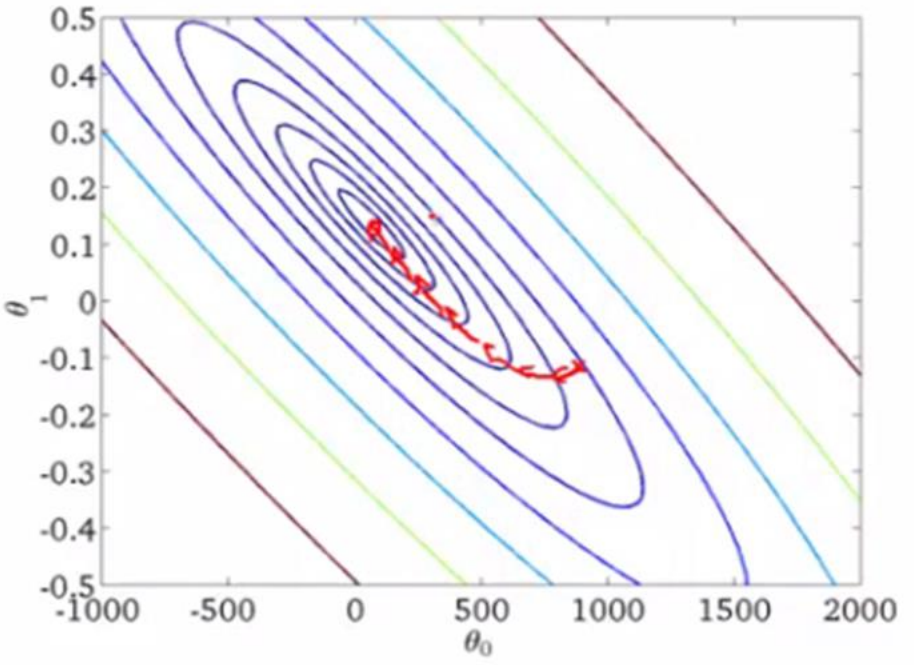
- 重复……

推广：随机选择一批训练样本

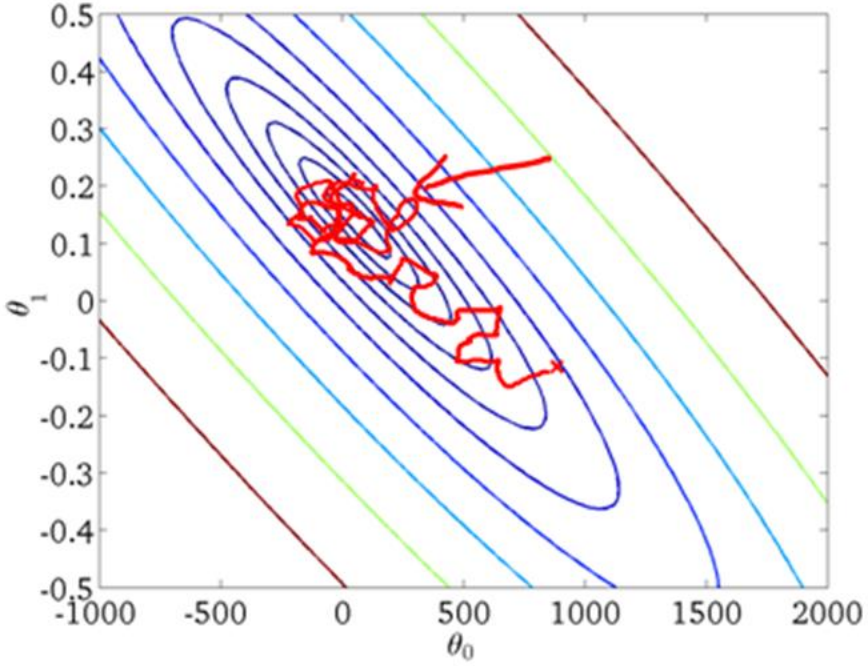
$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha \sum_{k=1}^{N_{batch}} \Big(y^{(k)} - h\big(\boldsymbol{x}^{(k)}\big)\Big)\boldsymbol{x}^{(k)}$$

