机器学习
Machine Learning

北京航空航天大学计算机学院

School of Computer Science and Engineering, Beihang University

刘庆杰 陈佳鑫

2025年春季学期

Spring 2025

部分内容来源于C. Bishop和A. NG等人的课程以及互联网资源

第四章 正则化与稀疏学习

Chapter 4: Regularization and Sparse Learning

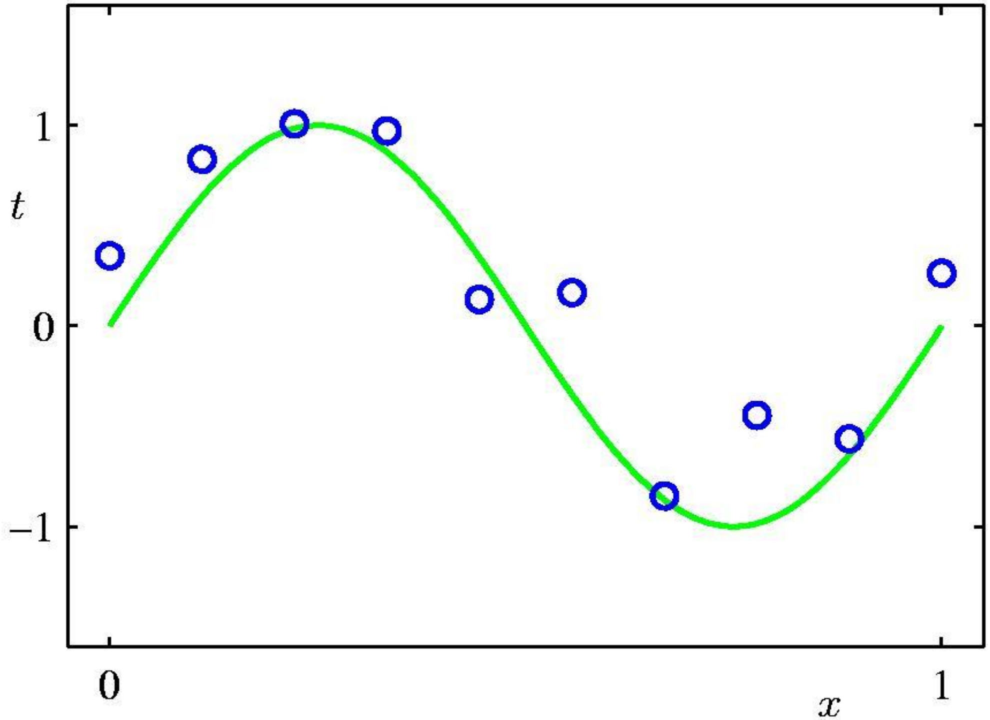
为什么要引入正则化?

- 过拟合问题回顾
- 什么是正则化?
- 正则化的形式

过拟合问题回顾

- **曲线拟合**

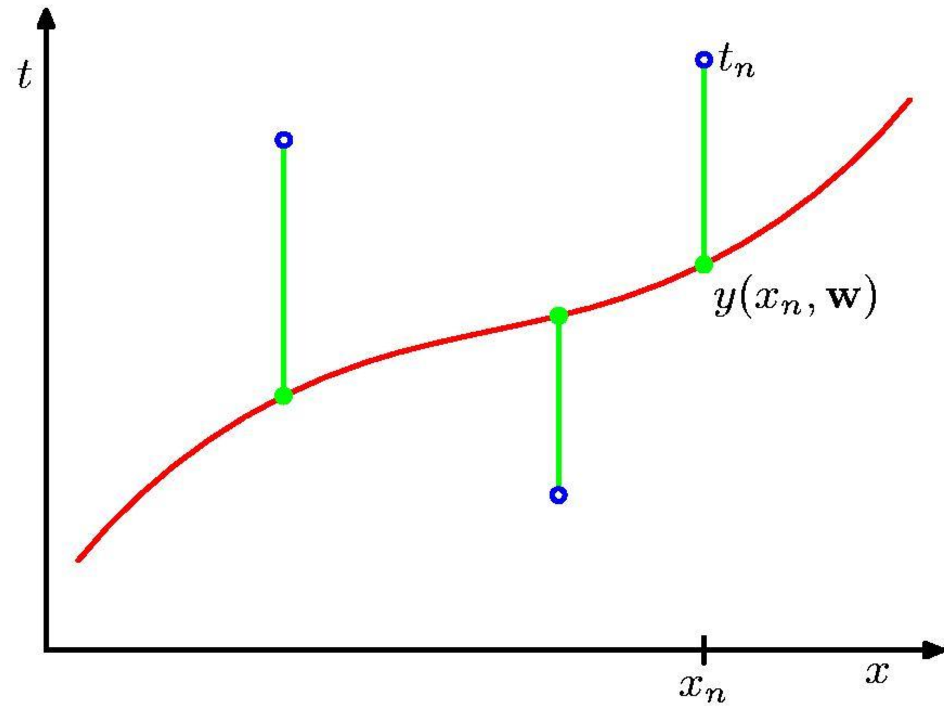
$$\sin(2\pi x)$$



$$y(x, \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + \ldots + w_M x^M = \sum_{j=0}^{M} w_j x^j$$

过拟合问题回顾

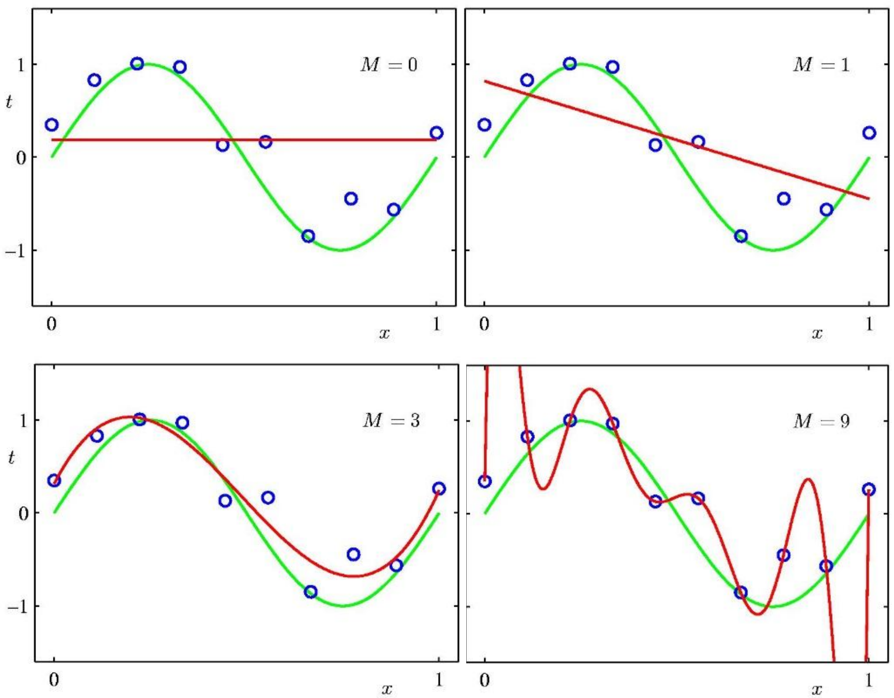
- 平方误差函数



$$E(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{y(x_n, \mathbf{w}) - t_n\}^2$$

