机器学习-第三章 逻辑回归

教师：胡俊杰 副教授

邮箱：hujunjie@scu.edu.cn

上节课程回顾

01 线性回归

02 梯度下降

03 正则化

04 回归的评价指标

线性回归-符号约定

m 代表训练集中样本的数量

n 代表特征的维度

x 代表输入特征/输入变量

y 代表目标变量/标签

$\hat{y}$ 代表模型的预测值

(x,y) 代表训练集中的样本

(x_i,y_i) 代表第i个样本

h 代表输入空间到输出空间映射的集合，也称为假设（hypothesis）

$\hat{y}=h(x)$,代表预测的值

建筑面积 $x^{(1)}$	总层数 $x^{(2)}$	楼层 $x^{(3)}$	实用面积 $x^{(4)}$	房价 y
143.7	31	10	105	**36200**
162.2	31	8	118	**37000**
199.5	10	10	170	**42500**
96.5	31	13	74	**31200**
……	……	……	……	……

x_i是特征矩阵中的第i行，即第i 个样本，是一个**向量**

上图的：

$$x_2=\begin{bmatrix}162.2\\31\\8\\118\end{bmatrix} \qquad y_2=37000$$

$x_i^{(j)}$代表特征矩阵中第 i 行的第 j 个特征

上图的$x_2^{(2)}=31, x_3^{(2)}=10$

注：1、有时表述简便，右上角的括号可省略

2、$h(x)$也可写成$f(x)$

线性回归-算法流程

$$\hat{y} = h(x) = w_0 + w_1 x^{(1)} + w_2 x^{(2)} + \ldots + w_n x^{(n)}$$

$\hat{y}$: 模型的预测　　　　y: 真实标签

距离/误差

- **代价函数/损失函数**(Cost Function/Loss Function)度量样本集的平均误差。
  - 常用$L(y, h(x))$或$J(w)$表示
  - 常用的代价函数包括0-1损失函数、平方和损失函数、绝对损失函数

损失函数/代价函数	数学表达式
0-1损失函数	$L(y, h(x)) = \begin{cases} 1, y! = h(x) \\ 0, y = h(x) \end{cases}$
平方和损失函数	$L(y, h(x)) = (h(x) - y)^2$
绝对损失函数	$L(y, h(x)) = \lvert h(x) - y \rvert$

线性回归-最小二乘法(LSM, Least Squares Method)

样本i：$\hat{y}_i = h(x_i) = w_0 1 + w_1 x_i^{(1)} + w_2 x_i^{(2)} + \ldots + w_n x_i^{(n)} = \sum_{k=0}^{n} w_k x_i^{(k)}$ ，其中$x_i^{(0)} = 1$

$$J(w) = \frac{1}{2}\sum_{i=1}^{m}(h(x_i) - y_i)^2 = \frac{1}{2}\sum_{i=1}^{m}((\sum_{k=0}^{n} w_k x_i^{(k)}) - y_i)^2$$

m个样本　n维特征

$$J(w) = \frac{1}{2}\boxed{\sum_{i=1}^{m}}((\boxed{\sum_{k=0}^{n}} w_k x_i^{(k)}) - y_i)^2$$

线性回归-最小二乘法(LSM, Least Squares Method)

$$\hat{y}_1 = h(x_1) = w_0 1 + w_1 x_1^{(1)} + w_2 x_1^{(2)} + \ldots + w_n x_1^{(n)}$$

$$\hat{y}_2 = h(x_2) = w_0 1 + w_1 x_2^{(1)} + w_2 x_2^{(2)} + \ldots + w_n x_2^{(n)}$$

$$\vdots$$

$$\hat{y}_m = h(x_m) = w_0 1 + w_1 x_\mathrm{m}^{(1)} + w_2 x_\mathrm{m}^{(2)} + \ldots + w_n x_m^{(n)}$$

$\Big\downarrow$ m个线性方程

$$\underset{\hat{Y}}{\begin{bmatrix} \hat{y}_1 \\ \hat{y}_2 \\ \cdots \\ \hat{y}_m \end{bmatrix}} = \underset{X}{\begin{bmatrix} 1 & x_1^{(1)} & x_1^{(2)} & x_1^{(3)} & \cdots & x_1^{(n)} \\ 1 & x_2^{(1)} & x_2^{(2)} & x_2^{(3)} & \cdots & x_2^{(n)} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_m^{(1)} & x_m^{(2)} & x_m^{(3)} & \cdots & x_m^{(n)} \end{bmatrix}} \underset{w}{\begin{bmatrix} w_0 \\ w_1 \\ \cdots \\ w_n \end{bmatrix}} \xrightarrow{\text{向量形式}} \hat{Y} = Xw$$

线性回归

模型假设（Hypothesis）： $h(x_i) = w_0 + w_1 x_i^{(1)} + w_2 x_i^{(2)} + \ldots + w_n x_i^{(n)}$

可学习参数: $(w_0, w_1, w_2, \ldots, w_n)$

代价函数: $J(w) = \frac{1}{2}\sum_{i=1}^{m}(h(x_i) - y_i)^2$

目标: 最小化$J(w)$

线性回归-最小二乘法(LSM, Least Squares Method)

$$J(w)=\frac{1}{2}\left(\hat{Y}-Y\right)^2=\frac{1}{2}(Xw-Y)^2=\frac{1}{2}(Xw-Y)^T(Xw-Y)$$

为使得$J(w)$最小，计算$\frac{\partial J(w)}{\partial w}$， 令$\frac{\partial J(w)}{\partial w}=0$

