四川大學
SICHUAN UNIVERSITY

机器学习-第二章 回归

教师：胡俊杰 副教授

邮箱：hujunjie@scu.edu.cn

本章目录

1. 线性回归

01 线性回归

02 梯度下降

03 正则化

04 回归的评价指标

回归的概念

回归和分类是监督学习中的两种代表性任务

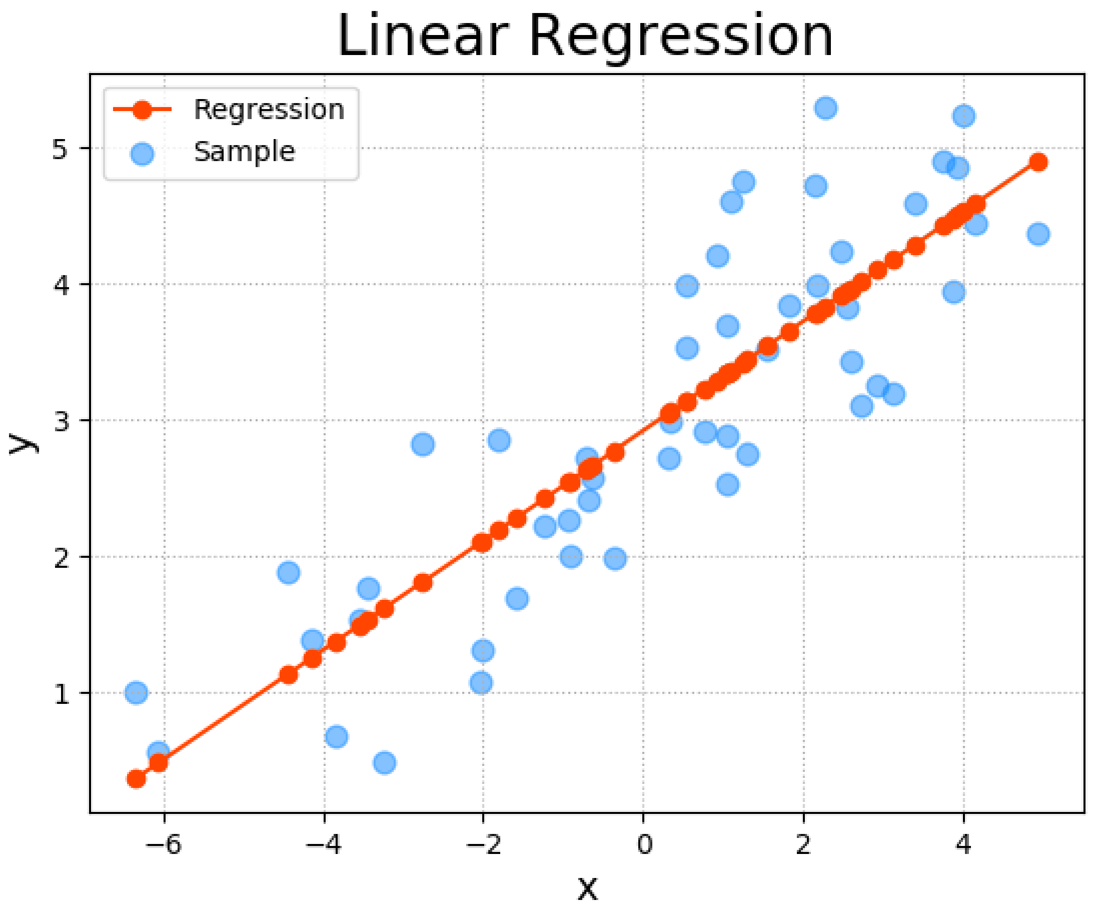
- ✓ 回归（Regression） **标签连续**
  - ✓ 根据成都市的人口密度、人均GPD，预测当前房价?

- ✓ 分类（Classification） **标签离散**
  - ✓ 根据患者的影像数据来判断肿瘤的良性或恶性?