过拟合问题回顾

- 多项式系数



	$M=0$	$M=1$	$M=3$	$M=9$
$w_0^\star$	0.19	0.82	0.31	0.35
$w_1^\star$		-1.27	7.99	232.37
$w_2^\star$			-25.43	-5321.83
$w_3^\star$			17.37	48568.31
$w_4^\star$				-231639.30
$w_5^\star$				640042.26
$w_6^\star$				-1061800.52
$w_7^\star$				1042400.18
$w_8^\star$				-557682.99
$w_9^\star$				125201.43

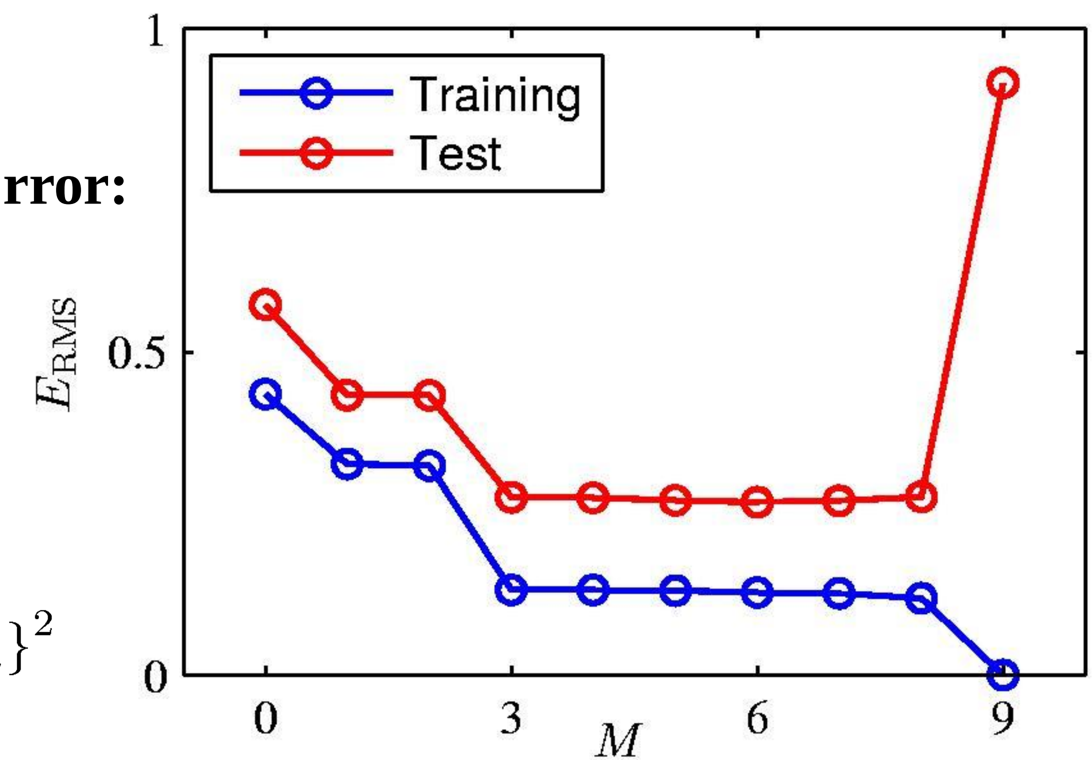
过拟合问题回顾

- **过拟合问题**

Root-Mean-Square (RMS) Error:

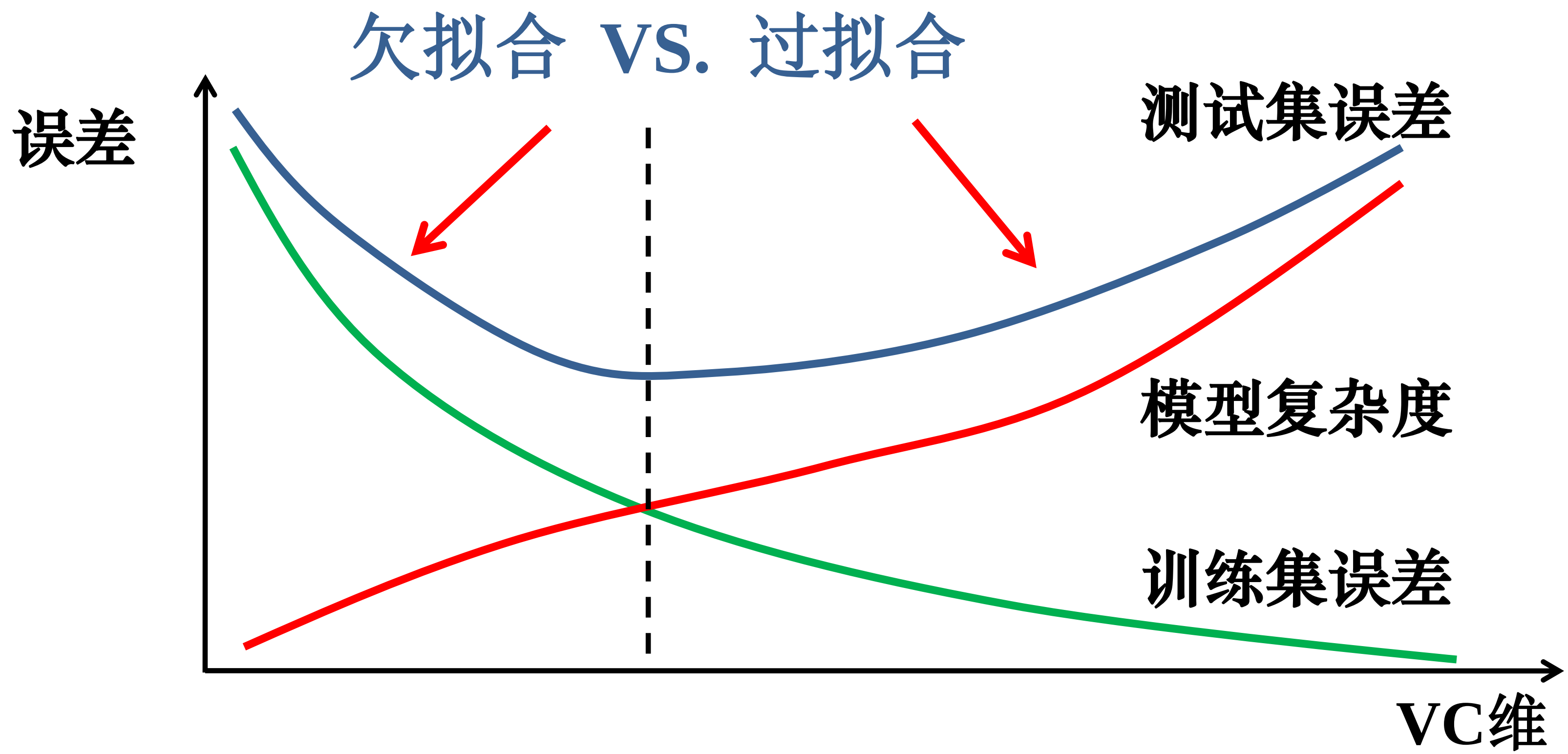
$$E(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{y(x_n,\mathbf{w}) - t_n\}^2$$