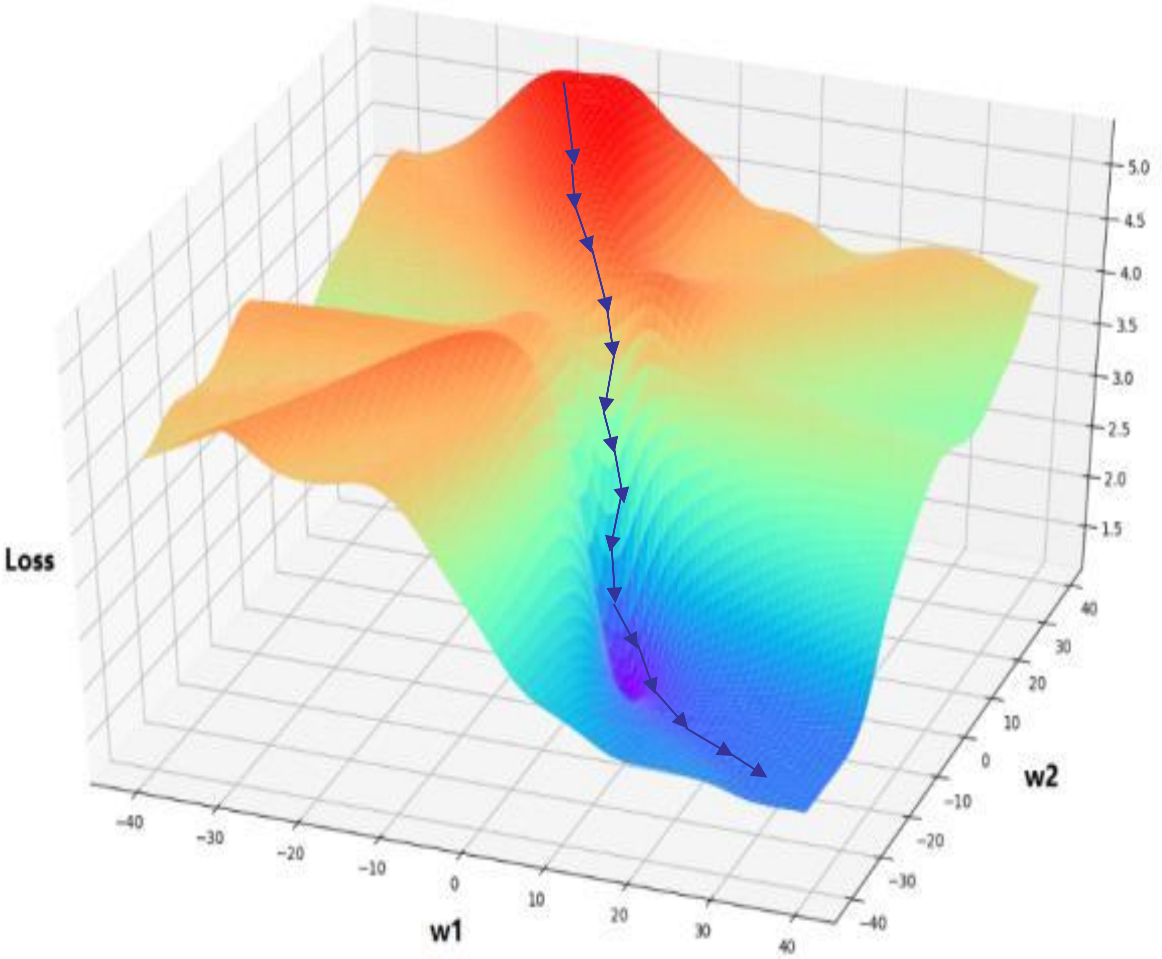
计算得到$w=(X^TX)^{-1}X^TY$

当矩阵X很大时，即训练样本量或样本特征维度很大，$(X^TX)^{-1}$计算量很大，矩阵求逆的时间复杂度为$O(n^3)$

$$\underset{\hat{Y}}{\begin{bmatrix}\hat{y}_1\\ \hat{y}_2\\ \cdots\\ \hat{y}_m\end{bmatrix}}=\underset{X}{\begin{bmatrix}1 & x_1^{(1)} & x_1^{(2)} & x_1^{(3)} & \cdots & x_1^{(n)}\\ 1 & x_2^{(1)} & x_2^{(2)} & x_2^{(3)} & \cdots & x_2^{(n)}\\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots\\ 1 & x_m^{(1)} & x_m^{(2)} & x_m^{(3)} & \cdots & x_m^{(n)}\end{bmatrix}}\underset{w}{\begin{bmatrix}w_0\\ w_1\\ \cdots\\ w_n\end{bmatrix}}$$

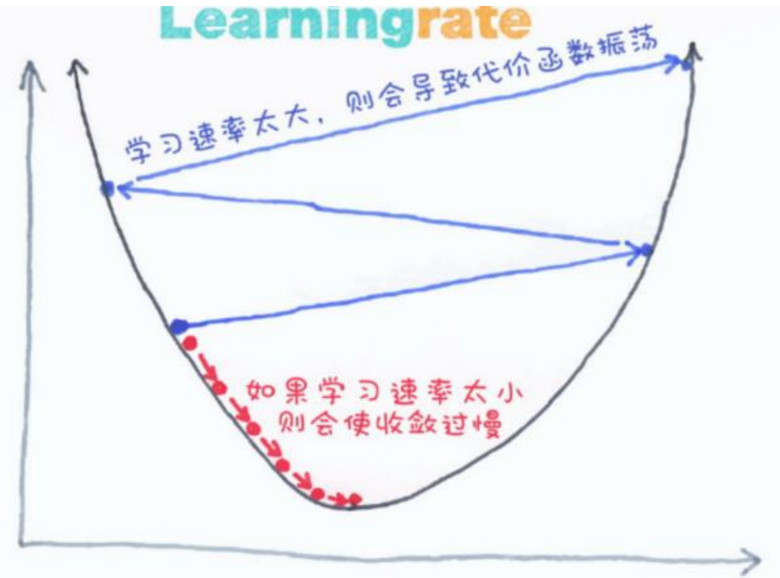
是否有更简单的优化$J(w)$的方法?