线性回归-概念

线性回归（Linear Regression）

是一种通过变量的线性组合来进行预测的线性模型，其目的是找到一条直线或者一个平面或者更高维的超平面，使得预测值与真实值之间的误差最小化



$$y = ax + b$$

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

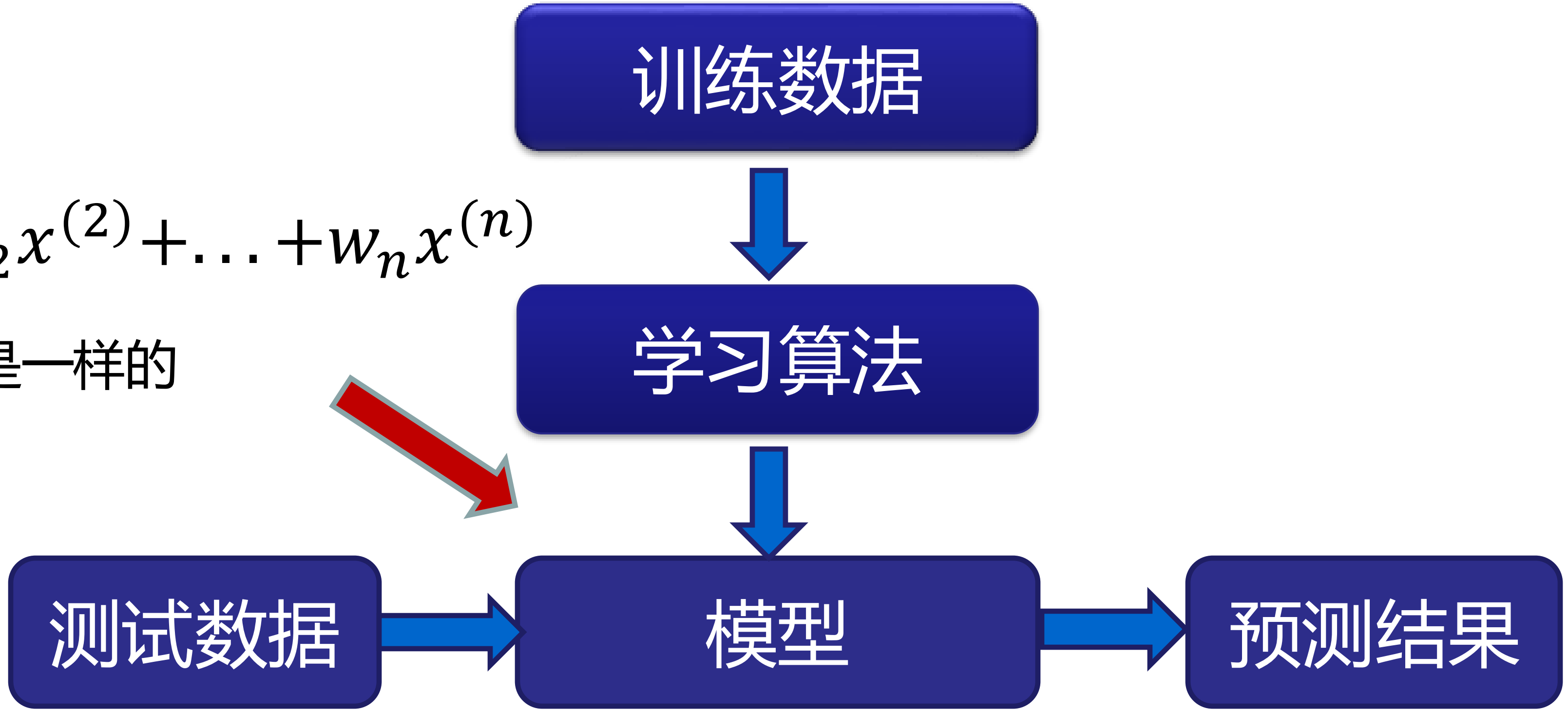
2、$h(x)$也可写成$f(x)$

线性回归-算法流程

x 和y 的关系

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

- 注：书中将w_i写为$w^{(i)}$，含义是一样的



可以设$x^{(0)} = 1$，则

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

向量形式： $h(x) = w^T x$

- 向量w与x均为列向量

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-算法流程

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

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

- 损失函数值越小，代表模型的预测值$h(x)$与标签y越接近，即模型越优
- 损失函数的期望

$$E[L(y, h(x))] = \int L(y, h(x))\, P(x, y)\, dxdy$$

也称为期望损失（expected loss），或风险函数（risk function）

- x和y为随机变量，然而实际应用过程中，仅能获取到数量有限的样本(x_i, y_i)，关于数据集$D = \{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\}$的平均损失也称为经验风险或经验损失

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

注：为了表述简便，后续方程省略了$\frac{1}{\mathrm{m}}$

线性回归-算法流程

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

损失函数采用平方和损失： $L(h(x_i), y_i) = \frac{1}{2}(h(x_i) - y_i)^2$

要找到一组 $(w_0, w_1, w_2, \ldots, w_n)$ ，使得对于全体m个训练样本

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

注：损失函数的系数1/2是为了便于计算，使对平方项求导后的常数系数为1，这样在形式上稍微简单一些。有些教科书把系数设为1/2，有些设置为1，这些都不影响结果

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)

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

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

$$\vdots$$

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

$\downarrow$ m个线性方程

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

线性回归-最小二乘法(LSM, Least Squares Method)

