

## 02 线性回归

# 本章目录

2

- 01** 线性回归
- 02** 梯度下降
- 03** 正则化
- 04** 回归的评价指标

# 1. 线性回归

3

**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

4

## 监督学习分为回归和分类

✓ 回归 ( Regression、Prediction )

标签连续

✓ 如何预测上海浦东的房价？

✓ 未来的股票市场走向？

✓ 分类 ( Classification )

标签离散

✓ 身高1.85m，体重100kg的男人穿什么尺码的T恤？

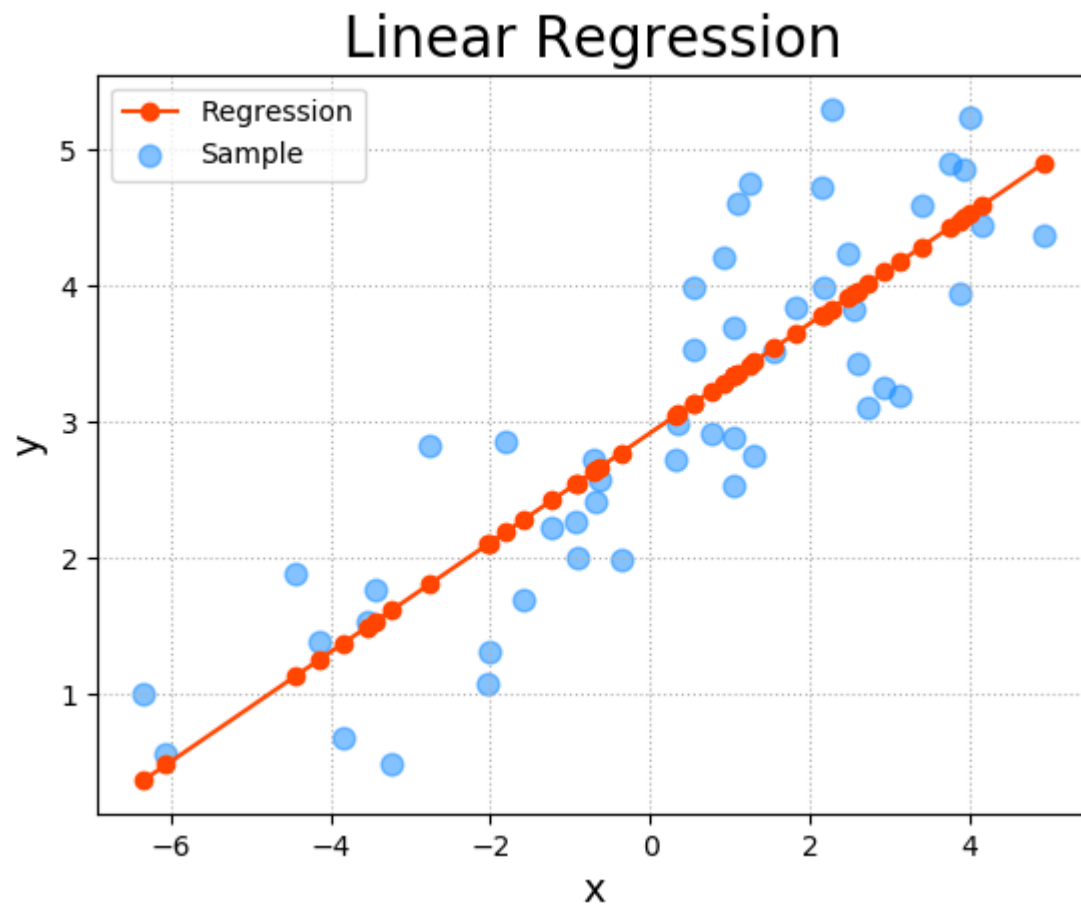
✓ 根据肿瘤的体积、患者的年龄来判断良性或恶性？

# 线性回归-概念

5

## 线性回归 ( Linear Regression )

是一种通过属性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。**



# 线性回归-符号约定

6

$m$  代表训练集中样本的数量

$n$  代表特征的数量

$x$  代表特征/输入变量

$y$  代表目标变量/输出变量

$(x, y)$  代表训练集中的样本

$(x^{(i)}, y^{(i)})$  代表第 $i$ 个观察样本

$h$  代表学习算法的解决方案或函数也称为假设 ( **hypothesis** )

$\hat{y} = h(x)$ , 代表预测的值

建筑面积	总层数	楼层	实用面积	房价
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，是一个**向量**。

上图的：
$$x^{(2)} = \begin{bmatrix} 162.2 \\ 31 \\ 8 \\ 118 \end{bmatrix} \quad y^{(2)} = 37000$$