随机梯度下降——适用于在线学习/增量学习

梯度下降 vs. 随机梯度下降



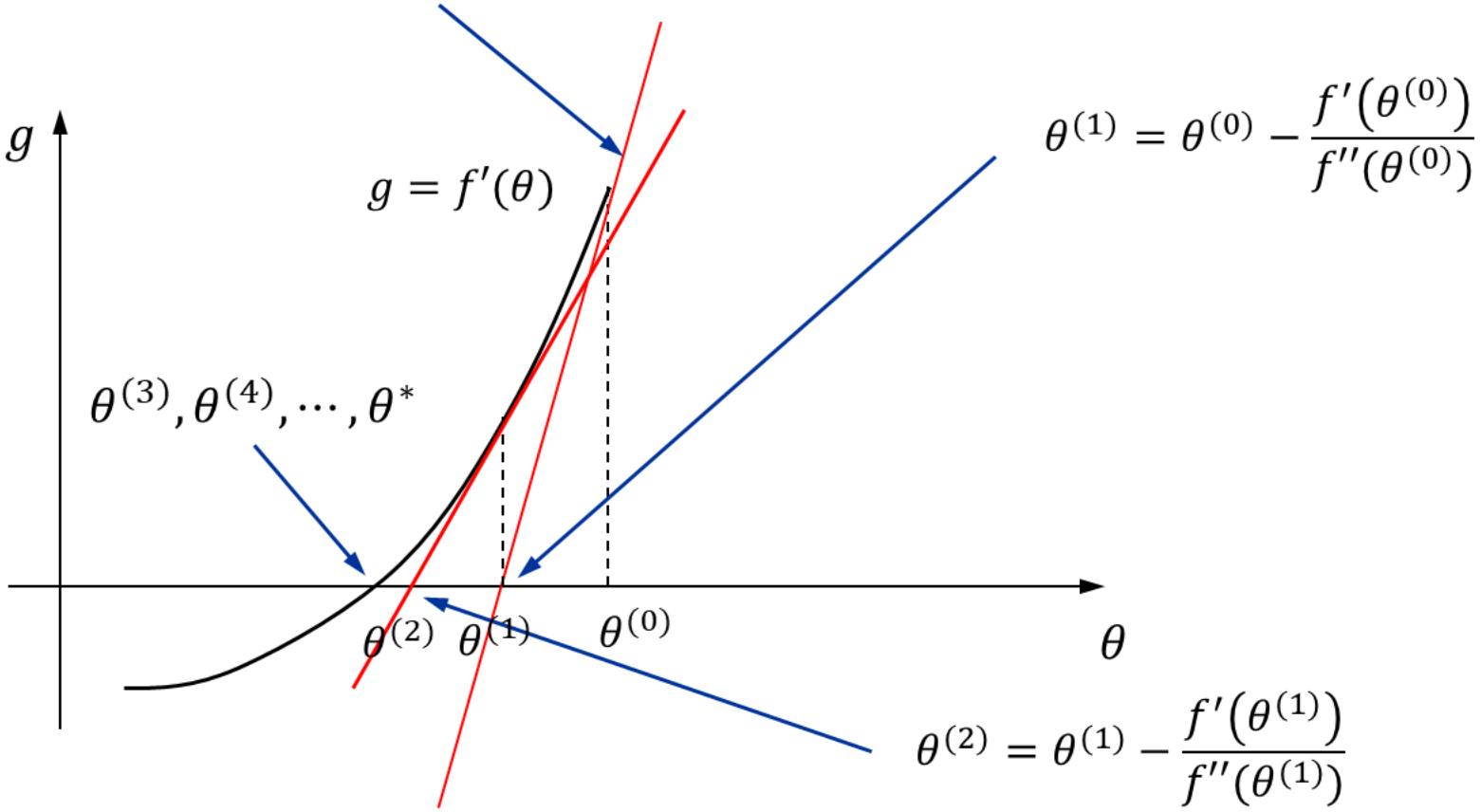
梯度下降(GD)



随机梯度下降(SGD)

牛顿法示意图

切线： $g = f'(\theta^{(0)}) + f''(\theta^{(0)})(\theta - \theta^{(0)})$



牛顿法

- 问题

$$\arg\min f(\boldsymbol{\theta}) \Leftrightarrow solve\ \nabla f(\boldsymbol{\theta}) = 0$$

- 二阶泰勒展开

$$\phi(\boldsymbol{\theta}) = f\left(\boldsymbol{\theta}^{(t)}\right) + \nabla f\left(\boldsymbol{\theta}^{(t)}\right)\left(\boldsymbol{\theta} - \boldsymbol{\theta}^{(t)}\right) + \frac{1}{2}\nabla^2 f(\boldsymbol{\theta}^{(t)})\left(\boldsymbol{\theta} - \boldsymbol{\theta}^{(t)}\right)^2 \approx f(\boldsymbol{\theta})$$

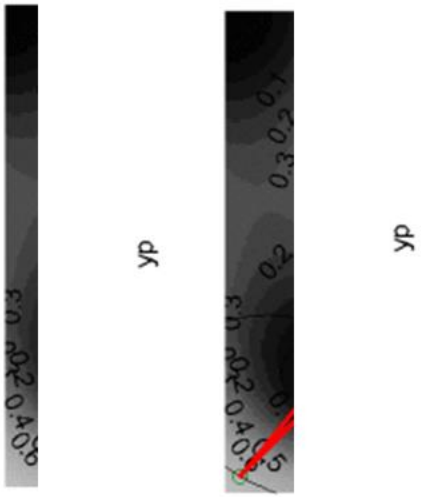
$$\nabla\phi(\boldsymbol{\theta}) = 0 \Rightarrow \boldsymbol{\theta} = \boldsymbol{\theta}^{(t)} - \nabla^2 f(\boldsymbol{\theta}^{(t)})^{-1}\nabla f(\boldsymbol{\theta}^{(t)})$$