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

$$\hat{Y} \qquad\qquad X \qquad\qquad w$$

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

需要用到向量平方的性质:

$$\sum_i z_i^2 = z^T z$$

使$J(w)$最小，即求$\frac{\partial J(w)}{\partial w}$，使得 $\frac{\partial J(w)}{\partial w} = 0$

线性回归-最小二乘法(LSM, Least Squares Method)

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

$$\frac{\partial J(w)}{\partial w}=\frac{1}{2}\frac{\partial}{\partial w}(Xw-Y)^T(Xw-Y)\quad=\frac{1}{2}\frac{\partial}{\partial w}(w^TX^TXw-Y^TXw-w^TX^TY+Y^TY)$$

$$\frac{\partial J(w)}{\partial w}=\frac{1}{2}(2X^TXw-X^TY-X^TY+0)=\frac{1}{2}(2X^TXw-2X^TY)$$

$$=X^TXw-X^TY$$

令$\frac{\partial J(w)}{\partial w}=0,$

则有$w=(X^TX)^{-1}X^TY$

需要用到以下几个矩阵的求导法则:

$$\frac{dAX}{dX}=A^T\qquad\frac{dX^TA}{dX}=A$$

$\frac{\partial X^TAX}{\partial X}=(A+A^T)X$,若A为对称阵, $\frac{\partial X^TAX}{\partial X}=2AX$

线性回归-最小二乘法(LSM, Least Squares Method)

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$J(w)$

1. 线性回归

线性回归-最小二乘法(LSM, Least Squares Method)