$x_j^{(i)}$  代表特征矩阵中第 $i$ 行的第 $j$ 个特征

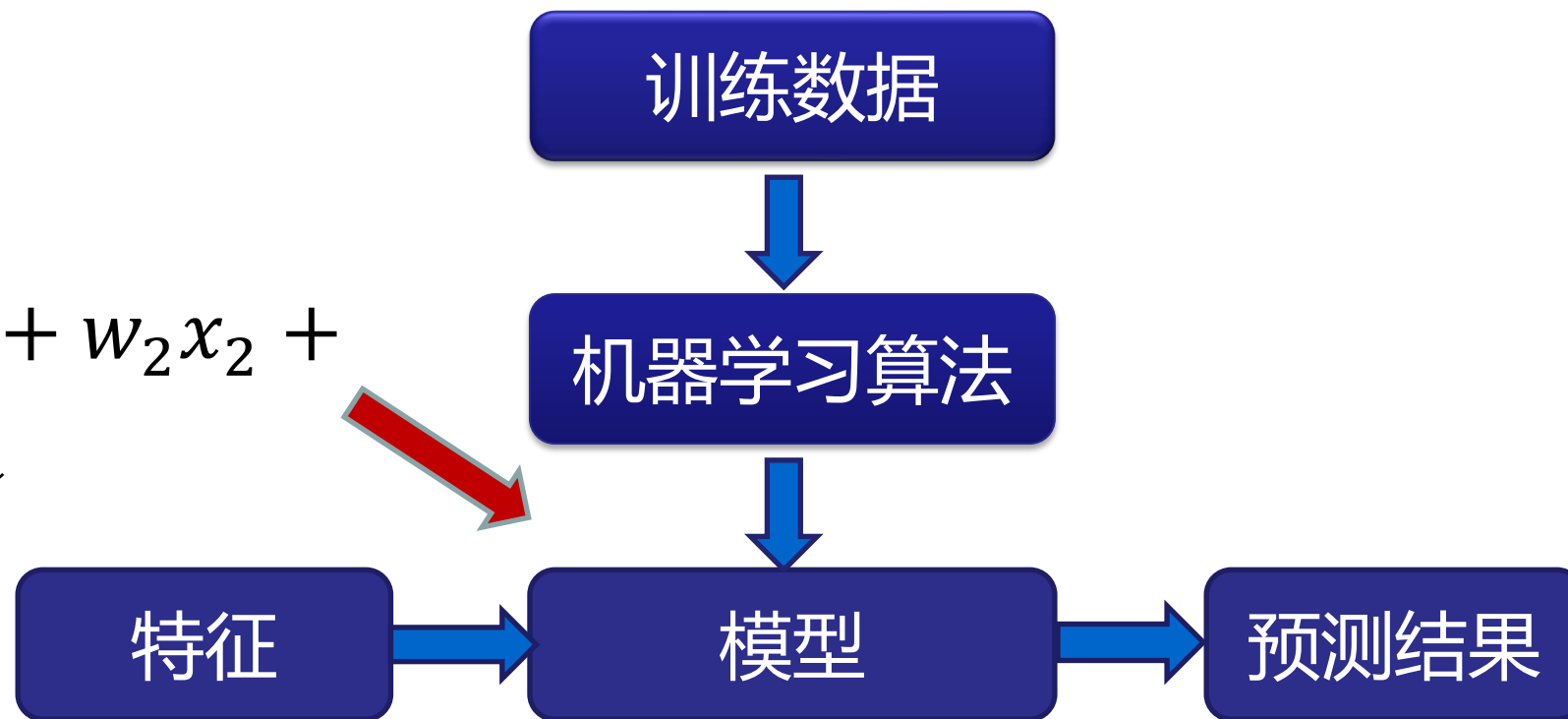
上图的 $x_2^{(2)} = 31, x_3^{(2)} = 8$

# 线性回归-算法流程

7

$x$  和  $y$  的关系

$$h(x) = w_0 + w_1x_1 + w_2x_2 + \dots + w_nx_n$$



可以设  $x_0 = 1$

则： $h(x) = w_0x_0 + w_1x_1 + w_2x_2 + \dots + w_nx_n = w^T X$

注意：若表达式  $h(x) = w_0 + w_1x_1 + w_2x_2 + \dots + w_nx_n + b$ ，则  $b$  可以融入到  $w_0$

# 线性回归-算法流程

8

$$h(x) = w_0 + w_1x_1 + w_2x_2 + \dots + w_nx_n$$

损失函数采用平方和损失：

$$l(x^{(i)}) = \frac{1}{2} (h(x^{(i)}) - y^{(i)})^2$$

要找到一组  $w(w_0, w_1, w_2, \dots, w_n)$  ,

$$\text{使得 } J(w) = \frac{1}{2} \sum_{i=1}^m (h(x^{(i)}) - y^{(i)})^2$$

(残差平方和) 最小

**损失函数**(Loss Function)度量单样本预测的错误程度，损失函数值越小，模型就越好。常用的损失函数包括：0-1损失函数、平方损失函数、绝对损失函数、对数损失函数等。

**代价函数**(Cost Function)度量全部样本集的平均误差。常用的代价函数包括均方误差、均方根误差、平均绝对误差等。

**目标函数**(Object Function)代价函数和正则化函数，最终要优化的函数。



# 线性回归-最小二乘法(LSM)

9

要找到一组  $w(w_0, w_1, w_2, \dots, w_n)$  , 使得  $J(w) = \frac{1}{2} \sum_{i=1}^m (h(x^{(i)}) - y^{(i)})^2$