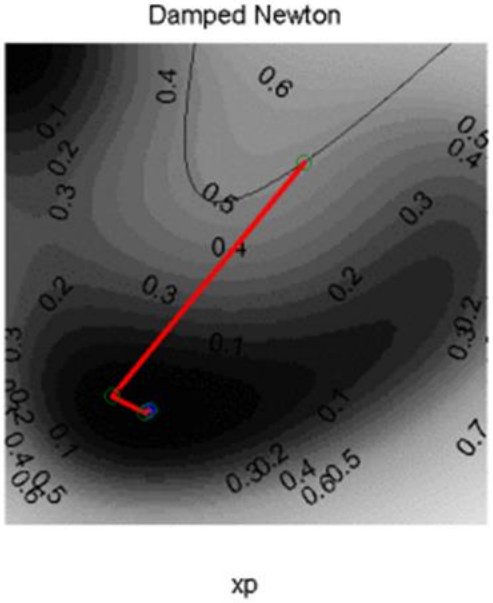
- 牛顿法（又称牛顿-拉夫逊法）

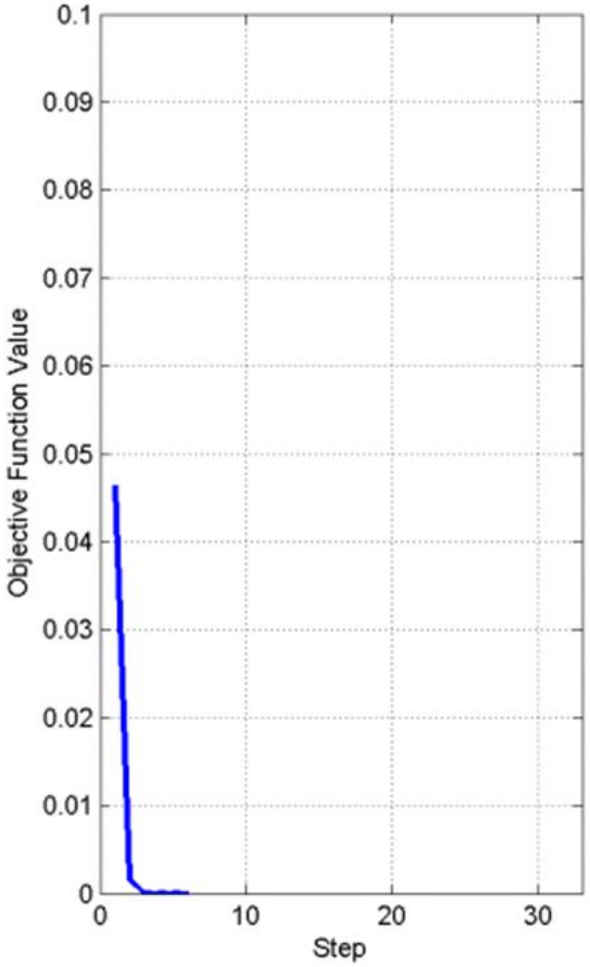
$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} - \boxed{\nabla^2 f(\boldsymbol{\theta}^{(t)})}^{-1}\nabla f(\boldsymbol{\theta}^{(t)})$$

Hessian 矩阵

梯度法 vs. 牛顿法




Damped Newton
yp
xp


Objective Function Value
Step
0.1
0.09
0.08
0.07
0.06
0.05
0.04
0.03
0.02
0.01
0
0
10
20
30

基于牛顿法的Logistic回归

- 最优化问题

$$\arg\min J(\boldsymbol{\theta}) = -\frac{1}{N}\sum_{k=1}^{N} y^{(k)} \log h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(k)}\right) + \left(1 - y^{(k)}\right)\log\left(1 - h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(k)}\right)\right)$$

- 一阶、二阶求导

$$\nabla J(\boldsymbol{\theta}) = \frac{\partial J(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \frac{1}{N}\sum_{k=1}^{N}\left(h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(k)}\right) - y^{(k)}\right)\boldsymbol{x}^{(k)}$$

$$\boldsymbol{H} = \frac{\partial^2 J(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}\partial \boldsymbol{\theta}^{\mathrm{T}}} = \frac{1}{N}\sum_{k=1}^{N} h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(k)}\right)\left(1 - h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(k)}\right)\right)\boldsymbol{x}^{(k)}\left(\boldsymbol{x}^{(k)}\right)^{\mathrm{T}}$$

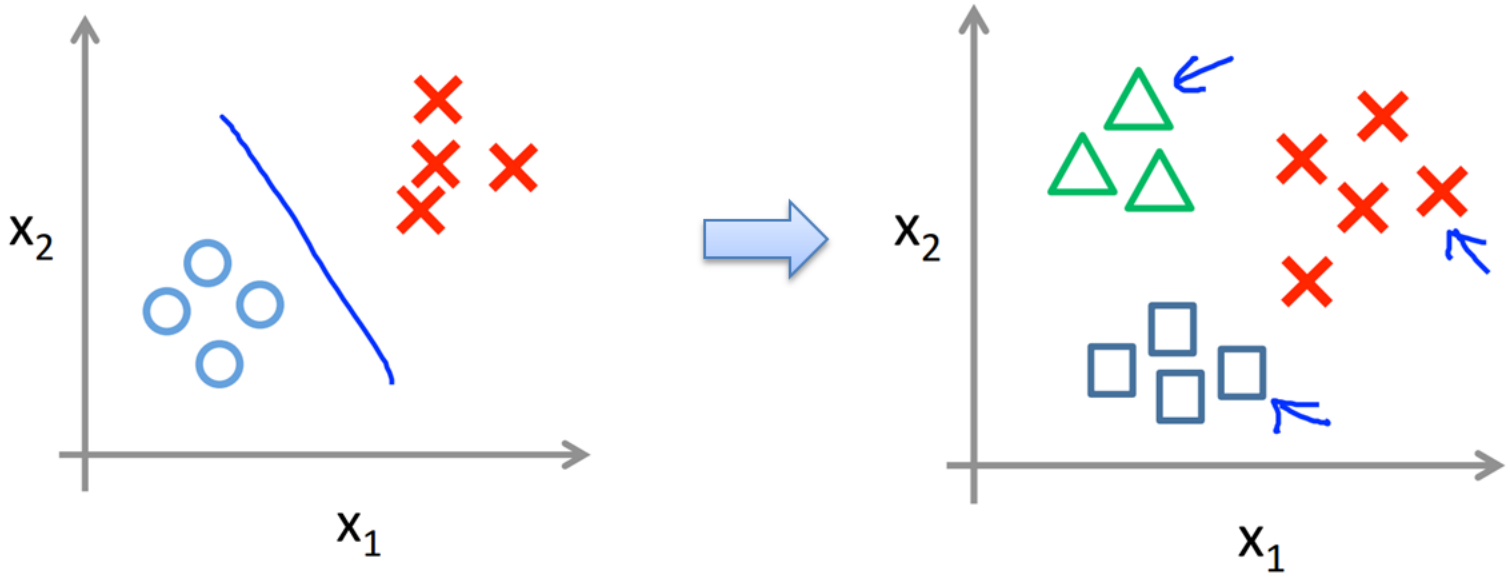
- 使用牛顿法更新权重

$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} - \boldsymbol{H}^{-1}\nabla J(\boldsymbol{\theta}^{(t)})$$

Softmax 回归

Softmax回归

- Softmax回归是一种多分类模型，也称做多类Logistic回归
- 在自然语言处理（NLP）领域，与最大熵模型等价
- 它是一种广泛使用的分类算法，常常作为深度学习模型的最后一层执行分类预测