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

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

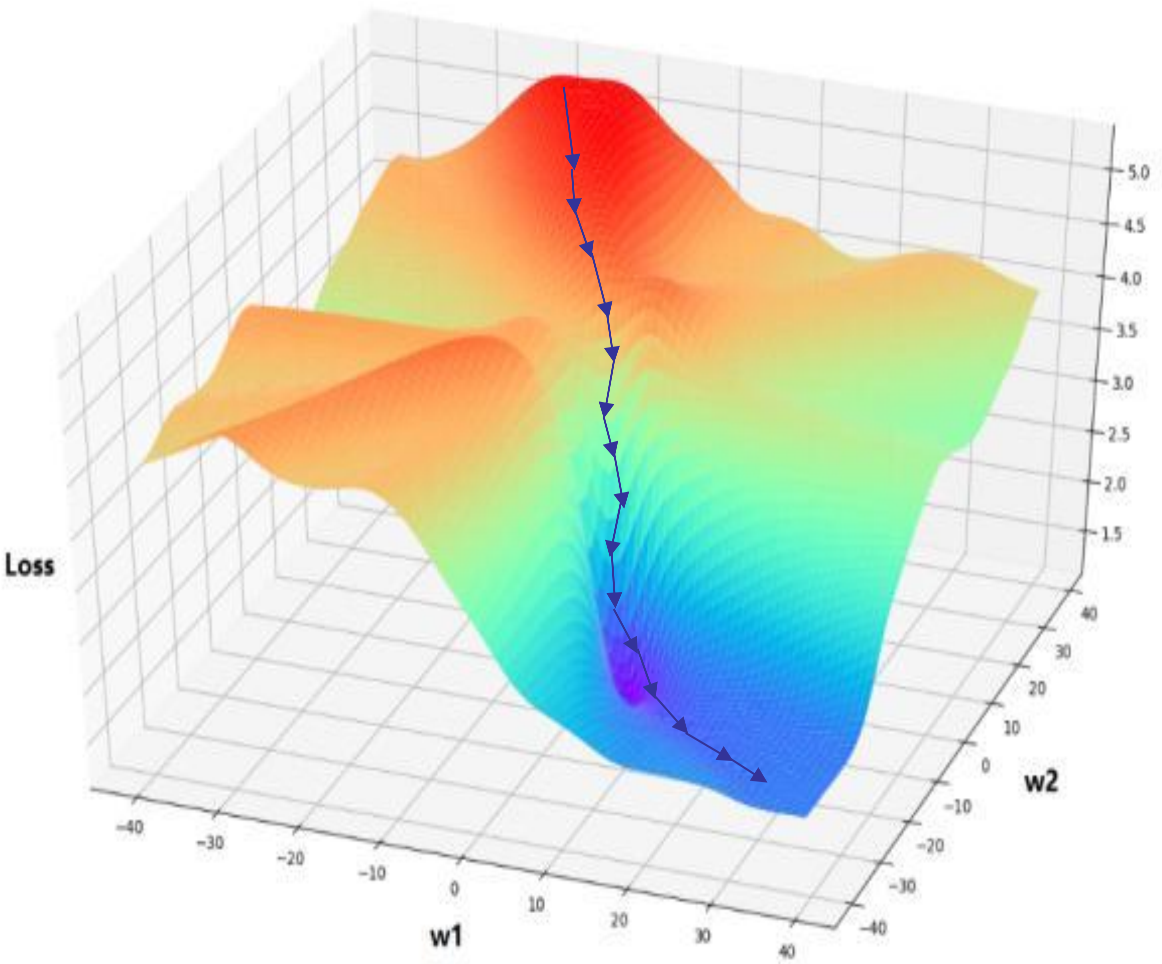
计算得到$w=(X^TX)^{-1}X^TY$

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$O(n^3)$

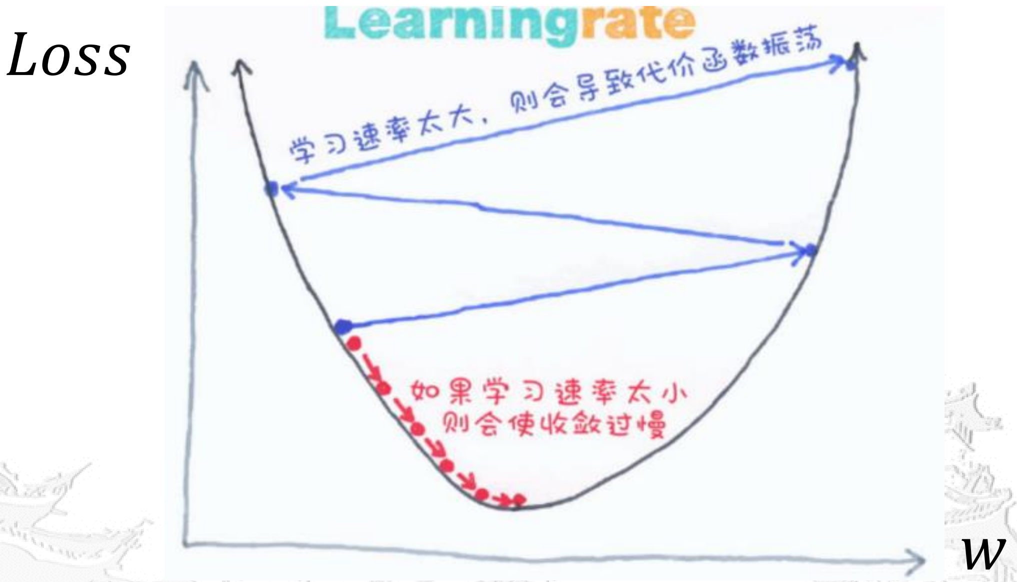
$$\underbrace{\begin{bmatrix}\hat{y}_1\\ \hat{y}_2\\ \cdots\\ \hat{y}_m\end{bmatrix}}_{\hat{Y}}=\underbrace{\begin{bmatrix}1 & x_1^{(1)} & x_1^{(2)} & x_1^{(3)} & \cdots & x_1^{(n)}\\ 1 & x_2^{(1)} & x_2^{(2)} & x_2^{(3)} & \cdots & x_2^{(n)}\\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots\\ 1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

是否有更简单的优化$J(w)$的方法?