梯度下降



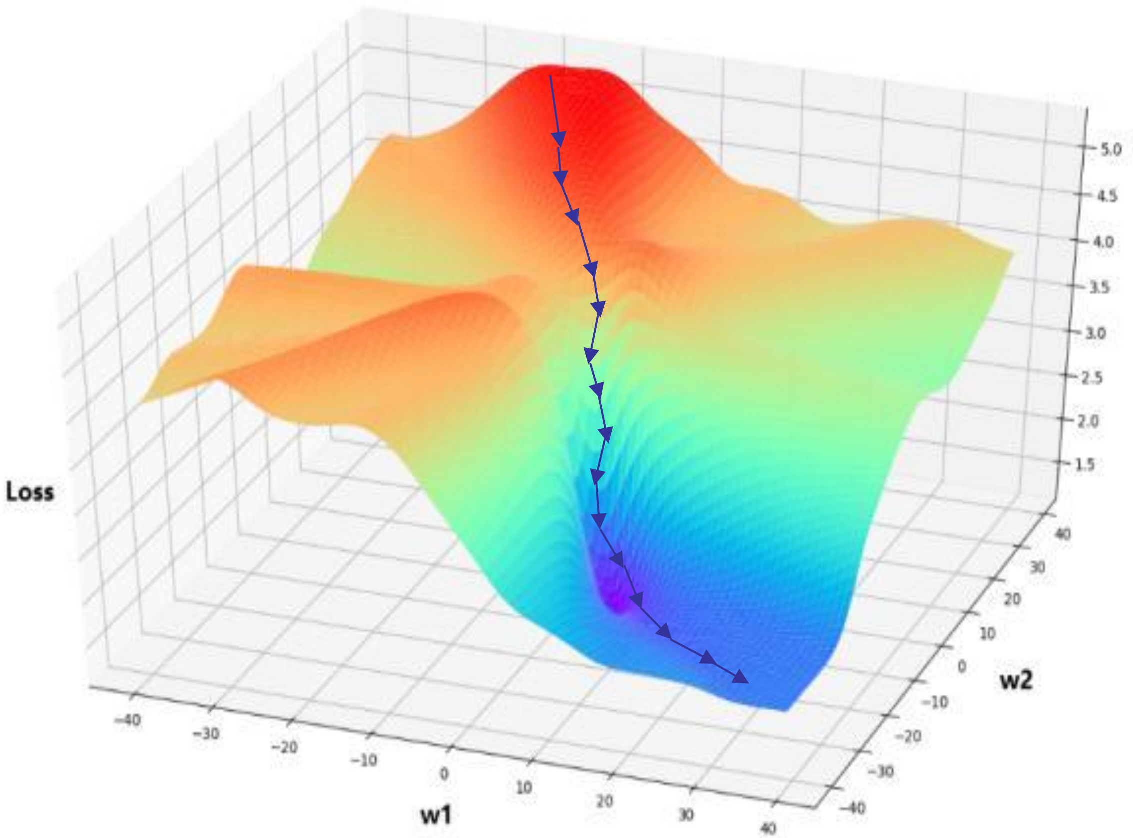
- 沿着梯度的方向，函数下降/上升最快，因此梯度下降有时也称为最速下降法
- 学习率(Learning rate) α，也称为步长(Step)

$Loss$



w

梯度下降



梯度下降算法的一般表达形式：

$$w := w - \alpha \frac{\partial J(w)}{\partial w} \text{ 或 } w \leftarrow w - \alpha \frac{\partial J(w)}{\partial w}$$

- w: 可学习的参数
- α: 学习率/步长
- $\frac{\partial J(w)}{\partial w}$: 目标函数对$w$的梯度

$$\frac{\partial J(w)}{\partial w}$$

梯度下降

线性回归模型：$h(x_i) = w_0 x_i^{(0)} + w_1 x_i^{(1)} + w_2 x_i^{(2)} + \ldots + w_n x_i^{(n)}$

对于单个训练样本i：$J(w) = \frac{1}{2}(h(x_i) - y_i)^2$

$$\frac{\partial J(w)}{\partial w_j} = \frac{\partial}{\partial w_j}\frac{1}{2}(h(x_i) - y_i)^2 = 2 \cdot \frac{1}{2}(h(x_i) - y_i) \cdot \frac{\partial}{\partial w_j}(h(x_i) - y_i)$$

$$= (h(x_i) - y_i) \cdot \frac{\partial}{\partial w_j}(\sum_{k=0}^{n}(w_k x_i^{(k)}) - y_i)$$

$$= (h(x_i) - y_i) \cdot x_i^{(j)}$$

随机梯度下降算法

第1步. 准备训练数据集 $D=\{(x,y)\}$

第2步. 随机初始化$(w_0,w_1,w_2,\dots,w_n)$, 设置学习率 α.