$$E_{\mathrm{RMS}} = \sqrt{2E(\mathbf{w}^\star)/N}$$



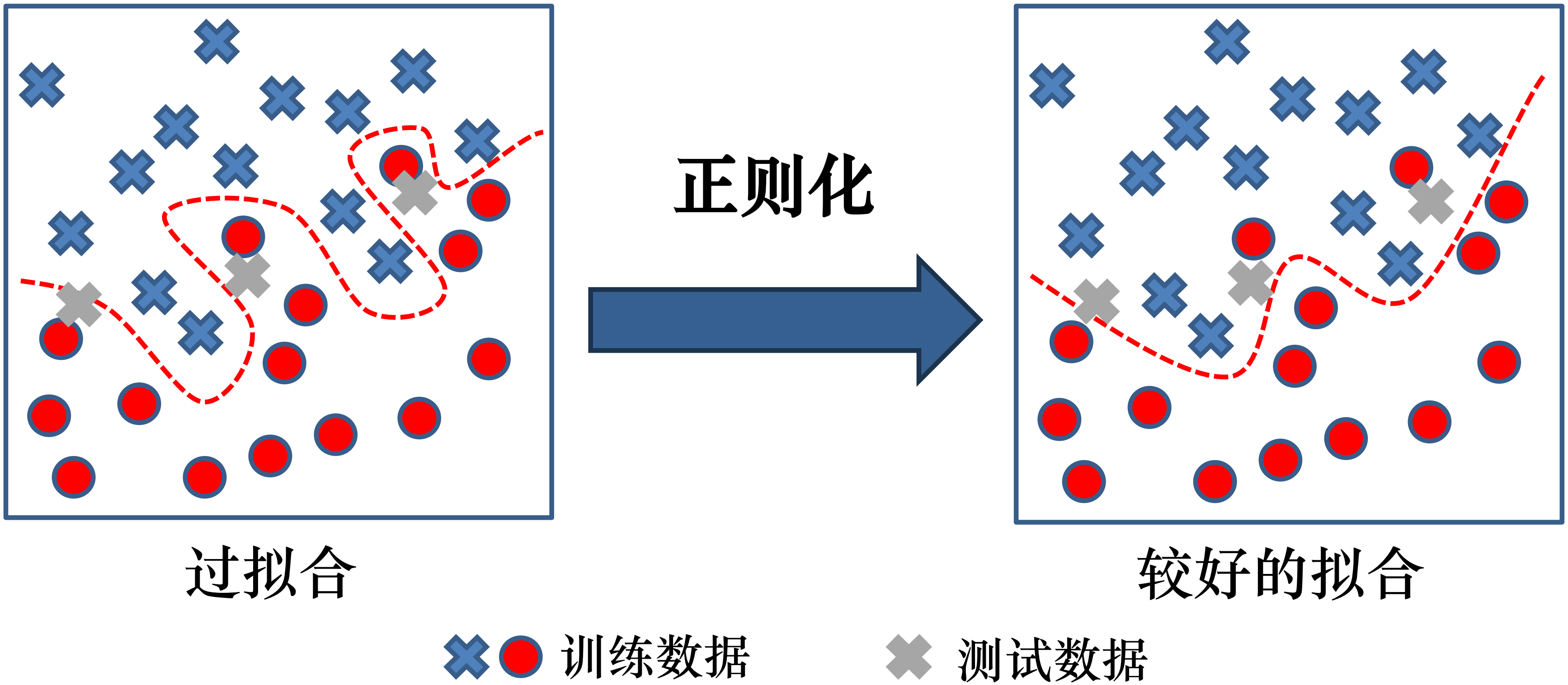
过拟合问题回顾

- **过拟合现象**
  - 模型复杂度高，在训练时过度拟合了训练集，但在测试集上却表现很差（泛化性差），具有高方差、低偏差的特点。



什么是正则化?

- **正则化（Regularization）是指为缓解过拟合而加入额外先验约束的过程。训练机器学习模型时通过引入正则化来增强模型的泛化性能。**



什么是正则化?

- 机器学习训练过程可以简化为训练集D损失函数$L(F)$的最小化问题，为了对抗过拟合，我们向损失函数中加入描述模型复杂程度的正则化项$\Omega(F)$，其一般形式为：

$$F^* = \arg\min_F l(F) \quad \Rightarrow \quad F^* = \arg\min_F \left(\underbrace{\boxed{l(F)}}_{\text{训练集损失}} + \underbrace{\boxed{\gamma}}_{\text{正则化系数}} \underbrace{\boxed{\Omega(F)}}_{\text{正则化项}} \right), \gamma > 0$$

- 正则化项是在模型训练过程中引入了**模型参数的先验约束**。通过引入表示模型复杂度的正则化项，降低模型的复杂度，提高泛化性能。

正则化的形式

- **L_p-范数正则化**
  - 机器学习中广泛使用的正则化形式
  - 计算高效，目标函数可用梯度下降等方式求解最优化问题
  - L_p-范数表示**向量空间中的距离**，具备非负性、齐次性、三角不等式的特性

$$L_p(\vec{x}) = \| \vec{x} \|_p = \left(\sum_{i=1}^{n} |x_i|^p \right)^{1/p}, \vec{x} = \{x_1, x_2, \cdots, x_n\}, p \geq 1$$

非负性： $\| \vec{x} \|_p \geq 0$

齐次性： $\| \alpha \vec{x} \|_p = \alpha \| \vec{x} \|_p, \alpha > 0$

三角不等式（次可加性）： $\| \vec{x} + \vec{y} \|_p \leq \| \vec{x} \|_p + \| \vec{y} \|_p$

正则化的形式

- L_p-范数正则化项引入L_p-范数作为正则化项$\Omega(F)$，约束了**模型参数的范数上界**，防止模型过度拟合到训练集D的数据。通过引入L_p-范数正则化来约束模型的复杂度，提高了模型的泛化能力。

- 常见方法：L_2/L_1正则化
  - L_2正则化（$p=2$）：$\|\vec{x}\|_2=\sqrt{\sum_{i=1}^{n}|x_i|^2}$，即欧氏距离
  - L_1正则化（$p=1$）：$\|\vec{x}\|_1=\sum_{i=1}^{n}|x_i|$，即曼哈顿距离

L_2正则化

- L_2正则化的定义与求解
- 岭回归
- 权重衰减
- L_2正则化的几何理解

L_2正则化的定义

- **L_2正则化的定义**

  - **正则化的一般形式**

$$Obj\,(F) = \boxed{l(F)} + \boxed{\gamma}\boxed{\Omega(F),} \qquad \gamma > 0$$

训练集损失　　正则化系数　　正则化项

  - **L_2正则化的定义**

$$L(\boldsymbol{w}) = \boxed{l(\boldsymbol{w})} + \boxed{\frac{\lambda}{2}}\boxed{||\boldsymbol{w}||_2^2}$$

即在损失函数中添加一个参数的L_2范数惩罚项

  - $\gamma = \frac{\lambda}{2}$，$\lambda > 0$，$\lambda$为**正则化系数**，用于控制正则项对总损失的贡献程度。λ越大，模型权重越趋近 0