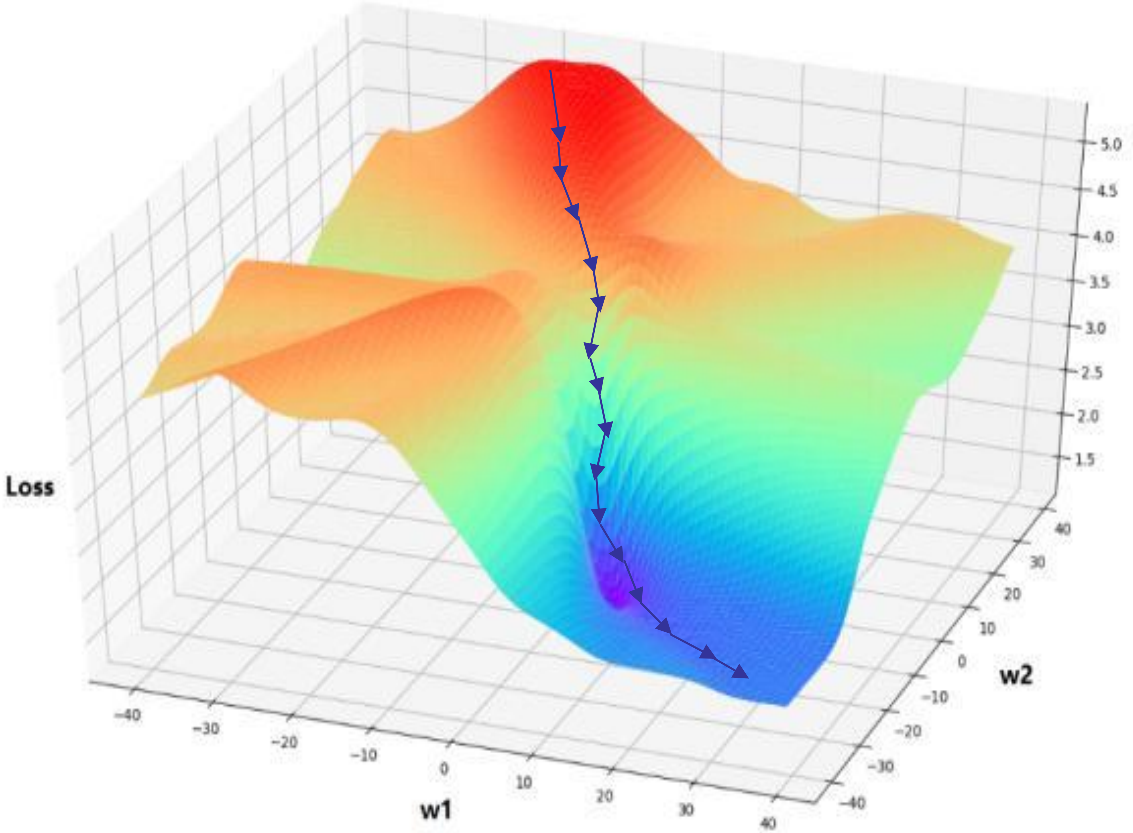
梯度下降



- 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)



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

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

梯度下降的两种形式

- 梯度下降（Gradient Descent, GD）
  - 使用全体训练样本计算梯度$\frac{\partial J(w)}{\partial w}$
- 随机梯度下降（Stochastic Gradient Descent, SGD）
  - 使用一个或多个样本计算梯度$\frac{\partial J(w)}{\partial w}$
  - 因为计算得到的梯度不准确，因此称为随机梯度下降

如何计算$\frac{\partial J(w)}{\partial w}$

梯度下降的两种形式

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

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

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

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

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

梯度下降的两种形式

梯度下降（Gradient Descent）

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

对于单个训练样本i：

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

学习率

$$w_j := w_j - \alpha \frac{1}{m} \sum_{i=1}^{m} \overbrace{\left((h(x_i) - y_i){x_i}^{(j)}\right)}^{\text{梯度}}$$

使用全体训练样本计算梯度，计算量大

使用全体m个训练样本

梯度下降的两种形式

随机梯度下降（Stochastic Gradient Descent）

梯度下降的每一步中，用到了一定批量的训练样本

每次使用b个训练样本计算梯度，便更新一次参数 w

$$w_j := w_j - \alpha \frac{1}{b} \sum_{k=i}^{i+b-1} \left((h(x_k) - y_k) x_k^{(j)} \right)$$

梯度下降的两种形式

梯度下降（Gradient Descent）

使用全体m个训练样本计算梯度，并更新参数w

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

随机梯度下降（Stochastic Gradient Descent, SGD）

使用部分b个训练样本计算梯度，并更新一次参数 w

$$w_j := w_j - \alpha \frac{1}{b} \sum_{k=i}^{i+b-1} \left((h(x_k) - y_k) x_k^{(j)} \right)$$

随机梯度下降算法

第1步. 准备训练数据集 $D=\{(x,y)\}$

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

第3步. 从 D中选择b个训练样本从$(x_i, y_i) \in D^b$

$$\frac{\partial J(w)}{\partial w_j} \leftarrow \frac{\partial J(w)}{\partial w_j} + (h(x_i) - y_i){x_i}^{(j)}$$ //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

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

梯度下降与最小二乘法比较

梯度下降：需要选择学习率α，多次迭代更新，当特征数量n大时也能较好适用

最小二乘法：不需要选择学习率α，直接计算解析解，需要计算$(X^TX)^{-1}$，如果特征数量 n 较大则运算代价大，因为矩阵逆的计算时间复杂度为$O(n^3)$，只适用于小规模的线性模型