第3步. 从 D中选择b个训练样本，$(x_i,y_i)\in D^b$

$$\frac{\partial J(w)}{\partial w_j}\leftarrow\frac{\partial J(w)}{\partial w_j}+(h(x_i)-y_i){x_i}^{(j)}$$ //计算并累积各样本的梯度

第4步. 更新参数

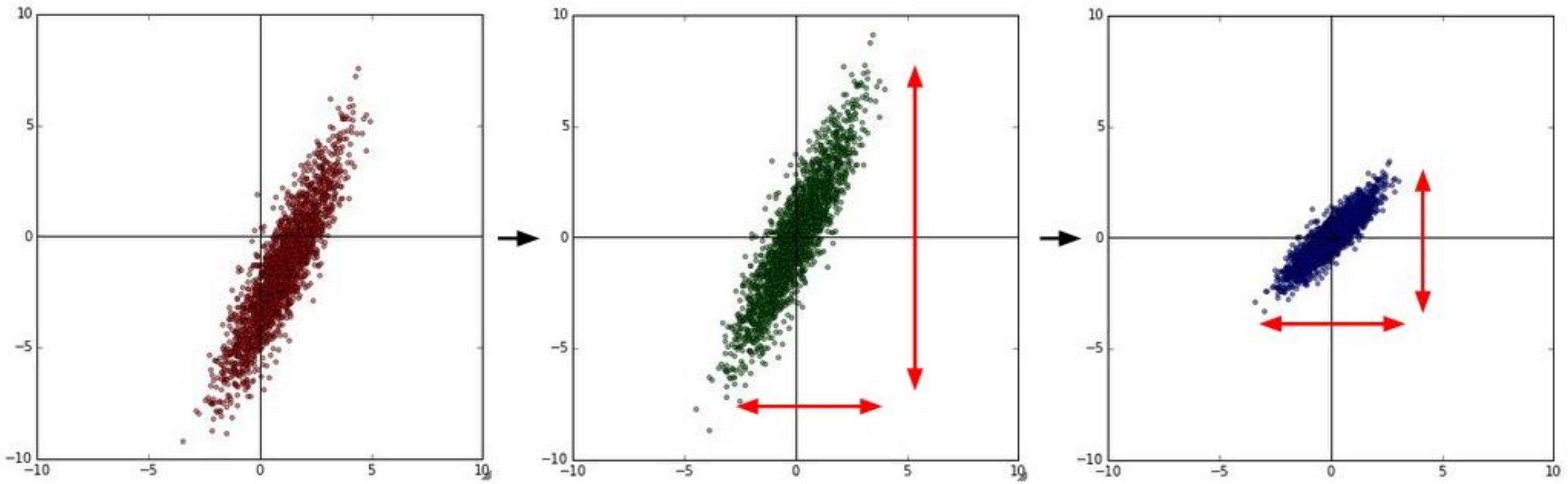
$$w_j:=w_j-\alpha\frac{1}{b}\frac{\partial J(w)}{\partial w_j}$$

第5步. 继续第3步，直到模型收敛.

验证集的代价函数小于某一阈值ϵ

$$J(w)=\frac{1}{2}\sum_{i=1}^{m}(h(x_i)-y_i)^2$$

数据归一化/标准化



原始的2维数据x

$$\hat{x} := x - \mu$$

均值为0，以原点为中心

$$x := \frac{\hat{x}}{\sigma}$$

拉伸至标准差为1

$$\mu = \frac{1}{m}\sum_{i=1}^{m} x_i$$

$$\sigma^2 = \frac{1}{m}\sum_{i=1}^{m} (x_i - \mu)^2$$

数据归一化/标准化

Z-Score标准化

$$x^* = \frac{x-\mu}{\sigma}$$

$$\mu = \frac{1}{m}\sum_{i=1}^{m} x_i$$

$$\sigma^2 = \frac{1}{m}\sum_{i=1}^{m} (x_i - \mu)^2$$

处理后的数据服从均值为0，方差为1的高斯分布

最大 - 最小标准化

$$x^* = \frac{x - x_{\min}}{x_{\max} - x_{\min}}$$

将数据映射到[0,1]区间

正则化（Regularization）

范数（Norm）

向量空间中的向量具有大小，使用范数度量向量的大小

L_p范数： $L_p = \|\boldsymbol{w}\|_p = \sqrt[p]{\sum_{i=1}^{n} w_i^p}$ 其中$\boldsymbol{w} = (w_1, w_2, \dots, w_n)$

L_0范数： 向量$\boldsymbol{w}$中的非0元素个数

L_1范数： $L_1 = \|\boldsymbol{w}\|_1 = \sum_{i=1}^{n} |w_i|$ L_2范数： $L_2 = \|\boldsymbol{w}\|_2 = \sqrt{\sum_{i=1}^{n} w_i^2}$

正则化（Regularization）

$$h(x) = w_0 + w_1 x^{(1)} + w_2 x^{(2)} + \ldots + w_n x^{(n)}$$

$$J(w) = \frac{1}{2}\sum_{i=1}^{m}\left(h(x_i) - y_i\right)^2$$

奥卡姆剃刀定律：在所有符合实验数据的模型中，简单的模型优于复杂模型。

$$\min_w J(w)$$
$$s.t. \|w\|_0 \le C$$

$s.t.$: subject to
C: 常量

- 该约束项的含义是增加模型中的零元素个数，以获得更简单的模型
- 由于该问题是一个NP问题，不易求解，因此稍微放宽约束条件，用L_1范数代替

$$\min_w J(w)$$
$$s.t. \|w\|_1 \le C$$

正则化（Regularization）

$$\boxed{\begin{aligned}&\min_{w} J(w)\\ &s.t. \|w\|_1 \le C\end{aligned}}$$

拉格朗日乘子法

正则化系数

$$J(w)=\frac{1}{2}\sum_{i=1}^{m}(h(x_i)-y_i)^2+\boxed{\lambda}\left(\sum_{j=1}^{n}|w_j|-C\right)$$

$$J(w)=\frac{1}{2}\sum_{i=1}^{m}(h(x_i)-y_i)^2$$

$$J(w)=\frac{1}{2}\sum_{i=1}^{m}(h(x_i)-y_i)^2+\lambda\sum_{j=1}^{n}|w_j|$$

正则化（Regularization）

正则化系数

L_1正则化：$J(w)=\frac{1}{2}\sum_{i=1}^{m}(h(x_i)-y_i)^2+\boxed{\lambda}\sum_{j=1}^{n}|w_j|$，Lasso regression（Lasso回归）