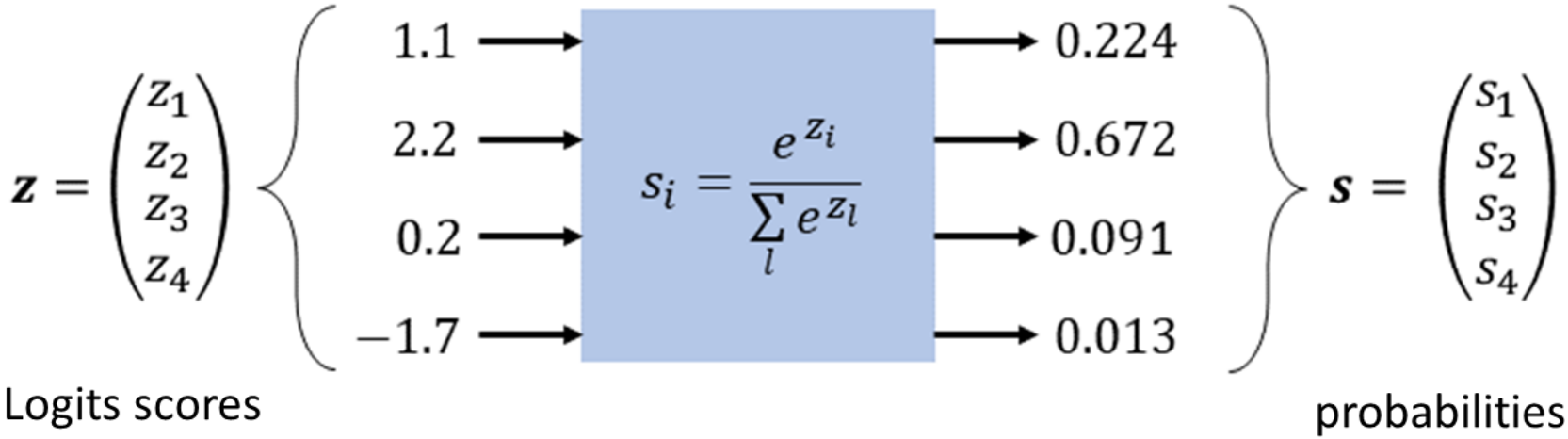


Softmax函数

- Softmax 函数（多维Logistic函数）

$$\boldsymbol{s} = \mathrm{softmax}(\boldsymbol{z}) \qquad \boldsymbol{z} \in \mathbb{R}^C \rightarrow \boldsymbol{s} \in \mathbb{R}^C$$

$$s_j = \frac{e^{z_j}}{\sum_{j'=1}^{C} e^{z_{j'}}} \quad \forall j = 1, \ldots, C$$ 归一化指数函数



Softmax函数导数性质

- Softmax函数导数性质

$$\boldsymbol{z} \in \mathbb{R}^C \rightarrow \boldsymbol{s} \in \mathbb{R}^C \quad \Rightarrow \quad J = \begin{bmatrix} \frac{\partial s_1}{\partial z_1} & \cdots & \frac{\partial s_1}{\partial z_C} \\ \vdots & \ddots & \vdots \\ \frac{\partial s_C}{\partial z_1} & \cdots & \frac{\partial s_C}{\partial z_C} \end{bmatrix}$$

Jacobian 矩阵

$$\frac{\partial s_j}{\partial z_l} = \begin{cases} s_j(1 - s_l), j = l \\ -s_j s_l, \quad j \neq l \end{cases} = s_j(I(j = l) - s_l)$$

$I(\mathrm{z})$: 指示函数

$$\text{Softmax} \cdot (I - \text{Softmax})$$

$$J_{softmax} = \begin{pmatrix} s_1 \cdot (1 - s_1) & -s_1 \cdot s_2 & -s_1 \cdot s_3 & -s_1 \cdot s_4 \\ -s_2 \cdot s_1 & s_2 \cdot (1 - s_2) & -s_2 \cdot s_3 & -s_2 \cdot s_4 \\ -s_3 \cdot s_1 & -s_3 \cdot s_2 & s_3 \cdot (1 - s_3) & -s_3 \cdot s_4 \\ -s_4 \cdot s_1 & -s_4 \cdot s_2 & -s_4 \cdot s_3 & s_4 \cdot (1 - s_4) \end{pmatrix}$$

Softmax函数求导过程

$$s_j = \frac{e^{z_j}}{\sum_{j'} e^{z_{j'}}} = \frac{e^{z_j}}{e^{z_1} + e^{z_2} + \cdots + e^{z_j} + \cdots + e^{z_c}} \quad \Rightarrow \quad \frac{\partial s_j}{\partial z_l} = ?$$

$$j \neq l: \quad \frac{\partial s_j}{\partial z_l} = e^{z_j} \cdot \frac{-1}{\left(\sum_{j'} e^{z_{j'}}\right)^2} \cdot e^{z_l} = \frac{e^{z_j}}{\sum_{j'} e^{z_{j'}}} \cdot \frac{-e^{z_l}}{\sum_{j'} e^{z_{j'}}} = -s_j s_l$$

$$j = l: \quad \frac{\partial s_j}{\partial z_l} = \frac{e^{z_j}}{\sum_{j'} e^{z_{j'}}} + e^{z_j} \cdot \frac{-1}{\left(\sum_{j'} e^{z_{j'}}\right)^2} \cdot e^{z_l} = \frac{e^{z_j}}{\sum_{j'} e^{z_{j'}}} \cdot \left(1 - \frac{e^{z_l}}{\sum_{j'} e^{z_{j'}}}\right) = s_j(1 - s_l)$$

统一形式： $$\frac{\partial s_j}{\partial z_l} = s_j(I(j = l) - s_l)$$

模型假设

- 模型假设

模型参数：$\boldsymbol{\theta}_{C\times M}$

$$p(y=j|\boldsymbol{x};\boldsymbol{\theta})=h_j(\boldsymbol{x})=\frac{e^{\boldsymbol{\theta}_j{}^{\mathrm{T}}\boldsymbol{x}}}{e^{\boldsymbol{\theta}_1{}^{\mathrm{T}}\boldsymbol{x}}+e^{\boldsymbol{\theta}_2{}^{\mathrm{T}}\boldsymbol{x}}+\cdots+e^{\boldsymbol{\theta}_C{}^{\mathrm{T}}\boldsymbol{x}}},j=1,\ldots,C$$