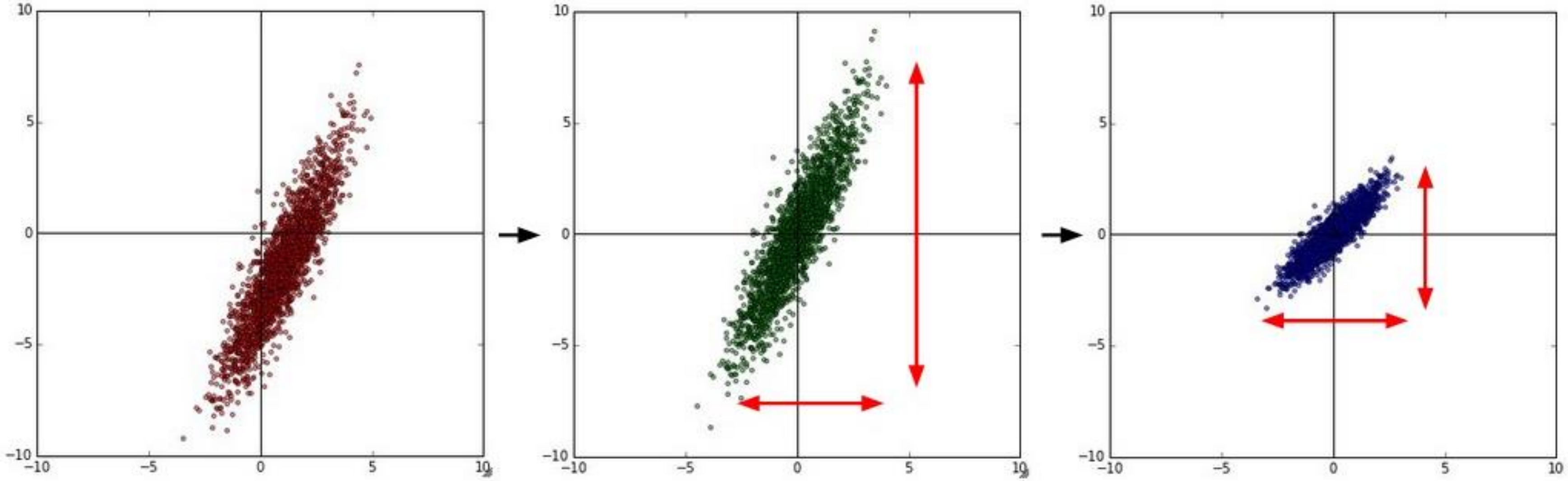
数据归一化/标准化

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**
……	……	……	……	……

各输入变量的数值范围不一致，将导致某些变量对模型的影响更大

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

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

3. 正则化

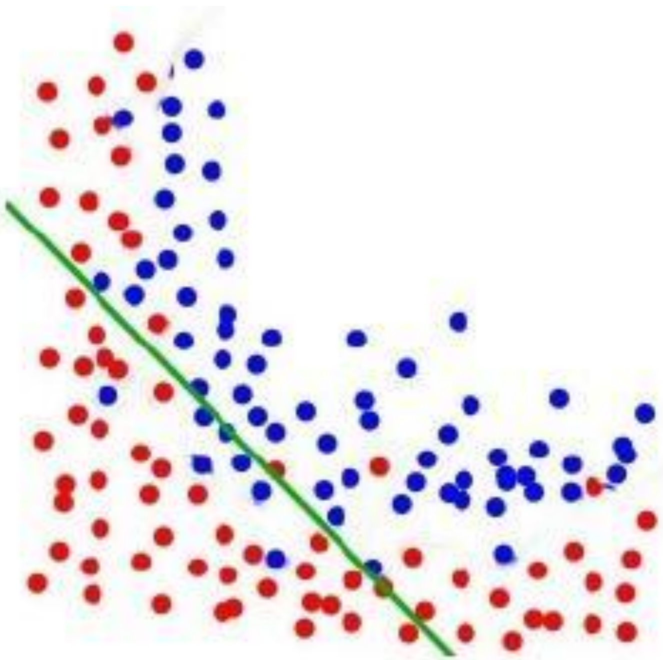
01 线性回归

02 梯度下降

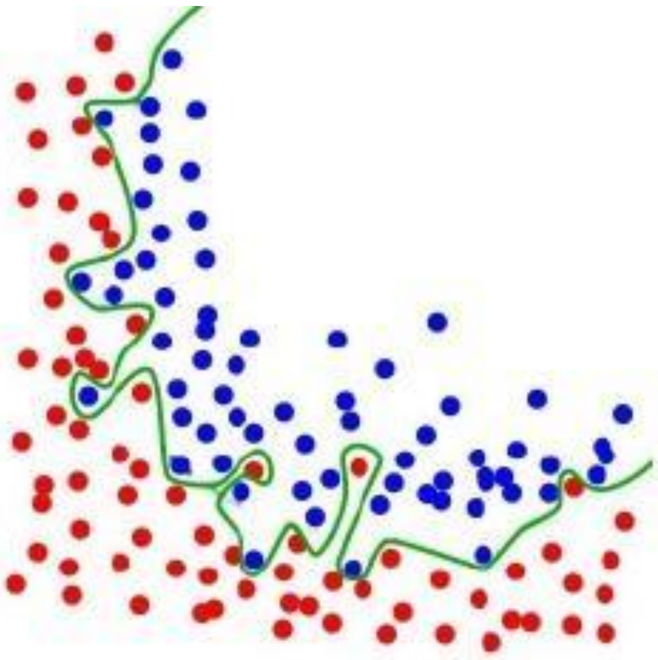
03 正则化

04 回归的评价指标

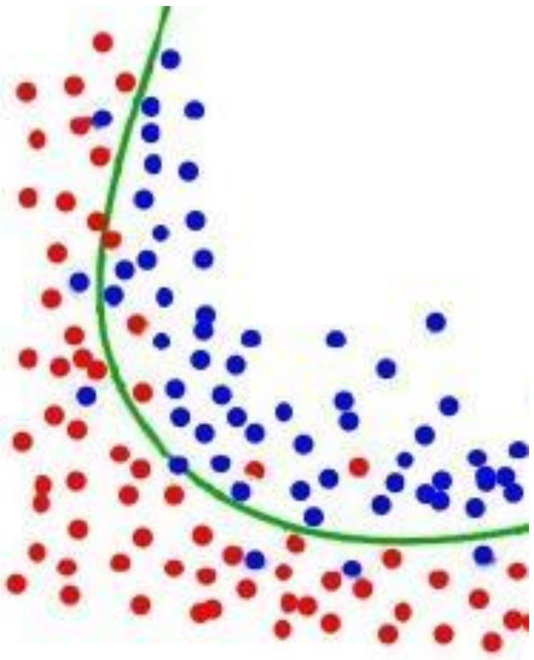
过拟合和欠拟合



欠拟合



过拟合



正合适

过拟合的处理

1.获得更多的有效训练数据

使用更多的训练数据是解决过拟合问题最有效的手段，因为更多的样本能够让模型学习到更多更有效的特征，减小噪声的影响

2.**降维**

即丢弃一些不能帮助模型预测的特征，可以是手工选择保留哪些特征，或者使用降维算法来帮忙（例如PCA）

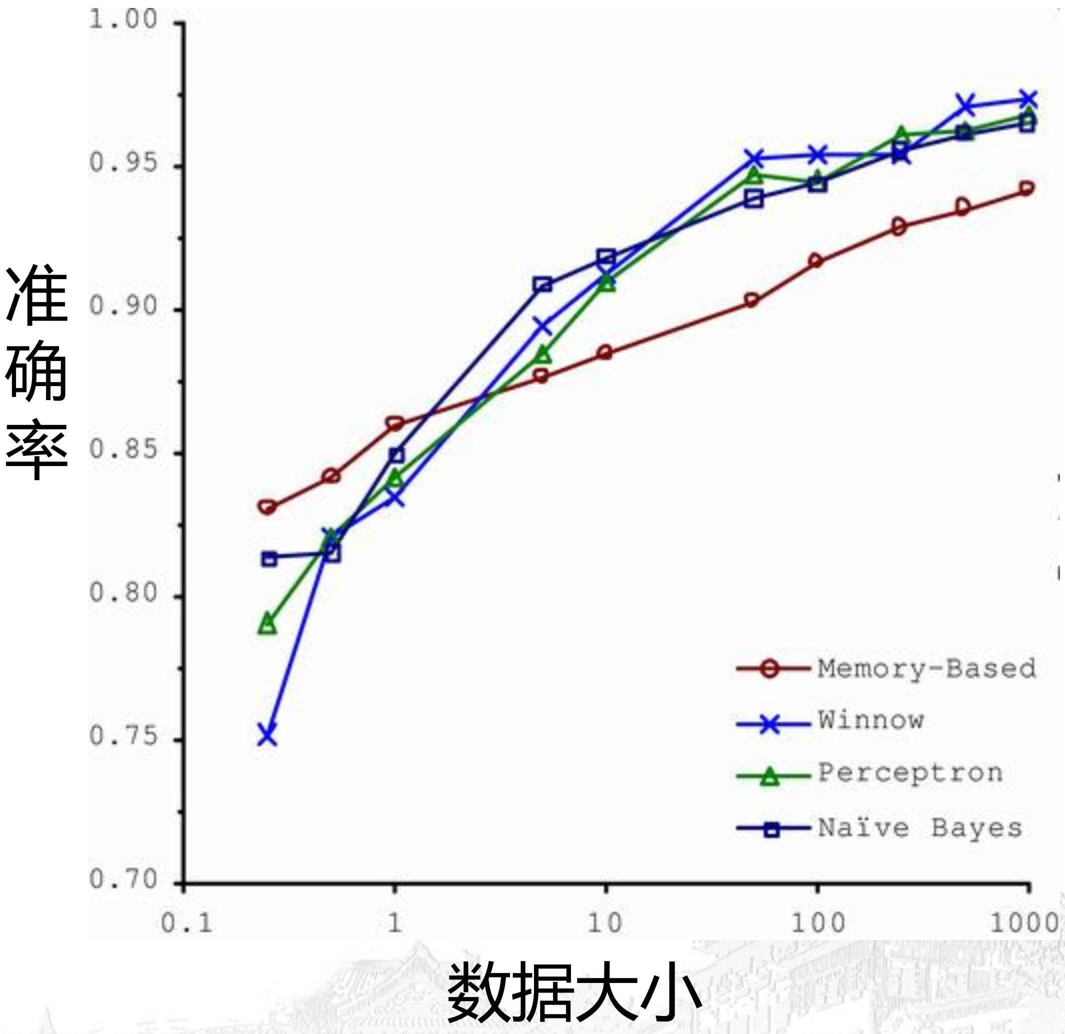
3.**正则化**

正则化(regularization)的技术，使模型参数稀疏，常见的正则化方法包括L1、L2正则化

4.**集成学习方法**

集成学习是把多个模型集成在一起，来降低单一模型的过拟合风险

数据决定一切

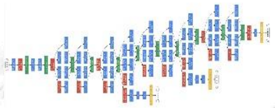


通过这张图可以看出，不同算法在输入的数据达到一定量级后，准确率均有提升

成功的机器学习应用不仅依赖好的算法，而且依赖高质量数据

大数据

大模型



大算力

欠拟合的处理

1.**添加新特征**

当特征不足或者现有特征与样本标签的相关性不强时，模型容易出现欠拟合。通过挖掘组合特征等新的特征，往往能够取得更好的效果

2.**增加模型复杂度**