L_2正则化：$J(w)=\frac{1}{2}\sum_{i=1}^{m}(h(x_i)-y_i)^2+\lambda\sum_{j=1}^{n}w_j^2$，Ridge regression （岭回归）

Elastic Net：$J(w)=\frac{1}{2}\sum_{i=1}^{m}(h(x_i)-y_i)^2+\lambda(\boxed{\rho}\cdot\sum_{j=1}^{n}|w_j|+(1-\rho)\cdot\sum_{j=1}^{n}w_j^2)$

(弹性网络)

比例系数

其中：

- λ为正则化系数，调整正则化项与训练误差的比例，λ>0。
- 1≥ρ≥0为比例系数，调整L_1正则化与L_2正则化的比例。

1.分类问题

分类问题

监督学习的最主要类型之一

标签离散

- ✓ 分类（Classification）
  - ✓ 身高1.85m，体重100kg的男人穿什么尺码的T恤?
  - ✓ 根据肿瘤的体积、患者的年龄来判断良性或恶性?
  - ✓ 根据用户的年龄、职业、存款数量来判断信用卡是否会违约?

输入变量可以是离散的，也可以是连续的

分类问题

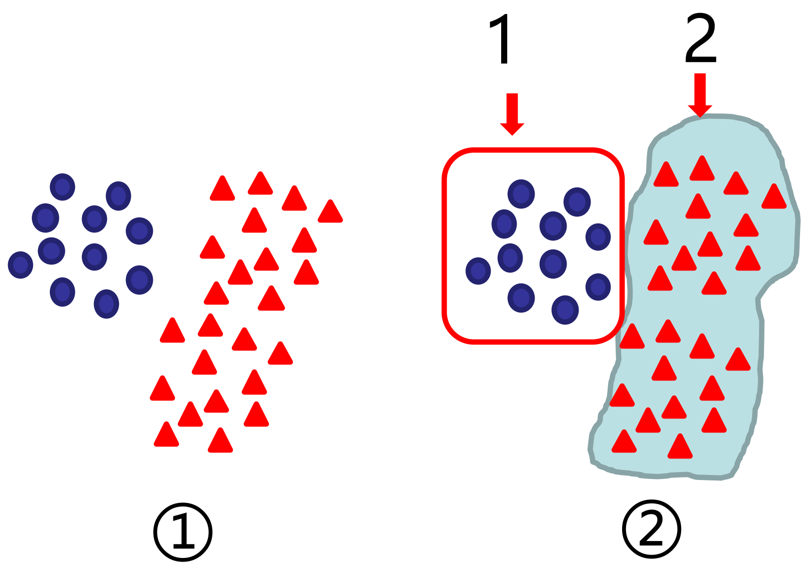
二分类

我们将蓝色圆形数据定义为类型1

，其余数据为类型2；

只需要分类1次

步骤：①->②



二分类

分类问题

多分类

我们先定义其中一类为类型1（正类）

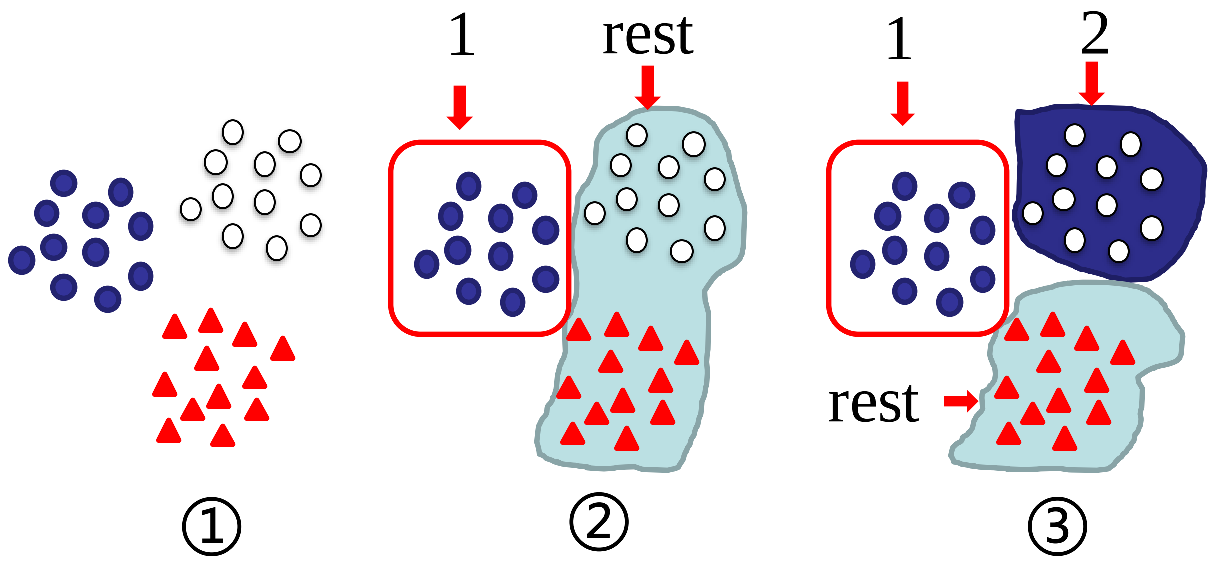
，其余数据为负类；

接下来去掉类型1数据，剩余部分再次

进行二分类，分成类型2和负类；

如果有n类，那就需要分类n-1次

步骤：①->②->③->……



One-vs-All (One-vs-Rest)