- 固定一个自由度

$\boldsymbol{\theta}_C=\vec{0}$ ⇒

$$p(y=j|\boldsymbol{x};\boldsymbol{\theta})=h_j(\boldsymbol{x})=\frac{e^{\boldsymbol{\theta}_j{}^{\mathrm{T}}\boldsymbol{x}}}{e^{\boldsymbol{\theta}_1{}^{\mathrm{T}}\boldsymbol{x}}+\cdots+e^{\boldsymbol{\theta}_{C-1}{}^{\mathrm{T}}\boldsymbol{x}}+1},j=1,\ldots,C-1$$

$$p(y=C|\boldsymbol{x};\boldsymbol{\theta})=h_C(\boldsymbol{x})=\frac{1}{e^{\boldsymbol{\theta}_1{}^{\mathrm{T}}\boldsymbol{x}}+\cdots+e^{\boldsymbol{\theta}_{C-1}{}^{\mathrm{T}}\boldsymbol{x}}+1}$$

$C=2$时，与Logistic回归模型等价

最大似然估计

- 对数似然函数（条件分布）

Softmax回归

$$p(y=j|\boldsymbol{x}) = h_j(\boldsymbol{x}) = \frac{e^{\boldsymbol{\theta}_j{}^{\mathrm{T}}\boldsymbol{x}}}{\sum_{j'=1}^{C} e^{\boldsymbol{\theta}_{j'}{}^{\mathrm{T}}\boldsymbol{x}}}$$

$$\begin{aligned} l(\boldsymbol{\theta}) &= \sum_{k=1}^{N} \log p(y^{(k)}|\boldsymbol{x}^{(k)};\boldsymbol{\theta}) \\ &= \sum_{k=1}^{N} \log \prod_{j=1}^{C} \left(h_j(\boldsymbol{x}^{(k)})\right)^{I(y^{(k)}=j)} \\ &= \sum_{k=1}^{N}\sum_{j=1}^{C} I\left(y^{(k)}=j\right) \log h_j(\boldsymbol{x}^{(k)}) \end{aligned}$$

Logistic回归

$$p(y=1|\boldsymbol{x}) = h(\boldsymbol{x}) = \frac{1}{1+e^{-\boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{x}}}$$

$$l(\boldsymbol{\theta}) = \sum_{k=1}^{N} y^{(k)} \log h\left(\boldsymbol{x}^{(k)}\right) + \left(1-y^{(k)}\right)\log\left(1-h\left(\boldsymbol{x}^{(k)}\right)\right)$$

梯度上升优化

- 梯度求解

$$\frac{\partial \log\, h_j(\boldsymbol{x})}{\partial \boldsymbol{\theta}_l} = \begin{cases} \big(1 - h_l(\boldsymbol{x})\big)\boldsymbol{x}, l = j \\ -h_l(\boldsymbol{x})\boldsymbol{x}, \quad l \neq j \end{cases} = \big(I(l = j) - h_l(\boldsymbol{x})\big)\boldsymbol{x}$$

- 具体推导过程

$$\frac{\partial s_j}{\partial z_l} = \begin{cases} s_j(1 - s_l), j = l \\ -s_j s_l, \quad j \neq l \end{cases} = s_j(I(j = l) - s_l)$$

$$\frac{\partial \log h_j(x)}{\partial \theta_l} = \frac{\partial \log s_j}{\partial z_l} \cdot \frac{\partial z_l}{\partial \theta_l} = \frac{1}{s_j} \cdot \frac{\partial s_j}{\partial z_l} \cdot \frac{\partial z_l}{\partial \theta_l} = (I(j = l) - s_l)\boldsymbol{x}$$

梯度上升优化

- 梯度求解

$$\frac{\partial \log h_j(\boldsymbol{x})}{\partial \boldsymbol{\theta}_l} = \begin{cases} \big(1 - h_l(\boldsymbol{x})\big)\boldsymbol{x}, l = j \\ -h_l(\boldsymbol{x})\boldsymbol{x}, \quad l \neq j \end{cases} = \big(I(l = j) - h_l(\boldsymbol{x})\big)\boldsymbol{x}$$

$$\frac{\partial \sum_{j=1}^{C} I(j = y) \log h_j(\boldsymbol{x})}{\partial \boldsymbol{\theta}_l} = \boxed{\big(I(l = y) - h_l(\boldsymbol{x})\big)\boldsymbol{x}}$$ —误差 · 输入

$$\frac{\partial l(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}_l} = \sum_{k=1}^{N} \left(I(l = y^{(k)}) - h_l\left(\boldsymbol{x}^{(k)}\right)\right)\boldsymbol{x}^{(k)}$$

- 梯度上升法

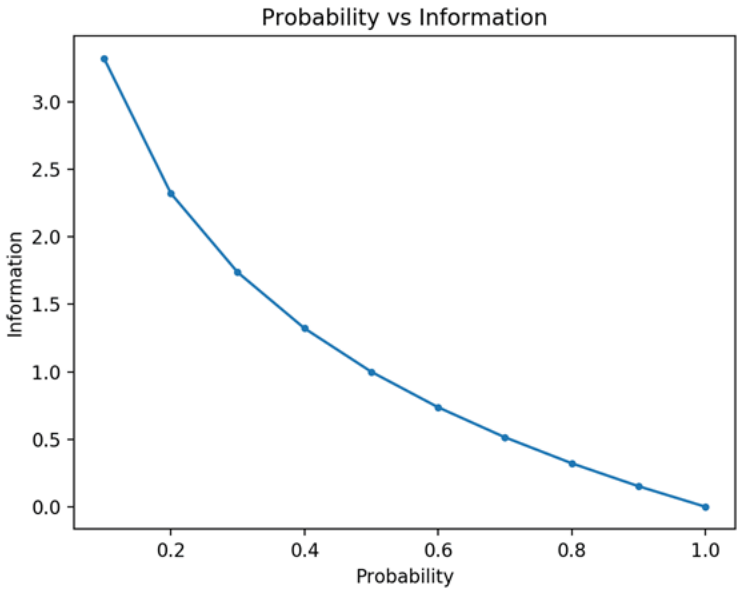
$$\boldsymbol{\theta}_j \leftarrow \boldsymbol{\theta}_j + \alpha \sum_{k=1}^{N_{batch}} \left(I\left(y^{(k)} = j\right) - h_j\left(\boldsymbol{x}^{(k)}\right)\right)\boldsymbol{x}^{(k)}$$