(残差平方和) 最小, 即最小化:  $\frac{\partial J(w)}{\partial w}$

将向量表达形式转为矩阵表达形式, 则有  $J(w) = \frac{1}{2} (Xw - Y)^2$  , 其中  $X$  为  $m$  行  $n + 1$  列的矩阵 (  $m$  为样本个数,  $n$  为特征个数 ) ,  $w$  为  $n + 1$  行 1 列的矩阵 (包含了  $w_0$ ) ,  $Y$  为  $m$  行 1 列的矩阵, 则  $J(w) = \frac{1}{2} (Xw - Y)^2 = \frac{1}{2} (Xw - Y)^T (Xw - Y)$

$$X = \begin{bmatrix} 1 & x_1^{(1)} & x_2^{(1)} & x_3^{(1)} & \dots & x_n^{(1)} \\ 1 & x_1^{(2)} & x_2^{(2)} & x_3^{(2)} & \dots & x_n^{(2)} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_1^{(m)} & x_2^{(m)} & x_3^{(m)} & \dots & x_n^{(m)} \end{bmatrix} \quad Y = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \dots \\ y^{(m)} \end{bmatrix}$$

需要用到向量平方的性质:

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

# 线性回归-最小二乘法(LSM)

10

为最小化，接下来对 $J(w)$ 偏导，

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

由于中间两项互为转置:

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

$$= X^T X w - X^T Y$$

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

$$\text{则有 } w = (X^T X)^{-1} X^T Y$$

需要用到以下几个矩阵的求导法则:

$$\frac{\partial X^T X}{\partial X} = 2X \quad \frac{\partial AX}{\partial X} = A^T$$