一对多 (一对余)

2.Sigmoid函数

2.Sigmoid函数

$$h(x) = w_0 + w_1 x^{(1)} + w_2 x^{(2)} + \ldots + w_n x^{(n)}$$

可以设$x^{(0)} = 1$，则

标量形式： $h(x) = w_0 x^{(0)} + w_1 x^{(1)} + w_2 x^{(2)} + \ldots + w_n x^{(n)} = \sum_{k=0}^{n} w_k x^{(k)}$

向量形式： $h(x) = w^T x$

$$z = h(x) = w^T x \qquad z \in (-\infty, +\infty)$$

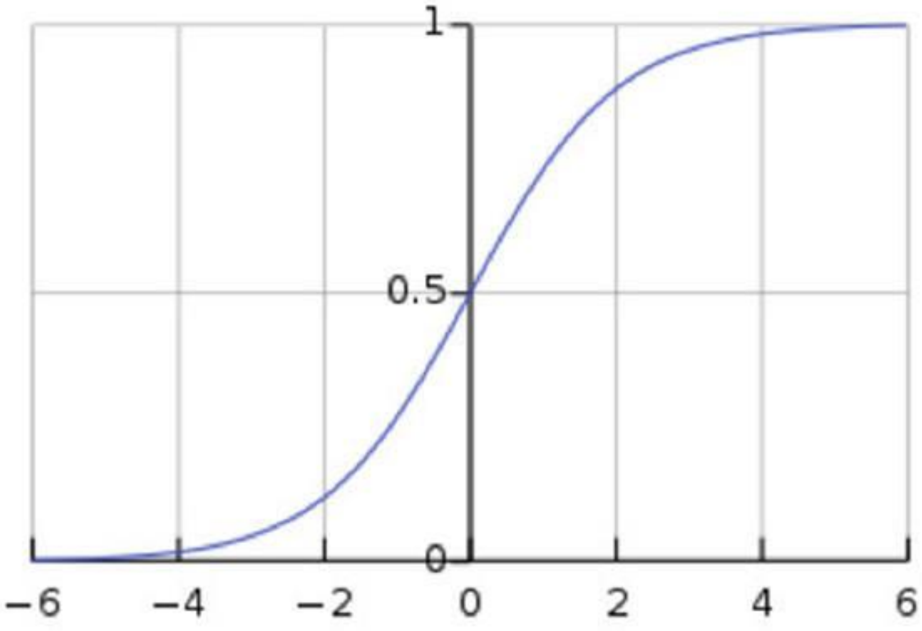
- 我们希望输出的值代表事件发生的概率，即$z \in [0,1]$
- $\sigma(z)$，对输出z作用一个函数，将z压缩至[0,1]区间

2.Sigmoid函数

Sigmoid 函数

$\sigma(z)$代表一个常用连续S形函数（Sigmoid function）或逻辑函数（ Logistic function ）

$$\sigma(z) = g(z) = \frac{1}{1+e^{-z}} \qquad z=w^T x$$



当$\sigma(z)$大于等于0.5时，预测为1

当$\sigma(z)$小于0.5时，预测为0

- 模型预测的类别不仅取决于模型的输出值，也依赖于设置的阈值

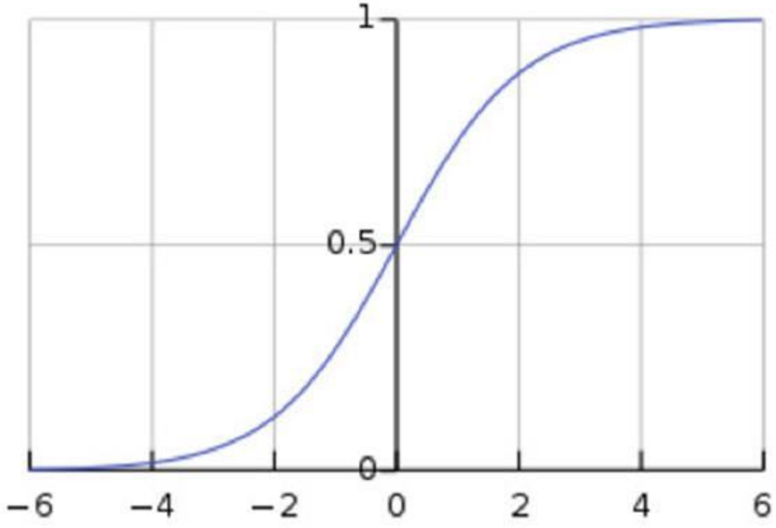
2.Sigmoid函数

在二分类模型中，事件的几率odds：事件发生与事件不发生的概率之比为$\frac{p}{1-p}$，

其中p为随机事件发生的概率，p的范围为[0,1]

取对数得到：$\log\frac{p}{1-p}$，$\log\frac{p}{1-p}=w^Tx=z$

求解得到：$p=\frac{1}{1+e^{-w^Tx}}=\frac{1}{1+e^{-z}}$



- 事件发生的概率: $p=\frac{1}{1+e^{-w^Tx}}=\frac{e^{w^Tx}}{1+e^{w^Tx}}$
- 事件不发生的概率: $1-p=\frac{e^{-w^Tx}}{1+e^{-w^Tx}}=\frac{1}{1+e^{w^Tx}}$