L_2正则化问题的求解

● 问题求解

➢ 目标函数

$$L(\boldsymbol{w}) = l(\boldsymbol{w}) + \frac{\lambda}{2}||\boldsymbol{w}||_2^2$$

➢ 求L对w的偏导数

$$\nabla_{\boldsymbol{w}} L(\boldsymbol{w}) = \nabla_{\boldsymbol{w}} l(\boldsymbol{w}) + \lambda \boldsymbol{w}$$

➢ 梯度下降更新

学习率

$$\begin{aligned}
\boldsymbol{w}_{t+1} &= \boldsymbol{w}_t - \boxed{\eta} * \nabla_{\boldsymbol{w}} L(\boldsymbol{w}) \\
&= \boldsymbol{w}_t - \eta(\nabla_{\boldsymbol{w}} l(\boldsymbol{w}) + \lambda \boldsymbol{w}) \\
&= (1 - \eta\lambda)\boldsymbol{w}_t - \eta \nabla_{\boldsymbol{w}} l(\boldsymbol{w})
\end{aligned}$$

L_2正则化问题的求解

- 正则化系数λ确定
  - 将数据集分成训练集和验证集
  - 在训练集上训练模型,并在验证集上评估模型的性能
  - 选择在验证集上性能最好的参数λ值作为最终的参数设置

	$\ln\lambda = -\infty$	$\ln\lambda = -18$	$\ln\lambda = 0$
$w_0^\star$	0.35	0.35	0.13
$w_1^\star$	232.37	4.74	-0.05
$w_2^\star$	-5321.83	-0.77	-0.06
$w_3^\star$	48568.31	-31.97	-0.05
$w_4^\star$	-231639.30	-3.89	-0.03
$w_5^\star$	640042.26	55.28	-0.02
$w_6^\star$	-1061800.52	41.32	-0.01
$w_7^\star$	1042400.18	-45.95	-0.00
$w_8^\star$	-557682.99	-91.53	0.00
$w_9^\star$	125201.43	72.68	0.01

岭回归(Ridge Regression)

- 岭回归：在线性回归的代价函数中加入L_2正则化项

$$L(\boldsymbol{w}) = \frac{1}{2}\sum_{i=1}^{n}((\boldsymbol{w}^T\boldsymbol{x}_i + b) - y_i)^2 + \boxed{\frac{\lambda}{2}||\boldsymbol{w}||_2^2}$$

- 将目标函数写成矩阵形式：

$$\begin{aligned} L(\boldsymbol{w}) &= \frac{1}{2}(\boldsymbol{X}\boldsymbol{w} - \boldsymbol{Y})^T(\boldsymbol{X}\boldsymbol{w} - \boldsymbol{Y}) + \frac{\lambda}{2}\boldsymbol{w}^T\boldsymbol{w} \\ &= \frac{1}{2}(\boldsymbol{w}^T\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{w} - \boldsymbol{w}^T\boldsymbol{X}^T\boldsymbol{Y} - \boldsymbol{Y}^T\boldsymbol{X}\boldsymbol{w} + \boldsymbol{Y}^T\boldsymbol{Y}) + \frac{\lambda}{2}\boldsymbol{w}^T\boldsymbol{w} \end{aligned}$$

- 求导： $\nabla_{\boldsymbol{w}} L(\boldsymbol{w}) = \boldsymbol{X}^T\boldsymbol{X}\boldsymbol{w} - \boldsymbol{X}^T\boldsymbol{Y} + \lambda\boldsymbol{w}$

- 得解： $\boldsymbol{w} = (\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{I})^{-1}\boldsymbol{X}^T\boldsymbol{Y}$

- 此外还可以用梯度下降法求解

岭回归(Ridge Regression)

- **示例：房价估计**
  - **问题：使用卧室数量、房屋面积等12个特征对房价预测，对比训练后两个模型的参数情况**
  - **建模：** $\min_{\boldsymbol{w}} \frac{1}{2}\|\boldsymbol{X}\boldsymbol{w}-\boldsymbol{y}\|_2^2+\frac{\lambda}{2}\|\boldsymbol{w}\|_2^2$
  - **求解：使用梯度下降法求解**
  - **性质特点：可以看出岭回归通过添加平方项来收缩系数，但不会将它们缩减至零**

	线性回归	岭回归 ($\boldsymbol{\lambda}$=1)
$\boldsymbol{w}_0^*$	-0.0561	-0.0549
$\boldsymbol{w}_1^*$	0.1108	0.1088
$\boldsymbol{w}_2^*$	0.0111	0.0084
$\boldsymbol{w}_3^*$	0.0914	0.0914
$\boldsymbol{w}_4^*$	-0.2092	-0.2060
$\boldsymbol{w}_5^*$	0.3546	0.3545
$\boldsymbol{w}_6^*$	-0.0187	-0.0181
$\boldsymbol{w}_7^*$	-0.3174	-0.3140
$\boldsymbol{w}_8^*$	0.2629	0.2534
$\boldsymbol{w}_9^*$	-0.2494	-0.2404
$\boldsymbol{w}_{10}^*$	-0.2072	-0.2064
$\boldsymbol{w}_{11}^*$	0.0897	0.0897
$\boldsymbol{w}_{12}^*$	-0.3526	-0.3512

*数据源：http://lib.stat.cmu.edu/datasets/boston

权重衰减

- L_2正则化又被称为权重衰减（Weight Decay）
  - 对于参数更新的变化

$$\boldsymbol{w}_{t+1} = (1-\eta\lambda)\boldsymbol{w}_t - \eta\nabla_{\boldsymbol{w}} l(\boldsymbol{w})$$

包含权重衰减（L_2正则化）的参数更新

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \eta\nabla_{\boldsymbol{w}} l(\boldsymbol{w})$$