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

# 1. 线性回归

11

**01** 线性回归

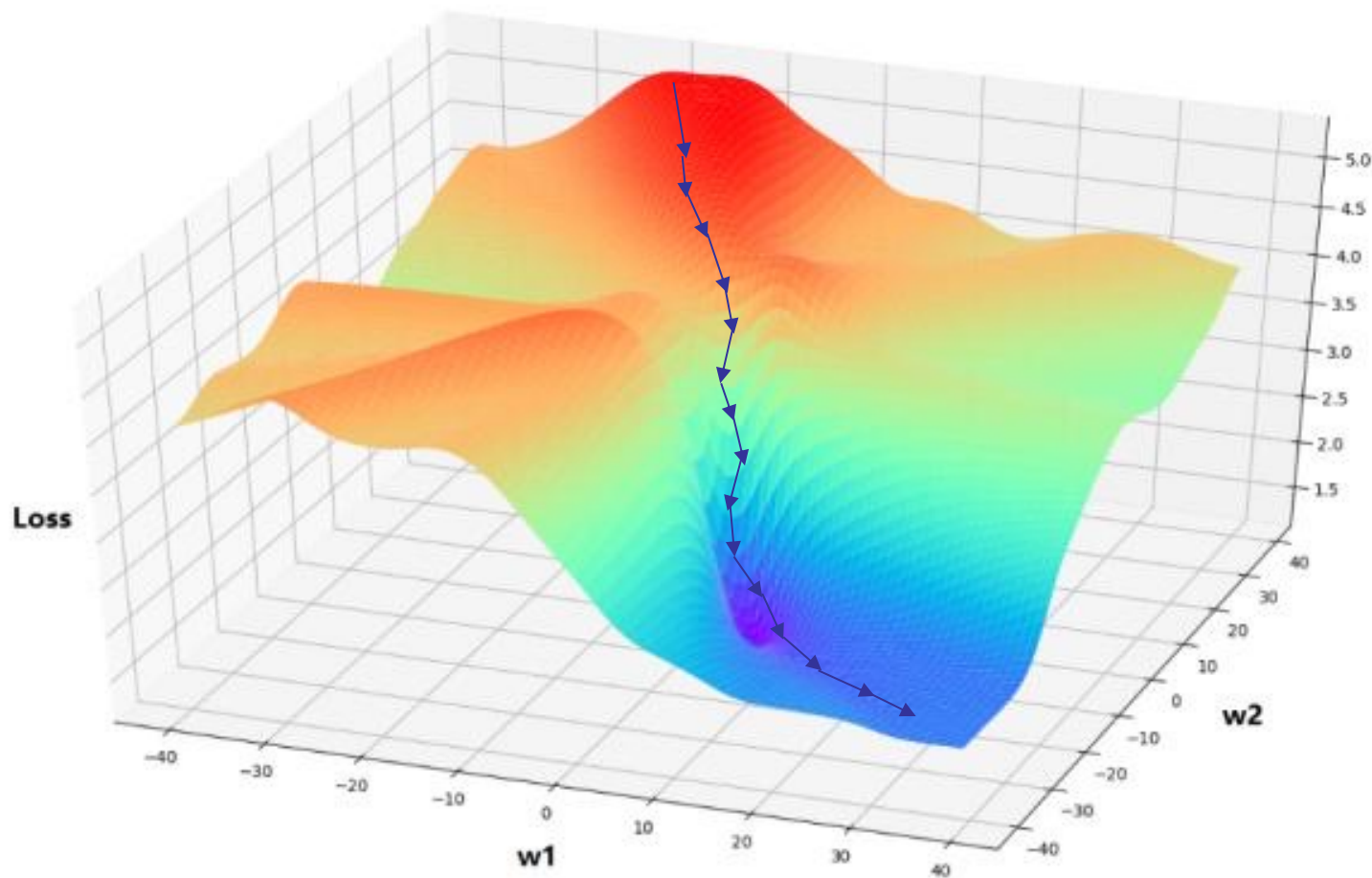
**02** 梯度下降

**03** 正则化

**04** 回归的评价指标

# 梯度下降

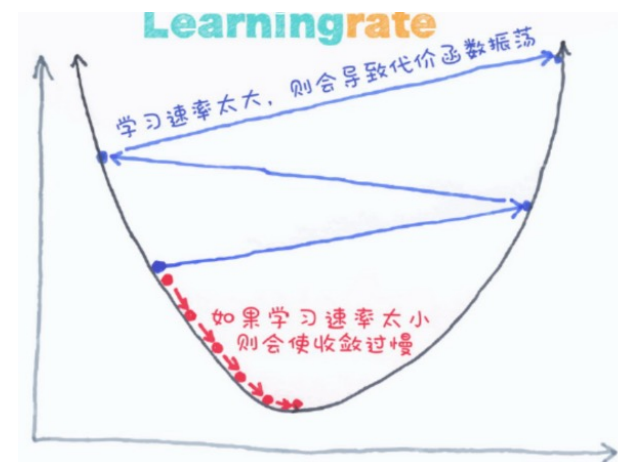
12



学习率

$\alpha$

步长



# 梯度下降的三种形式

13

## 批量梯度下降 ( Batch Gradient Descent, BGD )

梯度下降的每一步中，都用到了所有的训练样本

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

梯度下降的每一步中，用到一个样本，在每一次计算之后便更新参数，而不需要首先将所有的训练集求和

## 小批量梯度下降 ( Mini-Batch Gradient Descent, MBGD )

梯度下降的每一步中，用到了一定批量的训练样本

# 梯度下降的三种形式

14

## 批量梯度下降 ( Batch Gradient Descent )

梯度下降的每一步中，都用到了所有的训练样本

参数更新

学习率

梯度

$$w_j := w_j - \alpha \frac{1}{m} \sum_{i=1}^m \left( (h(x^{(i)}) - y^{(i)}) \cdot x_j^{(i)} \right)$$

(同步更新  $w_j$  ,  $(j=0,1,...,n)$  )

# 梯度下降的三种形式

15

## 随机梯度下降 ( Stochastic Gradient Descent )

**推导**  $w = w - \alpha \cdot \frac{\partial J(w)}{\partial w}$   $h(x) = w^T X = w_0 x_0 + w_1 x_1 + w_2 x_2 + \dots + w_n x_n$

$$J(w) = \frac{1}{2} (h(x^{(i)}) - y^{(i)})^2$$

$$\frac{\partial}{\partial w_j} J(w) = \frac{\partial}{\partial w_j} \frac{1}{2} (h(x^{(i)}) - y^{(i)})^2 = 2 \cdot \frac{1}{2} (h(x^{(i)}) - y^{(i)}) \cdot \frac{\partial}{\partial w_j} (h(x^{(i)}) - y^{(i)})$$

$$= (h(x^{(i)}) - y^{(i)}) \cdot \frac{\partial}{\partial w_j} \left( \sum_{i=0}^n (w_i x_i^{(i)} - y^{(i)}) \right)$$

$$= (h(x^{(i)}) - y^{(i)}) x_j^{(i)}$$

# 梯度下降的三种形式

16

## 随机梯度下降 ( Stochastic Gradient Descent )

梯度下降的每一步中，用到一个样本，在每一次计算之后便更新参数，而不需要首先将所有的训练集求和

### 参数更新

$$w_j := w_j - \alpha (h(x^{(i)}) - y^{(i)}) x_j^{(i)}$$

(同步更新  $w_j$  ,  $(j=0,1,...,n)$  )



# 梯度下降的三种形式

17

## 小批量梯度下降 ( Mini-Batch Gradient Descent )

梯度下降的每一步中，用到了一定批量的训练样本

每计算常数  $b$  次训练实例，便更新一次参数  $w$

### 参数更新

$$w_j := w_j - \alpha \frac{1}{b} \sum_{k=i}^{i+b-1} (h(x^{(k)}) - y^{(k)}) x_j^{(k)}$$

(同步更新  $w_j$  ,  $(j=0,1,...,n)$  )