2.Sigmoid函数

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

将结果用$g(z)$表示

2.Sigmoid函数

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

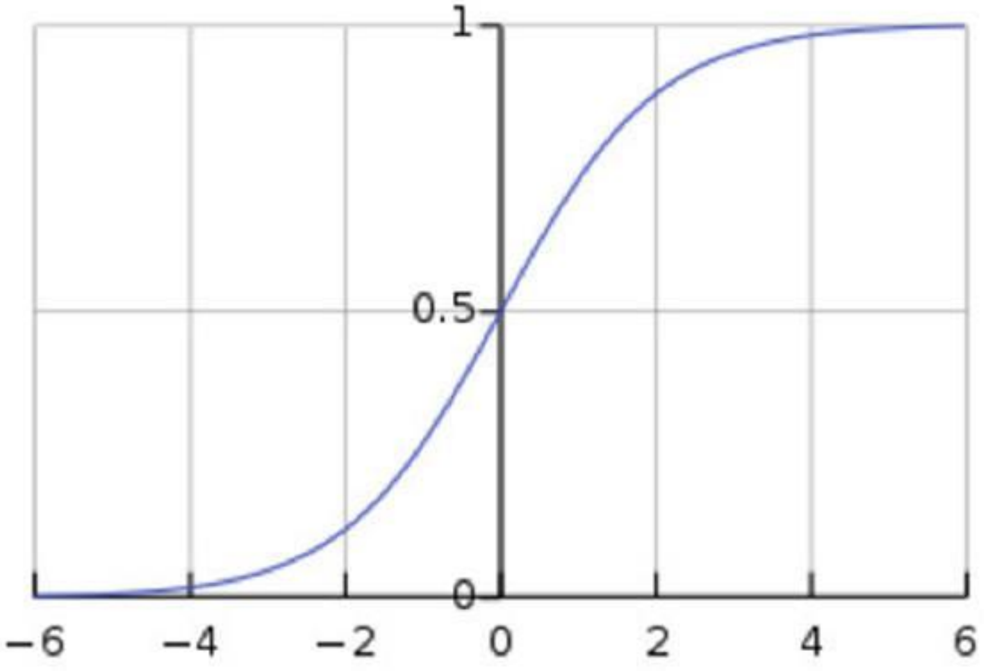
$$g'(z) = (\frac{1}{1+e^{-z}})'$$

$$= \frac{e^{-z}}{(1+e^{-z})^2}$$

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

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

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



$g(z)$

3.逻辑回归求解

01 分类问题

02 Sigmoid函数

03 逻辑回归求解

04 逻辑回归代码实现

3.逻辑回归求解

假设一个二分类模型：

$$P(y=1|x;w)=h(x)$$

$$P(y=0|x;w)=1-h(x)$$

则：

$$P(y|x;w)=(h(x))^y(1-h(x))^{1-y}$$

逻辑回归模型的假设是： $h(x)=g(w^Tx)=g(z)$，其中$z=w^Tx$

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

- 虽然称为逻辑回归，但用于解决分类问题

3.逻辑回归求解

$\hat{y}$ 表示模型的预测值$h(x)$

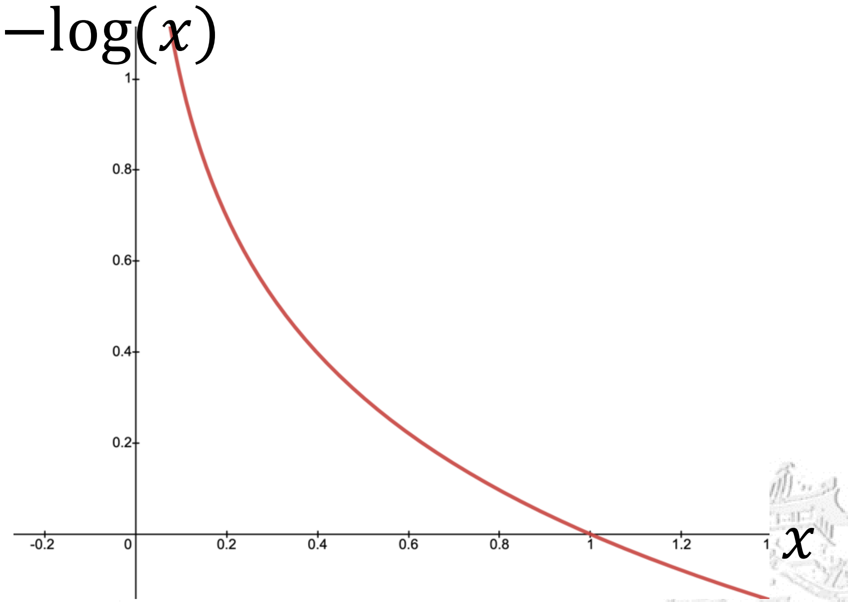
y 表示真实值(标签)

代价函数

$$J(w)=\frac{1}{m}\sum_{i=1}^{m}L(h(x_i),y_i)=\frac{1}{m}\sum_{i=1}^{m}L(\hat{y}_i,y_i)$$

$$h(x_i)=\frac{1}{1+e^{-w^Tx_i}},h(x_i)\in(0,1)$$

$$L(h(x_i),y_i)=\begin{cases}-\log(h(x_i)),if\ y=1\\-\log\left(1-h(x_i)\right),if\ y=0\end{cases}$$



- y只能等于0或1
- 当$x\in(0,1)$区间时，$-\log(x)>0$且单调递减
- 通过最小化$L(h(x_i),y_i)$，使得$h(x_i)\rightarrow y_i$

3.逻辑回归求解

代价函数

$\hat{y}$ 表示模型的预测值$h(x)$

y 表示真实值(标签)

$$L(h(x_i),y_i)=\begin{cases}-\log(h(x_i),if\ y_i=1\\ -\log\big(1-h(x_i)\big),if\ y_i=0\end{cases}$$

$$L(h(x_i),y_i)=-y_i\log(h(x_i)-(1-y_{\mathrm{i}})\log\big(1-h(x_i)\big)$$

$$J(w)=\frac{1}{m}\sum_{i=1}^{m}L(h(x_i),y_i)=\frac{1}{m}\sum_{i=1}^{m}(-y_{\mathrm{i}}\log(h(x_i))-(1-y_i)\log(1-h(x_i)))$$