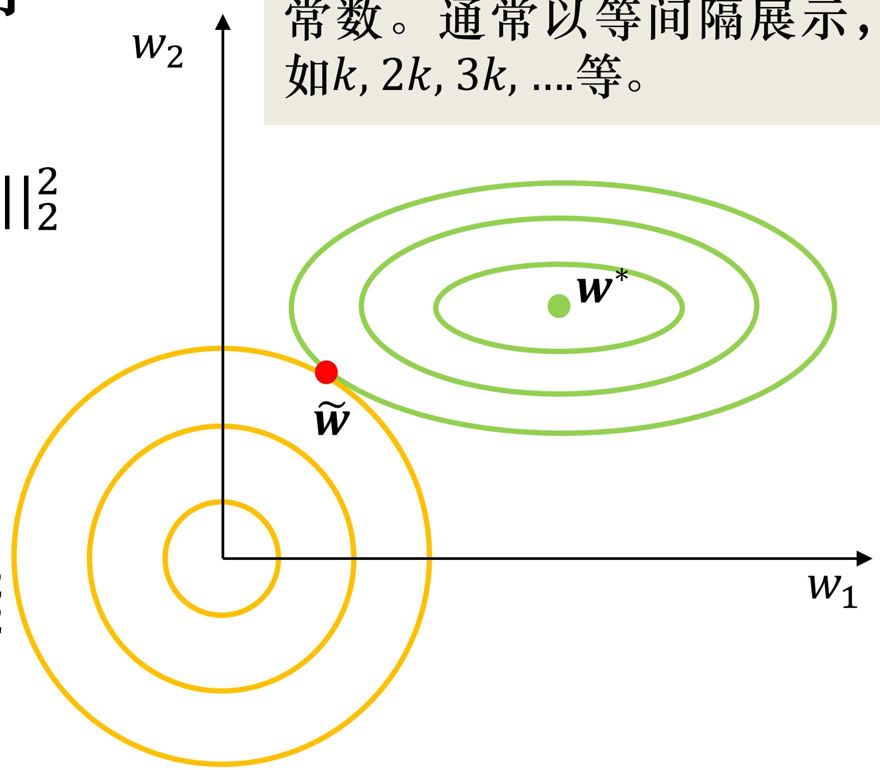
原始参数更新

加入权重衰减（L_2正则化）后会引起学习规则的修改，即在每步执行通常的梯度更新之前先**收缩权重向量**（将权重向量乘以一个常数因子）

L_2正则化的几何理解

- **以二维空间的线性模型为例**
  - **绿线：原始损失函数$l(\boldsymbol{w})$的等值线**
  - **橘线：L_2正则化约束限制$||\boldsymbol{w}||_2^2$**
  - **$\boldsymbol{w}^*$:原最优解**
    $$\boldsymbol{w}^* = \arg\min_{\boldsymbol{w}} l(\boldsymbol{w})$$
  - **$\widetilde{\boldsymbol{w}}$:加入L_2正则项后的最优解**
    $$\widetilde{\boldsymbol{w}} = \arg\min_{\boldsymbol{w}} l(\boldsymbol{w}) + \frac{\lambda}{2}||\boldsymbol{w}||_2^2$$

等值线：对于函数 $f(x,y)$，等值线为所有满足$f(x,y)=k$的点 (x,y)的集合，其中 k 为常数。通常以等间隔展示，如k, $2k$, $3k$, ...等。



L_2正则化的几何理解

- 以二维空间的线性模型为例
  - 已知目标函数

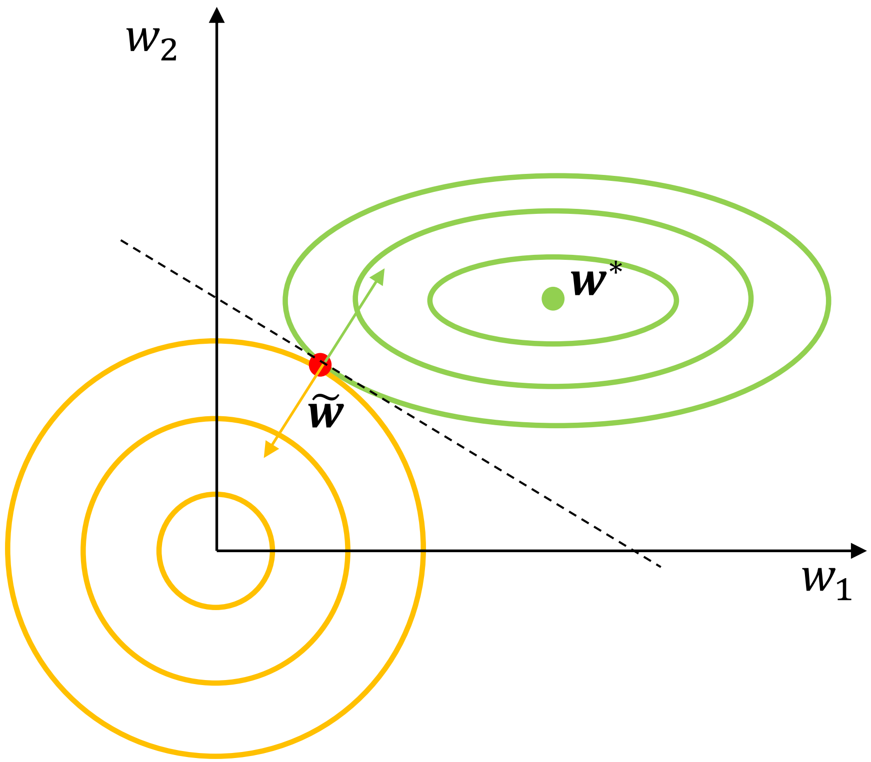
$$L(\boldsymbol{w}) = l(\boldsymbol{w}) + \frac{\lambda}{2}||\boldsymbol{w}||_2^2$$

  - 求最优解即取梯度为0

$$\nabla_{\boldsymbol{w}} L(\boldsymbol{w}) = \nabla_{\boldsymbol{w}} l(\boldsymbol{w}) + \lambda \boldsymbol{w} = 0$$

  - 原损失函数与L_2正则化约束项只有在切点处的梯度方向平行，才能达到相加为0

$\tilde{\boldsymbol{w}}$位于原损失函数与L_2正则化约束项等值线的切点



L_2正则化的几何理解

- **权重衰减的特点：**
  - 当 w^*沿着w_1方向移动时，$l(w)$变化不显著
    当 w^*沿着w_2方向移动时，$l(w)$变化显著
  - w^*在w_1方向的衰减较大
    w^*在 w_2方向的衰减较小

L₂正则化可以看作是**拟合训练数据**和**对小权重的偏好**之间的**权衡**

⇩

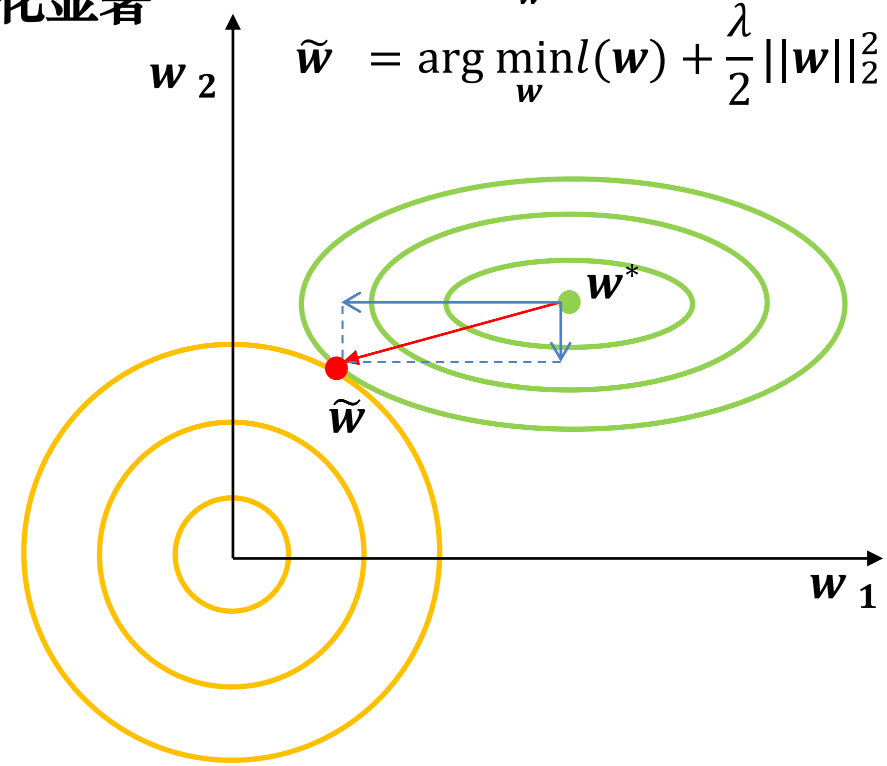
在权重衰减的同时，尽可能保证原损失函数变化小

⇩

在**$l(w)$变化显著的方向**上的参数**衰减较小**；在**$l(w)$变化不显著的方向**对应的参数会**衰减较大**

$$\boldsymbol{w}^* = \arg\min_{\boldsymbol{w}} l(\boldsymbol{w})$$

$$\widetilde{\boldsymbol{w}} = \arg\min_{\boldsymbol{w}} l(\boldsymbol{w}) + \frac{\lambda}{2}||\boldsymbol{w}||_2^2$$