熵、交叉熵

- 离散随机变量的熵

$$H(X) = \sum_{x \in \mathcal{X}} P(x) I(x)$$
$$= -\sum_{x \in \mathcal{X}} P(x) \log P(x)$$

其中$I(x) = \log P(x)$称为信息量



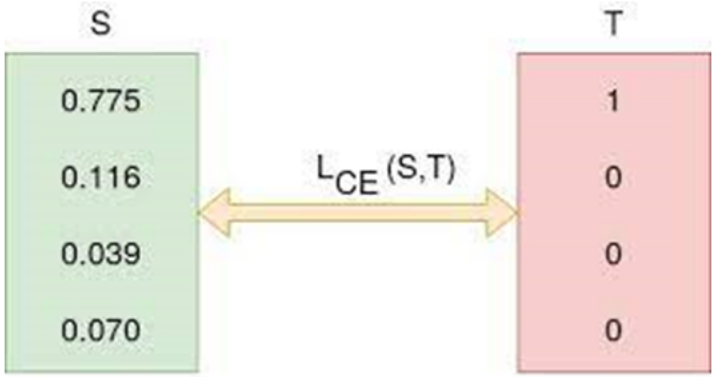
- $P(X)$、$Q(X)$为离散随机变量X的两个分布，则$Q(X)$相对于$P(X)$的交叉熵定义为：

$$H(P, Q) = -\sum_{x \in \mathcal{X}} P(x) \log Q(x)$$

交叉熵损失

- 机器学习中，针对一个训练样本$(\boldsymbol{x}, \boldsymbol{y})$，其预测类别分布$h(\boldsymbol{x})$相对于真实类别分布$\boldsymbol{y}$的交叉熵定义为：

$$H\big(\boldsymbol{y}, h(\boldsymbol{x})\big) = -\sum_{j=1}^{C} y_j \log h_j(\boldsymbol{x})$$



交叉熵损失等价于负对数似然函数

$$l(\boldsymbol{\theta}) = \sum_{j=1}^{C} I(y = j) \log h_j(\boldsymbol{x})$$

学习目标：使得训练样本上的预测分布与真实分布的差异性越小越好

分类任务的性能评估

二分类任务性能评估

针对二分类任务，在测试集上统计出以下四种情形的样本数目（混淆矩阵）：

	预测正类	预测负类
真实正类	真正例 (True Positive, TP)	假负例 (False Negative, FN)
真实负类	假正例 (False Positive, FP)	真负例 (True Negative, TN)