$b=1$  ( 随机梯度下降,SGD )  
 $b=m$  ( 批量梯度下降,BGD )  
 $b=batch\_size$  , 通常是2的指数倍 , 常见有32,64,128等。  
( 小批量梯度下降,MBGD )

# 梯度下降与最小二乘法比较

18

**梯度下降**：需要选择学习率 $\alpha$ ，需要多次迭代，当特征数量 $n$ 大时也能较好适用，适用于各种类型的模型。

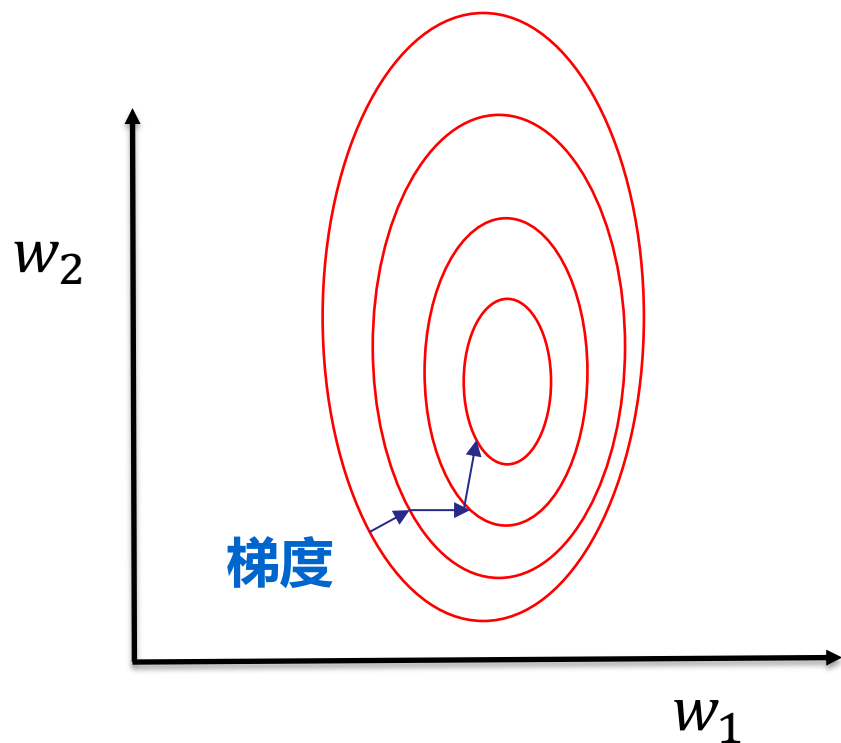
**最小二乘法**：不需要选择学习率 $\alpha$ ，一次计算得出，需要计算 $(X^T X)^{-1}$ ，如果特征数量 $n$ 较大则运算代价大，因为矩阵逆的计算时间复杂度为 $O(n^3)$ ，通常来说当 $n$ 小于10000 时还是可以接受的，只适用于线性模型，不适合逻辑回归模型等其他模型。