L_1正则化

- 特征选择
- 特征选择与L_0正则化
- L_1正则化的定义
- L_1正则化的求解
- L_1正则化与稀疏解

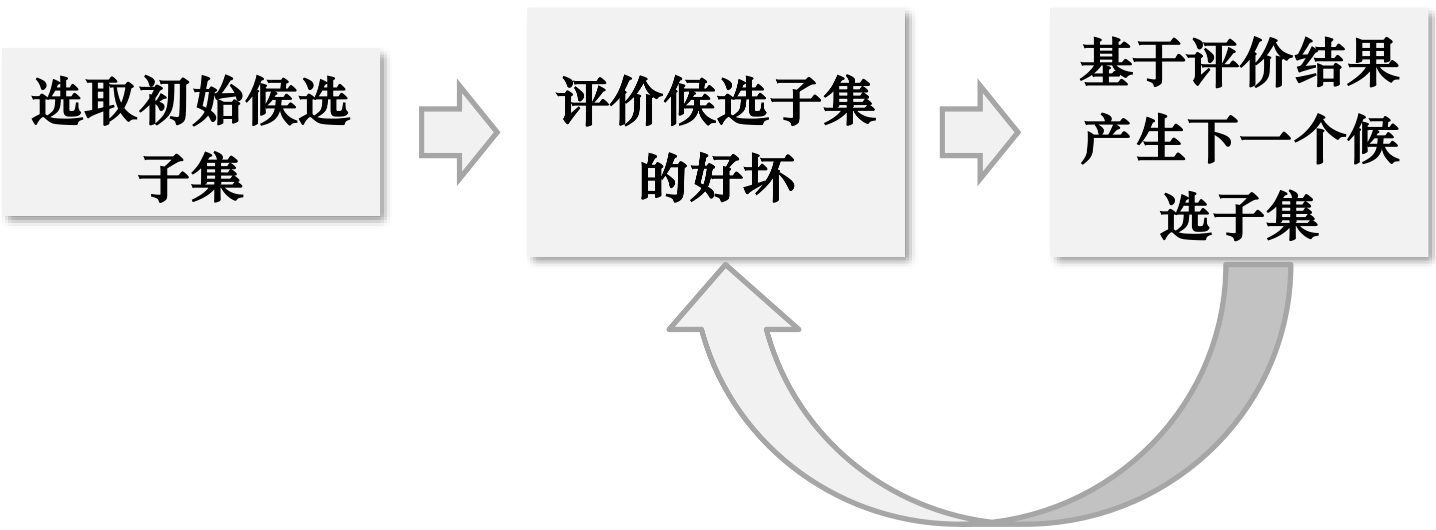
特征选择

- 对于高维特征场景，L_2正则化不适用
  - 基因表达数据通常具有非常高的维度，将数以万计的基因表达水平作为特征。大部分基因表达冗余，需要从中**提取少数关键**的基因特征来预测疾病，即要在加入正则项减少过拟合的同时，进行**特征选择**与**提升计算效率**
  - L_2正则化使权重减小得更均匀，而不是将它们降为0，即所有特征都会被保留

需要将大部分冗余的**参数设置为0**，使模型能保留关键特征并减小模型复杂度

特征选择的一般方法

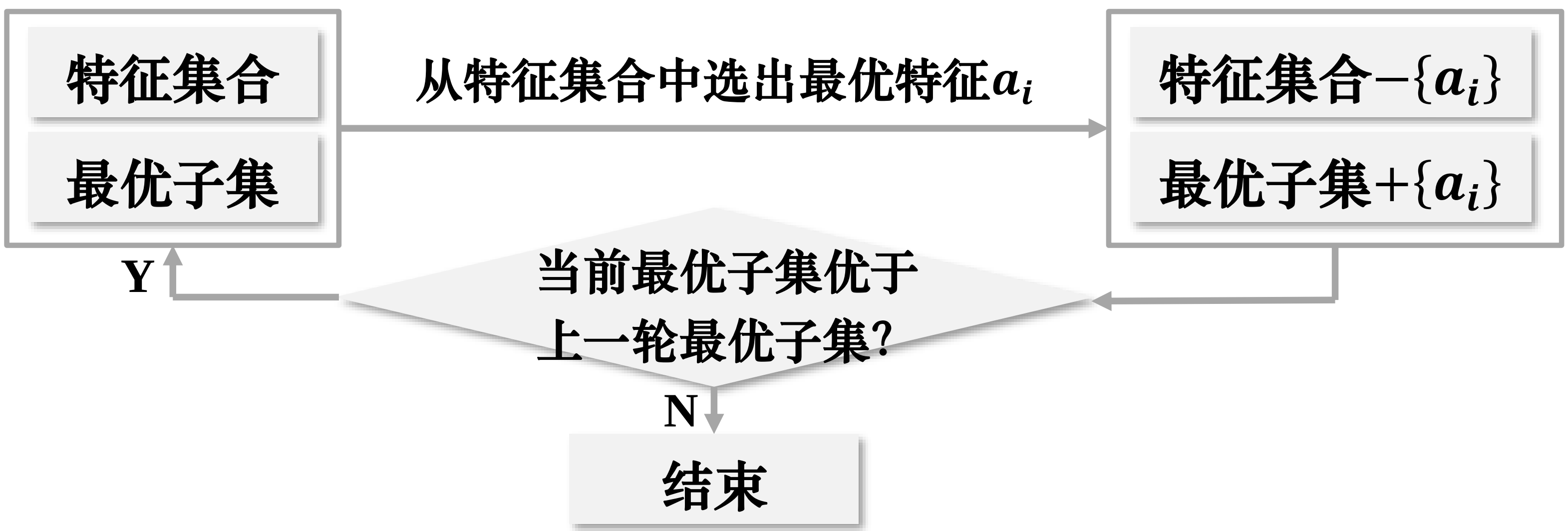
- 特征选择的一般方法
  - 遍历所有可能的子集
    - 计算上面临指数级计算复杂度，不可行
  - 可行方法



两个关键环节：子集搜索和子集评价

特征选择的一般方法

- 子集搜索
  - 用贪心策略选择包含重要信息的特征子集
  - 前向搜索：逐渐增加相关特征
    - 最优子集初始为空集，特征集合初始时包括所有给定特征



  - 后向搜索：从完整的特征集合开始，逐渐减少无关特征
  - 双向搜索：每一轮逐渐增加相关特征，同时减少无关特征

特征选择的一般方法

- 子集评价
  - 特征子集确定了对数据集的一个划分（分类）
    - 每个划分区域对应着特征子集的某种取值
  - 样本标记对应着对数据集的真实划分（分类）