- 召回率（Recall）

$$R = \frac{TP}{TP + FN}$$

- 精确率（Precision）

$$P = \frac{TP}{TP + FP}$$

- F值（F Score）

$$F_1 = \frac{2PR}{P + R}$$

- 正确率（Accuracy）

$$Acc = \frac{\#correct}{\#all} = \frac{TP + TN}{TP + FP + FN + TN}$$

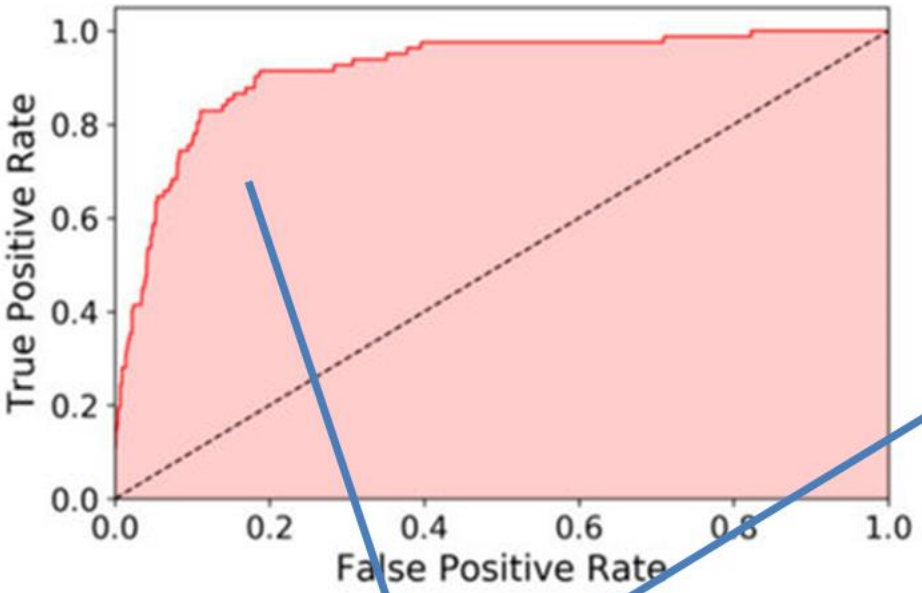
二分类任务性能评估

	真实正类(+)	真实负类(−)	全部
预测正类(+)	250	20	270
预测负类(−)	50	180	230
全部	300	200	500

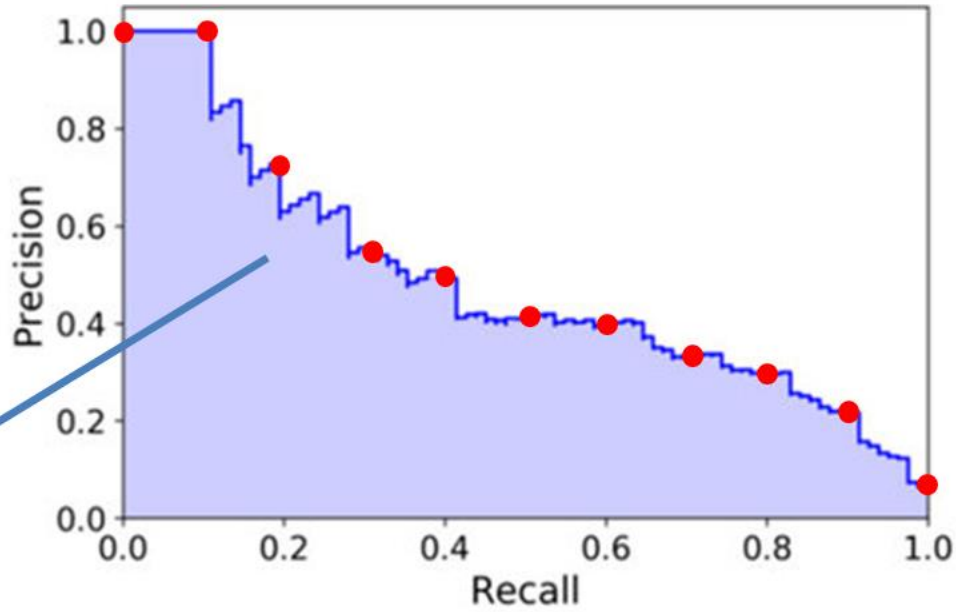
Recall	Precision	F_1	Acc

二分类任务性能评估

ROC曲线



P-R曲线



AUC（area under curve)
曲线下面积

mAP （mean average precision）
11点平均精确率

分类任务的性能评估

针对多分类任务，对每一类统计出以下四种情形的样本数目：

- 真正例 (True Positive, TP)：模型正确预测为正例（模型预测是该类，真实标签是该类）
- 真负例 (True Negative, TN): 模型正确预测为负例（模型预测非该类，真实标签非该类）
- 假正例 (False Positive, FP)：模型错误预测为正例（模型预测是该类，真实标签非该类）
- 假负例 (False Negative, FN)：模型错误预测为负例（模型预测非该类，真实标签是该类）

类别号	TP	FP	FN	TN
1	TP_1	FP_1	FN_1	TN_1
2	TP_2	FP_2	FN_2	TN_2
…	…	…	…	…
C	TP_C	FP_C	FN_C	TN_C

多分类任务性能评估

	每一类	（所有类的）宏平均	（所有类的）微平均
召回率 Recall	$R_j = \frac{TP_j}{TP_j + FN_j}$	$R_{macro} = \frac{1}{C}\sum_{j=1}^{C}\frac{TP_j}{TP_j + FN_j}$	$R_{micro} = \frac{\sum_{j=1}^{C} TP_j}{\sum_{j=1}^{C}(TP_j + FN_j)}$
精确率 Precision	$P_j = \frac{TP_j}{TP_j + FP_j}$	$P_{macro} = \frac{1}{C}\sum_{j=1}^{C}\frac{TP_j}{TP_j + FP_j}$	$P_{micro} = \frac{\sum_{j=1}^{C} TP_j}{\sum_{j=1}^{C}(TP_j + FP_j)}$
F值	$F_j = \frac{2P_jR_j}{P_j + R_j}$	$F_{macro} = \frac{1}{C}\sum_{j=1}^{C} F_j$	$F_{micro} = \frac{2P_{micro}R_{micro}}{P_{micro} + R_{micro}}$
正确率 Accuracy	$Acc = \frac{\#Correct}{\#All} = \frac{\sum_{j=1}^{C} TP_j}{TP_j + FP_j + FN_j + TN_j}$		

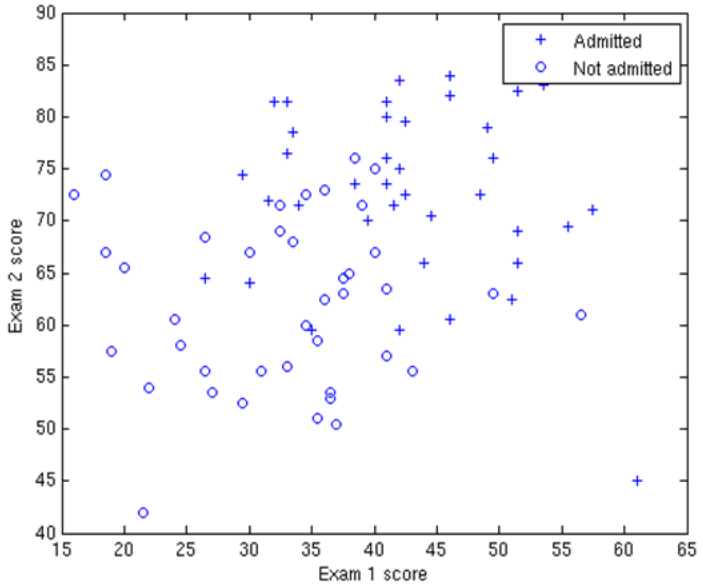
多分类任务性能评估

	真实类1	真实类2	真实类3	全部
预测类1	250	20	30	300
预测类2	50	130	50	230
预测类3	100	150	120	370
全部	400	300	200	900

	TP	FP	FN	TN	Recall	Precision	F_1	Acc
类1								
类2								
类3								
宏平均								
微平均								