# 数据归一化/标准化

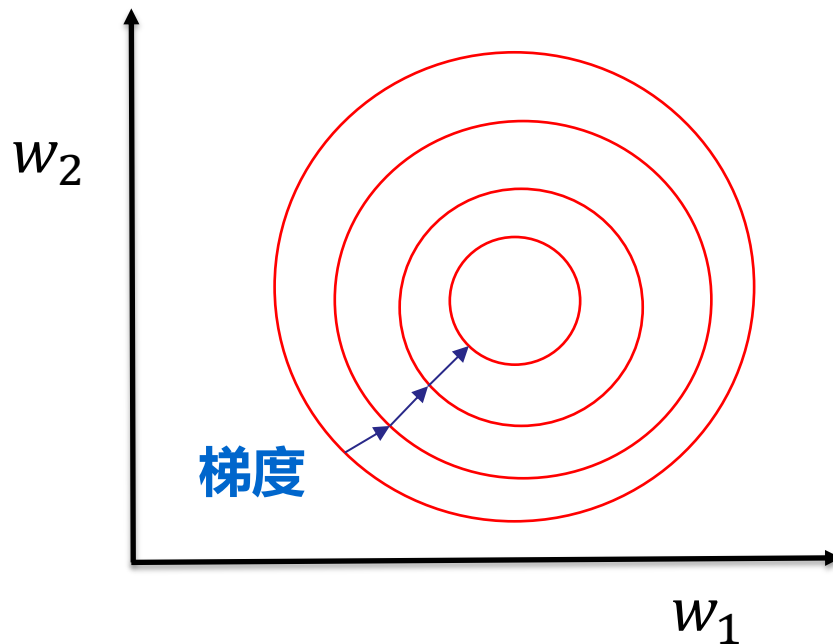
19

## 为什么要标准化/归一化？

**提升模型精度：**不同维度之间的特征在数值上有一定比较性，可以大大提高分类器的准确性。



**加速模型收敛：**最优解的寻优过程明显会变得平缓，更容易正确的收敛到最优解。



# 数据归一化/标准化

20

## 归一化（最大 - 最小规范化）

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

将数据映射到[0,1]区间

数据归一化的目的是使得各特征对目标变量的影响一致，会将特征数据进行伸缩变化，所以数据归一化是会改变特征数据分布的。

## Z-Score标准化

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

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

处理后的数据均值为0，方差为1

数据标准化为了不同特征之间具备可比性，经过标准化变换之后的特征数据分布没有发生改变。

就是当数据特征取值范围或单位差异较大时，最好是做一下标准化处理。

# 数据归一化/标准化

21

## 需要做数据归一化/标准化

线性模型，如基于距离度量的模型包括KNN(K近邻)、K-means聚类、感知机和SVM。另外，线性回归类的几个模型一般情况下也是需要做数据归一化/标准化处理的。

## 不需要做数据归一化/标准化

决策树、基于决策树的Boosting和Bagging等集成学习模型对于特征取值大小并不敏感，如随机森林、XGBoost、LightGBM等树模型，以及朴素贝叶斯，以上这些模型一般不需要做数据归一化/标准化处理。

# 3. 正则化

22

**01** 线性回归

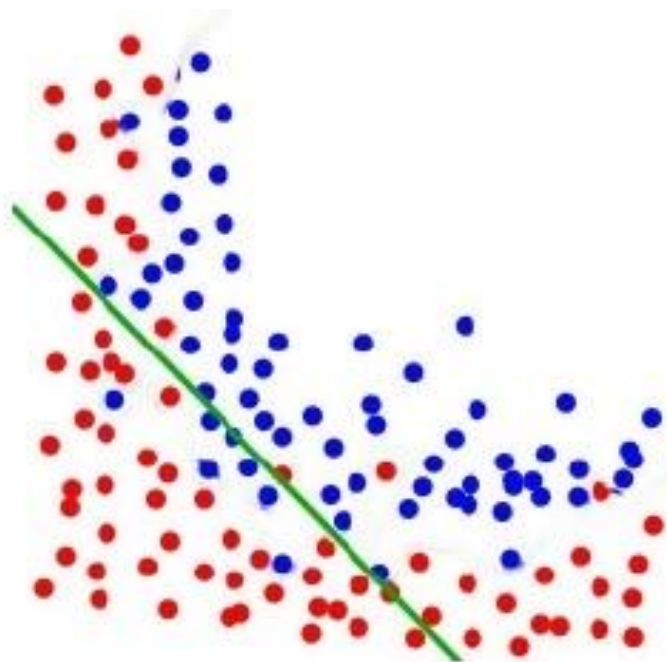
**02** 梯度下降

**03** 正则化

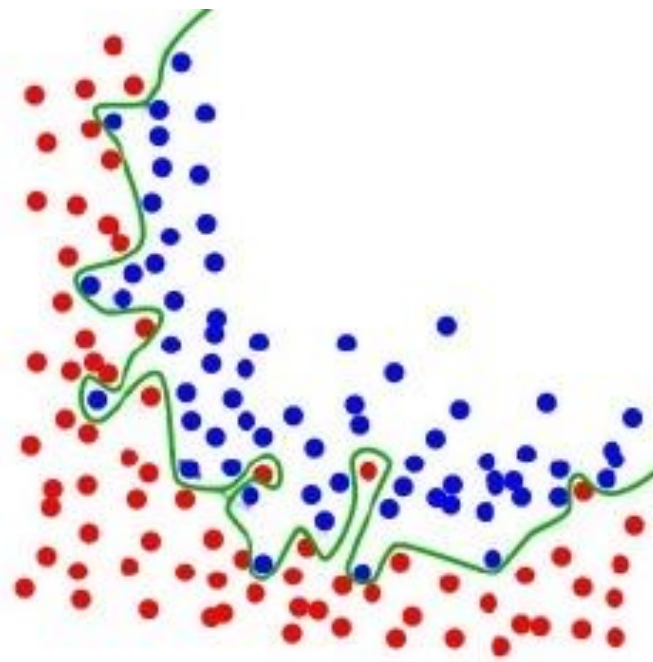
**04** 回归的评价指标

# 过拟合和欠拟合

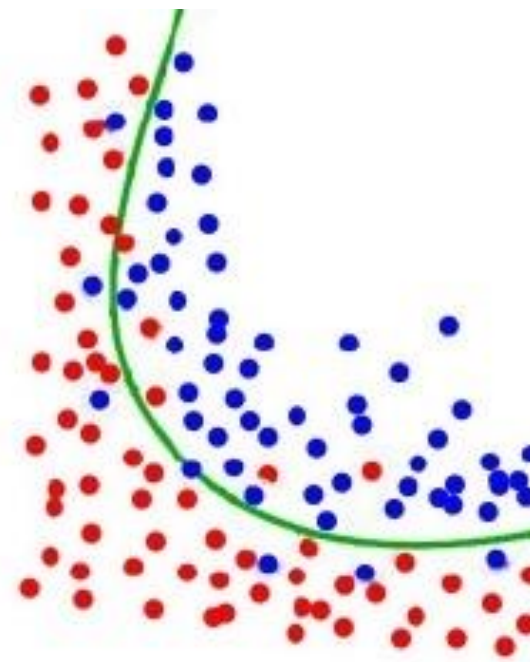
23



欠拟合



过拟合



正合适

# 过拟合的处理

24

## 1. 获得更多的训练数据

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我们正确预测的特征。可以是手工选择保留哪些特征，或者使用一些模型选择的算法来帮忙（例如PCA）。