通过估算两个划分的差异，就能对特征子集进行评价；与样本标记对应的划分的差异越小，说明当前特征子集越好

特征选择的一般方法

- 用信息熵进行子集评价
  - 特征子集A确定了对数据集D的一个划分
    - A上的取值将数据集D分为V份，每一份用D^v表示
    - $\mathrm{Ent}(D^v)$表示D^v上的信息熵
  - 样本标记Y对应着对数据集D的真实划分
    - $\mathrm{Ent}(D)$表示D上的信息熵

D上的信息熵定义为

$\mathrm{Ent}(D)=\sum_{i=1}^{|Y|}p_i\log_2 p_i$

第i类样本所占比例为p_i

特征子集A的信息增益为

$$\mathrm{Gain}(A)=\mathrm{Ent}(D)-\sum_{v=1}^{V}\frac{|D^v|}{|D|}\mathrm{Ent}(D^v)$$

特征选择的一般方法

- **常见的特征选择方法**
  - 将特征子集搜索机制与子集评价机制相结合，即可得到特征选择方法
  - 常见的特征选择方法大致分为如下三类：
    - 过滤式
    - 包裹式
    - 嵌入式

特征选择的一般方法

- **过滤式选择**
  - 先用特征选择过程过滤原始数据，再用过滤后的特征来训练模型；特征选择过程与后续学习器无关
  - Relief (Relevant Features)方法【Kira and Rendell, 1992】
    - 为每个初始特征赋予一个“相关统计量”，度量特征的重要性
    - 特征子集的重要性由子集中每个特征所对应的相关统计量之和决定
    - 设计一个阈值，然后选择比阈值大的相关统计量分量所对应的特征
    - 或者指定欲选取的特征个数，然后选择相关统计量分量最大的指定个数特征

如何确定相关统计量?

特征选择的一般方法

- **Relief方法中相关统计量的确定**
  - 猜中近邻 (near-hit)： x_i的同类样本中的最近邻$x_{i,nh}$
  - 猜错近邻 (near-miss)： x_i的异类样本中的最近邻$x_{i,nm}$
  - 相关统计量对应于属性的分量为

$$\delta^j=\sum_i -\mathrm{diff}\left(x_i^j,x_{i,nh}^j\right)^2+\mathrm{diff}\left(x_i^j,x_{i,nm}^j\right)^2$$

若j为离散型，则$x_a^j=x_b^j$时$\mathrm{diff}\left(x_a^j,x_b^j\right)=0$，否则为1；若$j$为连续型，则$\mathrm{diff}\left(x_a^j,x_b^j\right)=\left|x_a^j-x_b^j\right|$，注意$x_a^j,x_b^j$已规范到$[0,1]$区间

  - 相关统计量越大，属性j上，猜中近邻比猜错近邻越近，即属性j对区分对错越有用
  - Relief方法的时间开销随采样次数以及原始特征数线性增长，运行效率高

特征选择的一般方法

● Relief方法的多类扩展

- Relief方法是为二分类问题涉及的，其扩展变体Relief-F【Kononenko, 1994】能处理多分类问题
- 数据集中的样本来自$|y|$个类别，其中x_i属于第k类
- 猜中近邻：第k类中x_i的最近邻$x_{i,nh}$
- 猜错近邻：第k类之外的每个类中找到一个x_i的最近邻作为猜错近邻，记为$x_{i,l,nm}(l = 1, 2, \cdots, |y|; l \neq k)$
- 相关统计量对应于属性的分量为

$$\delta^j = \sum_i -\mathrm{diff}\left(x_i^j, x_{i,nh}^j\right)^2 + \sum_{l \neq k}\left(p_l \times \mathrm{diff}\left(x_i^j, x_{i,l,nm}^j\right)^2\right)$$

p_l为第l类样本在数据集D中所占的比例

特征选择的一般方法

- 包裹式选择
  - 包裹式选择直接把最终将要使用的学习器的性能作为特征子集的评价准则
    - 包裹式特征选择的目的就是为给定学习器选择最有利于其性能、“量身定做”的特征子集
    - 包裹式选择方法直接针对给定学习器进行优化，因此从最终学习器性能来看，包裹式特征选择比过滤式特征选择更好
    - 包裹式特征选择过程中需多次训练学习器，计算开销通常比过滤式特征选择大

特征选择的一般方法

● LVW包裹式特征选择方法

- LVW (Las Vegas Wrapper) 【Liu and Setiono, 1996】在拉斯维加斯方法框架下使用随机策略来进行子集搜索，并以最终分类器的误差作为特征子集评价准则
- 基本步骤

过程：

步骤1：在循环的每一轮随机产生一个特征子集；

步骤2：在随机产生的特征子集上通过交叉验证推断当前特征子集的误差；

步骤3：进行多次循环，在多个随机产生的特征子集中选择误差最小的特征子集作为最终解*；

*若运行时间限制，则该算法有可能给不出解

特征选择的一般方法

- **嵌入式选择**
  - ➢ 嵌入式特征选择是将特征选择过程与学习器训练过程融为一体，两者在同一个优化过程中完成，在学习器训练过程中自动地进行特征选择
    - ■嵌入式特征选择过程中不需要多次训练学习器，计算开销通常比包裹式特征选择小
    - ■代表性方法：L1正则化

L_0正则化

- **L_0正则化的定义**
  - **L_0范数：** $\|x\|_0=\sum_{i=1}^{n}\mathbb{I}_{x_i\neq 0}$，**即向量中非0元素个数**
  - **对于一般模型，它的L_0正则化可以表示**

$$\mathrm{Obj}(F)=l(F)+\gamma\Omega(F)$$

$$L(\boldsymbol{w})=l(\boldsymbol{w})+\boxed{\lambda\|\boldsymbol{w}\|_0}$$

L_0正则化项

  - **优化问题可表达为** $\boxed{\min_{\boldsymbol{w}} l(\boldsymbol{w})+\lambda\|\boldsymbol{w}\|_0}$ **NP难，难求解**

由于L_0范数本身**离散非凸**，导致优化目标离散和非凸的，使得优化问题变得十分复杂

  - **可以用L_1范数近似代替L_0范数**

范数

- 为什么能使用L_1范数近似替代L_0范数

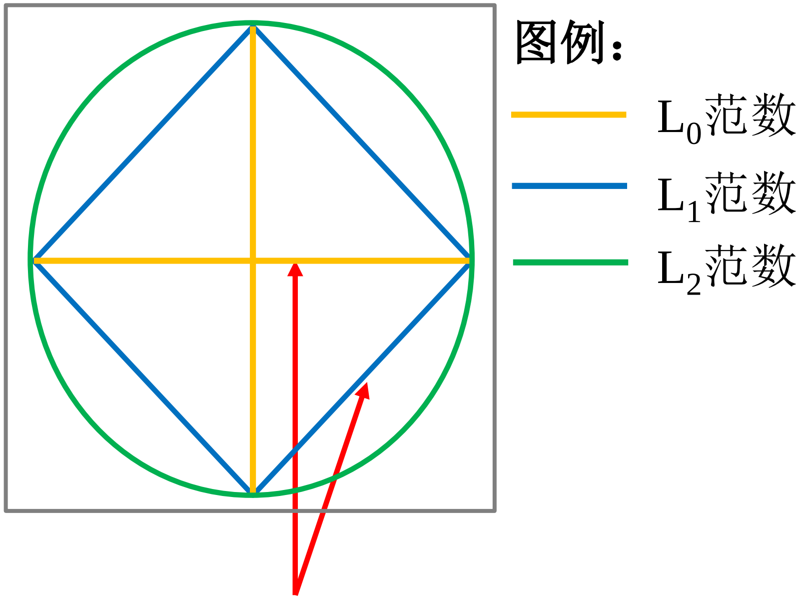
➢ **L_1-范数**：$\|\boldsymbol{x}\|_1 = \sum_{i=1}^{n} |x_i|$