2.似然函数（Likelihood function）

$L(\theta|x)$: 给定x时，关于参数θ的似然函数（Likelihood function）。代表给定数据x，参数θ生成该数据的可能性

极大似然估计（Maximum Likelihood Estimation, MLE）

- $\hat{\theta} = \operatorname{argmax} L(\theta|x_1, x_2, \ldots, x_n)$
- 在θ的所有可能取值中，寻找到$\hat{\theta}$使得似然函数最大，$\hat{\theta}$即称为θ的极大似然估计

3.逻辑回归求解

求解过程：

似然函数为： $L(h(x_i),y_i)=\prod_{i=1}^{m} P\,(y_i|x_i;w)=\prod_{i=1}^{m}(\,h(x_i))^{y_i}(1-h(x_i))^{1-y_i}$

似然函数两边取对数，则累乘号变成了累加号：

$$\log L(h(x_i),y_i)\rightarrow\sum_{i=1}^{m}(\,y_i\log(h(x_i))+(1-y_i)\log(1-h(x_i)))$$

最大化

代价函数为：

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

最小化

3.逻辑回归求解

梯度下降求解过程：

$$w_j := w_j - \alpha \frac{\partial J(w)}{\partial w_j}$$

$$J(w) = -\frac{1}{m}\sum_{i=1}^{m}\left(y_{\mathrm{i}}\log(h(x_i)) + (1 - y_{\mathrm{i}})\log(1 - h(x_i))\right)$$

$$\boxed{\frac{\partial J(w)}{\partial w_j} = \frac{1}{m}\sum_{i=1}^{m}(h(x_i) - y_{\mathrm{i}})x_i^{(j)}}$$

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

$$w_j := w_j - \alpha \frac{1}{m}\sum_{i=1}^{m}(h(x_i) - y_{\mathrm{i}})x_i^{(j)}$$

3.逻辑回归求解

求解过程： $\frac{\partial J(w)}{\partial w_j} = \frac{1}{m}\sum_{i=1}^{m}(h(x_i) - y_i)x_i^{(j)}$：

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

$$\Downarrow \quad h(x_i) = \frac{1}{1+e^{-w^T x_i}}$$

$$\begin{aligned} & y_i\log(h(x_i)) + (1-y_i)\log(1-h(x_i)) \\ &= y_i\log(\frac{1}{1+e^{-w^T x_i}}) + (1-y_i)\log(1-\frac{1}{1+e^{-w^T x_i}}) \\ &= -y_i\log(1+e^{-w^T x_i}) - (1-y_i)\log(1+e^{w^T x_i}) \end{aligned}$$

3.逻辑回归求解

求解过程： $\frac{\partial J(w)}{\partial w_j} = \frac{1}{m}\sum_{i=1}^{m}(h(x_i) - y_i)x_i^{(j)}$ ：

$$\frac{\partial J(w)}{\partial w_j} = \frac{\partial}{\partial w_j}\left(-\frac{1}{m}\sum_{i=1}^{m}\left(-y_i \log\left(1+e^{-w^T x_i}\right) - (1-y_i)\log\left(1+e^{w^T x_i}\right)\right)\right)$$

$$= -\frac{1}{m}\sum_{i=1}^{m}\left(-y_i \frac{-x_i^{(j)} e^{-w^T x_i}}{1+e^{-w^T x_i}} - (1-y_i)\frac{x_i^{(j)} e^{w^T x_i}}{1+e^{w^T x_i}}\right)$$

$$= -\frac{1}{m}\sum_{i=1}^{m}\left(y_i \frac{e^{-w^T x_i}}{1+e^{-w^T x_i}} + y_i \frac{e^{w^T x_i}}{1+e^{w^T x_i}} - \frac{e^{w^T x_i}}{1+e^{w^T x_i}}\right)x_i^{(j)}$$

$$= -\frac{1}{m}\sum_{i=1}^{m}\left(y_i\left(\frac{1}{1+e^{w^T x_i}} + \frac{e^{w^T x_i}}{1+e^{w^T x_i}}\right) - \boxed{\frac{1}{1+e^{-w^T x_i}}}\right)x_i^{(j)}$$

$$\boxed{h(x_i) = \frac{1}{1+e^{-w^T x_i}}}$$

$$= -\frac{1}{m}\sum_{i=1}^{m}(y_i - h(x_i))x_i^{(j)} \quad = \frac{1}{m}\sum_{i=1}^{m}(h(x_i) - y_i)x_i^{(j)}$$

4.逻辑回归代码实现

课后作业

Sigmoid 函数

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

```python
def sigmoid(z):
    """sigmoid函数定义
    """
    return 1 / (1+np.exp(-z))
```

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

- 请写出向量形式的逻辑回归梯度计算python代码

标量形式：$\frac{\partial J(w)}{\partial w_j} = \frac{1}{m}\sum_{i=1}^{m}(h(x_i) - y_i)x_i^{(j)}$

```python
def compute_gradient(X, Y, w):
    """计算梯度
    args:
        X: 输入, 维度为[m, n], m个样本, 每个样本维度为n
        Y: 标签, 维度为[m, 1]
        w: 可学习参数, 维度为[n, 1]
    outputs:
        dw: 梯度, 维度为[n ,1]
    """
```

向量形式：$\frac{\partial J(w)}{\partial w} = \frac{1}{m}X^T(h(X) - Y)$

将代码整理成word文档，发送至2385464960@qq.com （周楚皓），邮件主题为“机器学习第三次课_学号_姓名”

谢 谢！