## 3. 正则化

正则化(regularization)的技术，保留所有的特征，但是减少参数的大小（magnitude），它可以改善或者减少过拟合问题。

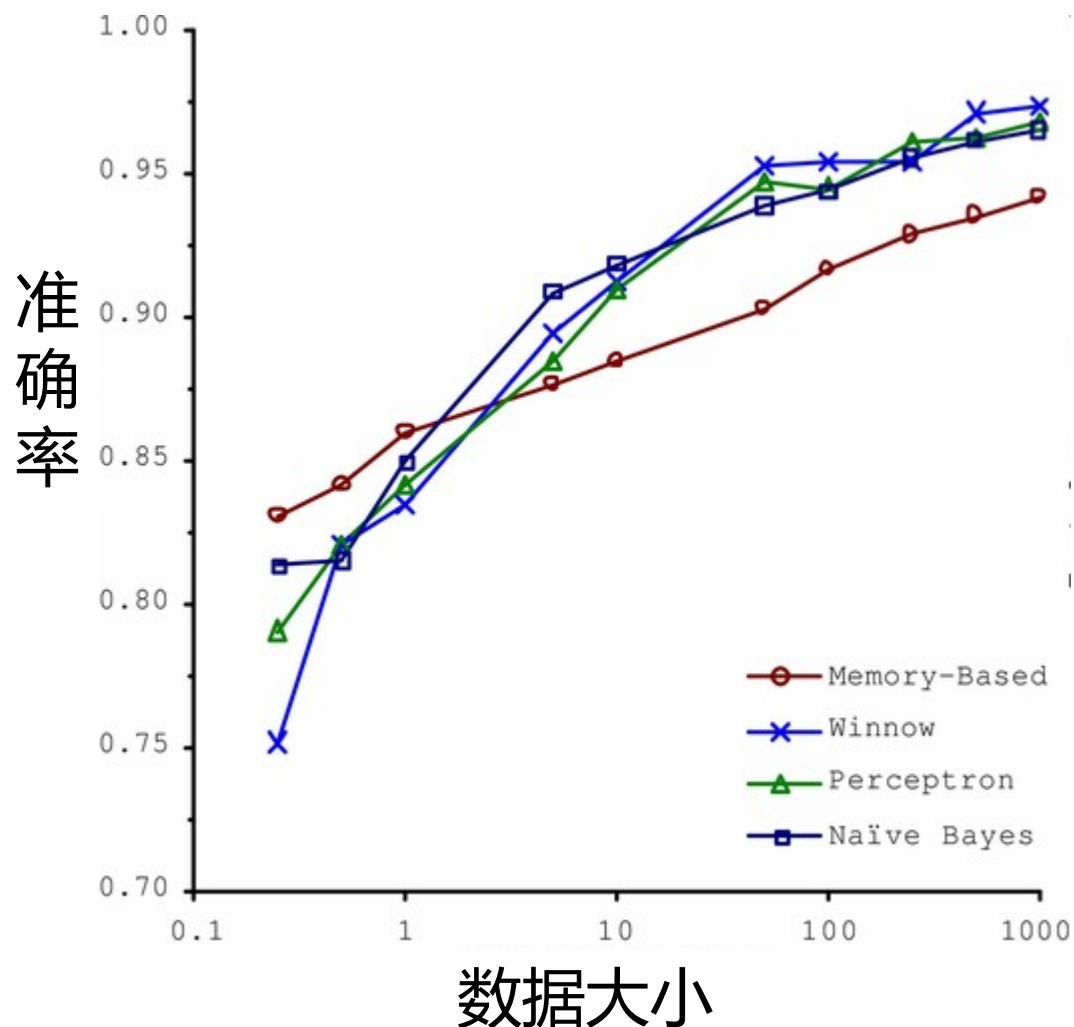
## 4. 集成学习方法

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

25



通过这张图可以看出，各种不同算法在输入的数据量达到一定级数后，都有相近的高准确度。于是诞生了机器学习界的名言：

**成功的机器学习应用不是拥有最好的算法，而是拥有最多的数据！**

# 欠拟合的处理

26

## 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，在神经网络模型中增加网络层数或神经元个数等。

## 3. 减小正则化系数

正则化是用来防止过拟合的，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系数。

# 正则化

27

**L1正则化** :  $J(w) = \frac{1}{2} \sum_{i=1}^m (h(x^{(i)}) - y^{(i)})^2 + \lambda \sum_{j=1}^n |w_j|$  , Lasso Regression ( Lasso回归 )