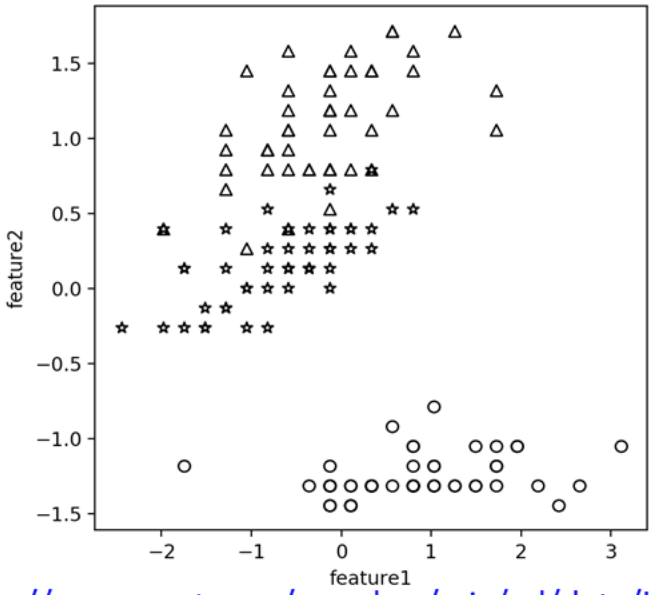
作业#2

(1) Binary Exam Dataset



http://www.nustm.cn/member/rxia/ml/data/Exam.zip

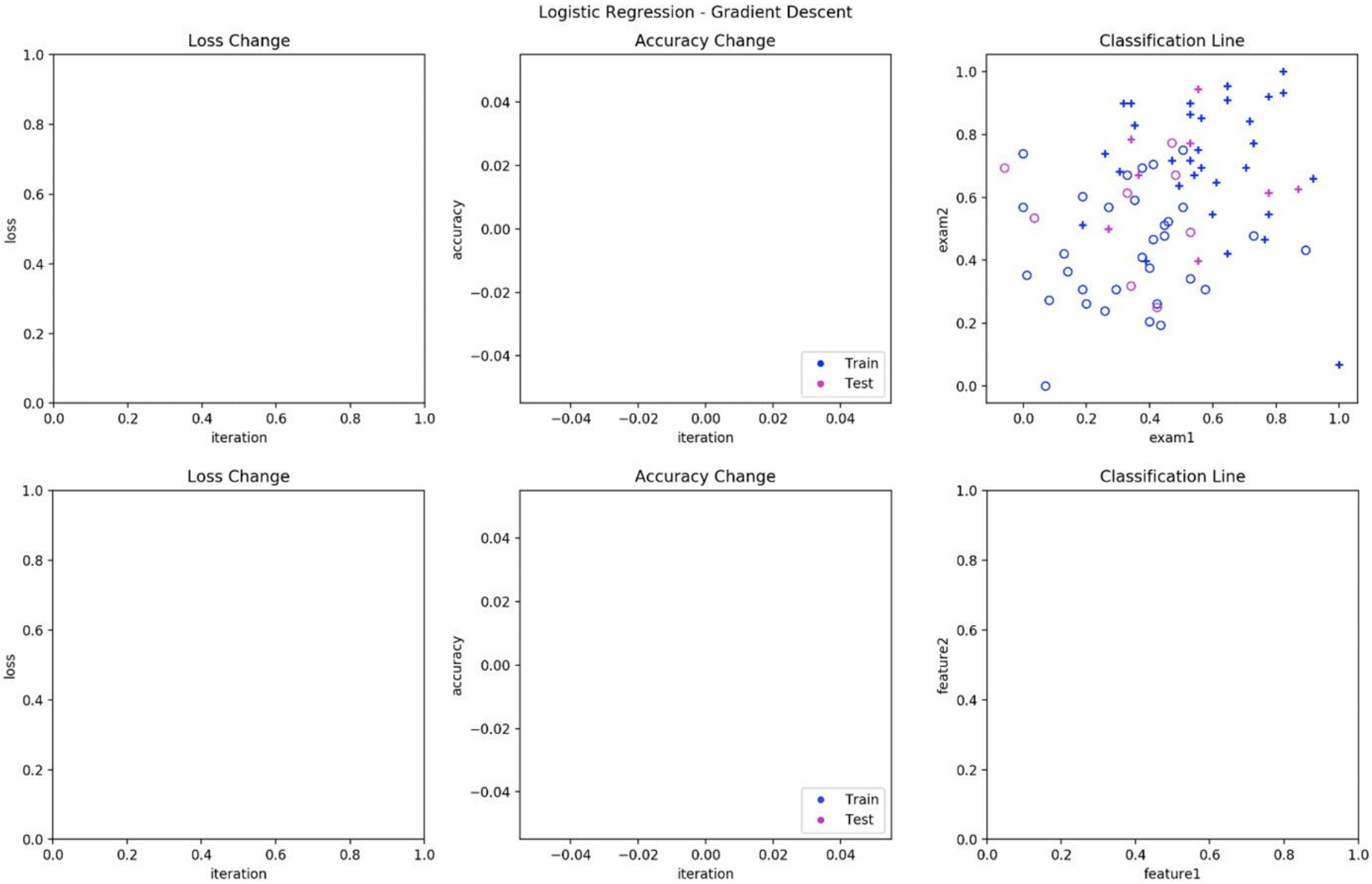
(2) Multi-class Iris Data



http://www.nustm.cn/member/rxia/ml/data/Iris.zip

- 编程实现Logistic回归，支持三种最优化方法：1) GD；2) SGD；3) 牛顿法，并用于考试录取二分类问题（数据集1），报告模型训练和测试结果，并绘制动态图。
- 编程实现Softmax回归，支持两种最优化方法： 1) GD；2) SGD，并用于考试录取二分类问题（数据集1）和 Iris三分类问题（数据集2），报告模型训练和测试结果，并绘制动态图。从模型和结果两个角度，比较Logistic回归和Softmax回归。

Demo


Logistic Regression - Gradient Descent
Loss Change
loss
iteration
Accuracy Change
accuracy
iteration
Train
Test
Classification Line
exam2
exam1
Loss Change
loss
iteration
Accuracy Change
accuracy
iteration
Train
Test
Classification Line
feature2
feature1

本讲结束 欢迎提问