简单模型的学习能力较差，通过增加模型的复杂度可以使模型拥有更强的拟合能力。例如，在线性模型中添加高次项，以及在神经网络模型中增加网络层数或神经元个数等

3.**减小正则化强度**

正则化是用来缓解过拟合的，但当模型出现欠拟合现象时，则需要有针对性地减小正则化强度

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）

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

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

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

$$\boxed{\begin{aligned} &\min_w J(w) \\ &s.t. \|w\|_0 \le C \end{aligned}}$$

$s.t.$: 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

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

正则化（Regularization）

$$\min_{w} J(w)$$
$$s.t. \|w\|_1 \leq C$$

拉格朗日乘子法

正则化系数

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

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

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

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，$\lambda>0$
- $0\le\rho\le0$为比例系数，调整L_1正则化与L_2正则化的比例

4. 回归的评价指标

01 线性回归

02 梯度下降

03 正则化

04 回归的评价指标

回归的评价指标

均方误差（Mean Square Error,MSE）

$$MSE=\frac{1}{m}\sum_{i=1}^{m}(y_i-\hat{y}_i)^2$$

均方根误差 RMSE(Root Mean Square Error,RMSE)

$$RMSE(y,\hat{y})=\sqrt{\frac{1}{m}\sum_{i=1}^{m}(y_i-\hat{y}_i)^2}$$

平均绝对误差（Mean Absolute Error,MAE）

$$MAE(y,\hat{y})=\frac{1}{m}\sum_{i=1}^{m}|y_i-\hat{y}_i|$$

其中，y_i和$\hat{y}_i$分别表示第i个样本的真实值和预测值，m 为样本个数

回归的评价指标

R方 [$RSquared$]

$$R^2(y,\hat{y}) = 1 - \frac{\sum_{i=1}^{m}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{m}(y_i - \bar{y})^2} = 1 - \frac{\text{SSE}}{\text{SST}}$$

$$\bar{y} = \frac{1}{m}\sum_{i=1}^{m} y_i$$

$$R^2(y,\hat{y}) = 1 - \frac{\sum_{i=1}^{m}(y_i - \hat{y}_i)^2/m}{\sum_{i=1}^{m}(y_i - \bar{y})^2/m} = 1 - \frac{\text{MSE}}{\text{Var}}$$

$$SSR = \sum_{i=1}^{m}(\hat{y}_i - \bar{y})^2$$

$$SSE = \sum_{i=1}^{m}(y_i - \hat{y}_i)^2$$

$$SST = \sum_{i=1}^{m}(y_i - \bar{y})^2$$

越接近于1,说明模型拟合得越好

其中，y_i和$\hat{y}_i$分别表示第i个样本的真实值和预测值，m 为样本个数

课后作业

- 请写出右侧随机梯度下降算法的伪代码

将伪代码整理成word文档，发送至2385464960@qq.com （周楚皓），邮件主题为“机器学习第二次课_学号_姓名”

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

谢 谢！