**L2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** :  $J(w) = \frac{1}{2} \sum_{i=1}^m (h(x^{(i)}) - y^{(i)})^2 + \lambda(\rho \cdot \sum_{j=1}^n |w_j| + (1 - \rho) \cdot \sum_{j=1}^n w_j^2)$   
( 弹性网络 )

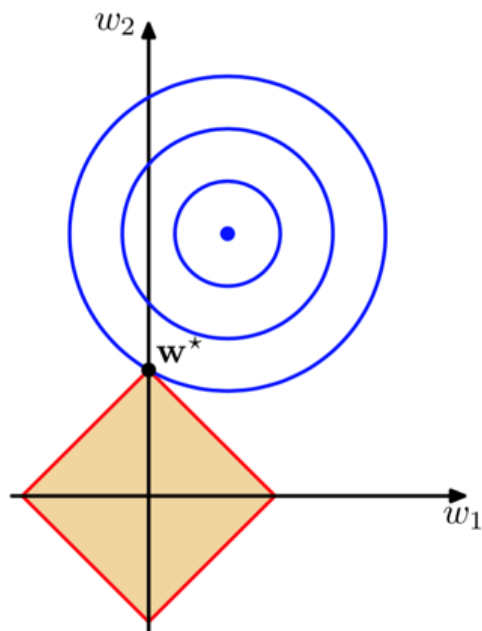


其中：

- $\lambda$ 为正则化系数，调整正则化项与训练误差的比例， $\lambda > 0$ 。
- $1 \geq \rho \geq 0$ 为比例系数，调整L1正则化与L2正则化的比例。

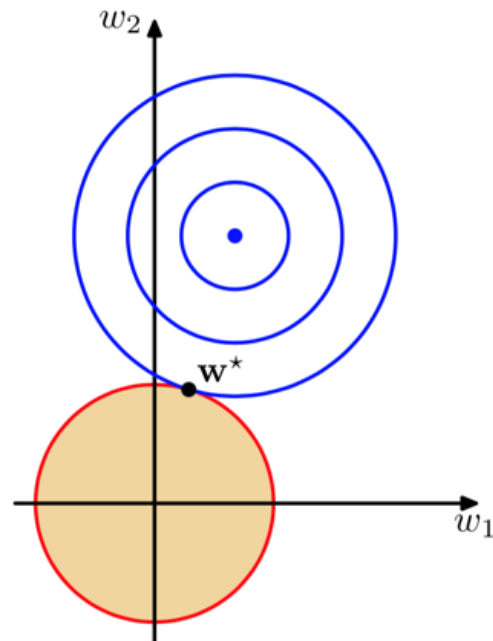
# 正则化

28



L1正则化是指在损失函数中加入权值向量 $w$ 的绝对值之和，L1的功能是使权重稀疏

## L1正则化可以产生稀疏模型



在损失函数中加入权值向量 $w$ 的平方和，L2的功能是使权重平滑。

## L2正则化可以防止过拟合

图上面中的蓝色轮廓线是没有正则化损失函数的等高线，中心的蓝色点为最优解，左图、右图分别为L1、L2正则化给出的限制。

可以看到在正则化的限制之下，L1正则化给出的最优解 $w^*$ 是使解更加靠近原点，也就是说L2正则化能降低参数范数的总和。

L1正则化给出的最优解 $w^*$ 是使解更加靠近某些轴，而其它的轴则为0，所以L1正则化能使得到的参数稀疏化。

## 4. 回归的评价指标

29

**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

30

**均方误差** ( Mean Square Error,MSE )

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

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

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

**平均绝对误差** ( Mean Absolute Error,MAE )

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

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

31

**R方** [*RSquared(r2score)*]

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

$$= 1 - \frac{SSE}{SST}$$

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

$$= 1 - \frac{MSE}{Var}$$

越接近于1,说明模型拟合得越好

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

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

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

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。

- [1] Andrew Ng. Machine Learning[EB/OL]. StanfordUniversity,2014.<https://www.coursera.org/course/ml>
- [2] 李航. 统计学习方法[M]. 北京: 清华大学出版社,2019.
- [3] 周志华. 机器学习[M]. 北京: 清华大学出版社,2016.
- [4] WEINBERGER K. Distance metric learning for large margin nearest neighbor classification[J]. Advances in Neural Information Processing Systems, 2006, 18.
- [5] HOERL A E, KENNARD R W. Ridge regression: applications to nonorthogonal problems[J]. Technometrics, 1970, 12(1): 69–82.
- [6] TIBSHIRANI R. Regression selection and shrinkage via the lasso[J]. Journal of the Royal Statistical Society Series B, 1996, 58(1): 267–288.
- [7] TIBSHIRANI R, BICKEL P, RITOV Y, et al. Least absolute shrinkage and selection operator[J]. Software: <http://www.stat.stanford.edu/tibs/lasso.html>, 1996.