L_1范数被称作曼哈顿距离(Manhattan distance)，**连续且凸**，但**在零点不可导**

➢ **L_0-范数**：$\|\boldsymbol{x}\|_0 = \sum_{i=1}^{n} \mathbb{I}_{x_i \neq 0}$

L_0范数实际上不是一个真正的数学范数，因为它不满足次可加性，它**非凸**且**不连续**

➢ **L_1范数是L_0范数的最优凸近似**



在图中L_1是能包住L_0的最小凸的图形

L_1是L_0的最优凸近似证明：Best Convex Lower Approximations of the L_0 Pseudonorm on Unit Balls

L_1是L_0的最优凸近似几何解释： Why L_1 is a good approximation to L_0 : A geometric explanation

L_1正则化

- **L_1正则化的定义**
  - **对一般模型，L_1正则化问题可以写为**

$$\mathrm{Obj}(F) = l(F) + \gamma\Omega(F)$$

$$L(\boldsymbol{w}) = l(\boldsymbol{w}) + \boxed{\lambda\|\boldsymbol{w}\|_1}$$

L_1正则化项，$\lambda>0$

  - **例如，带L_1正则项的线性回归模型(Lasso回归)的损失函数**

$$\min_{\boldsymbol{w}} \frac{1}{2}\|\boldsymbol{X}\boldsymbol{w} - \boldsymbol{y}\|_2^2 + \lambda\|\boldsymbol{w}\|_1$$

  - **该问题无法直接使用梯度下降方法求解**

目标函数为同时包含可微分项与不可微分项时，可以使用近端梯度下降(PGD)来求解

L_1正则化的求解

- **近端梯度下降(PGD)**
  - 近端梯度下降一般是求解部分带有不可微函数的优化问题

$$\min_{\boldsymbol{x}} \underbrace{g(\boldsymbol{x})}_{\text{可微函数}} + \underbrace{h(\boldsymbol{x})}_{\text{不可微函数}}$$

  - 主要步骤

    步骤1. 可微部分优化：先求可微部分的解， 即先求$g(\boldsymbol{x})$

    步骤2. 近端映射：再根据可微部分的解更新整体，即再求$g(\boldsymbol{x}) + h(\boldsymbol{x})$

    步骤3. 迭代求解：迭代步骤1和步骤2，直至收敛或达到最大迭代次数

L_1正则化的求解

- **可微部分优化**
  - **对L_1正则化问题的优化目标** $\min_{\boldsymbol{w}} \boxed{l(\boldsymbol{w})} + \lambda\|\boldsymbol{w}\|_1$ (5.3.1)

    先考虑可微部分的迭代
  - **为了求$l(\boldsymbol{w})$的更新方向，对$l(\boldsymbol{w})$在$\boldsymbol{w}_k$附近通过二阶泰勒展开近似，这样可以利用函数的曲率信息来自适应地调整优化步长，以提高收敛速度和稳定性**

$$\begin{aligned}\hat{l}(\boldsymbol{w}) &\simeq l(\boldsymbol{w}_k) + \langle \nabla l(\boldsymbol{w}_k), \boldsymbol{w} - \boldsymbol{w}_k \rangle + \frac{\nabla^2 l(\boldsymbol{w}_k)}{2} \|\boldsymbol{w} - \boldsymbol{w}_k\|^2 \\ &\leq l(\boldsymbol{w}_k) + \langle \nabla l(\boldsymbol{w}_k), \boldsymbol{w} - \boldsymbol{w}_k \rangle + \frac{L}{2} \|\boldsymbol{w} - \boldsymbol{w}_k\|^2\end{aligned}$$

L-Lipschitz 条件 (5.3.2)

对可微部分$l(\boldsymbol{w})$需要满足函数变化不能有跳变，$\nabla l(\boldsymbol{w})$需满足L-Lipschitz条件，即存在常数L>0使得：

$$\|\nabla l(\boldsymbol{w}') - \nabla l(\boldsymbol{w})\|_2 \leq L\|\boldsymbol{w}' - \boldsymbol{w}\|_2 \qquad (\forall \boldsymbol{w}', \boldsymbol{w})$$

L_1正则化的求解

- **可微部分优化**

  - **将(5.3.2)合并展开**

$$l(\boldsymbol{w}_k)+\frac{L}{2}\left((\boldsymbol{w}-\boldsymbol{w}_k)+\frac{1}{L}\nabla l(\boldsymbol{w}_k)\right)^T\left((\boldsymbol{w}-\boldsymbol{w}_k)+\frac{1}{L}\nabla l(\boldsymbol{w}_k)\right)-\frac{1}{2L}\nabla l(\boldsymbol{w}_k)^T\nabla l(\boldsymbol{w}_k)$$

$$=\frac{L}{2}\left\|(\boldsymbol{w}-\boldsymbol{w}_k)+\frac{1}{L}\nabla l(\boldsymbol{w}_k)\right\|_2^2+C \qquad (5.3.3)$$

$l(\boldsymbol{w}_k)-\frac{1}{2L}\nabla l(\boldsymbol{w}_k)^T\nabla l(\boldsymbol{w}_k)$

视为常数项

  - **取(5.3.3)式的最小值，有**$\left\|(\boldsymbol{w}-\boldsymbol{w}_k)+\frac{1}{L}\nabla l(\boldsymbol{w}_k)\right\|_2^2=\mathbf{0}$

  - **那么**$l(\boldsymbol{w})$**的最小值，可由如下**$\boldsymbol{w}_{k+1}$**迭代求得**

$$\boldsymbol{w}_{k+1}=\boldsymbol{w}_k-\frac{1}{L}\nabla l(\boldsymbol{w}_k) \qquad (5.3.4)$$

L_1正则化的求解

- **近端映射**
  - 将通过梯度下降法对$l(\boldsymbol{w})$进行最小化的思想**带入整体优化问题(5.3.1)**

$$\boldsymbol{w}_{k+1} = \underset{\boldsymbol{w}}{\operatorname{argmin}} \frac{L}{2}\left\|(\boldsymbol{w}-\boldsymbol{w}_k)+\frac{1}{L}\nabla l(\boldsymbol{w}_k)\right\|_2^2 + \lambda\|\boldsymbol{w}\|_1 \quad (5.3.5)$$

即在每一步梯度下降迭代时，**考虑L_1范数最小化**

  - 对公式(5.3.5)，可以**先计算可微部分** $\boldsymbol{z} = \boldsymbol{w}_k - \frac{1}{L}\nabla l(\boldsymbol{w}_k)$ 迭代式
  - 然后**对整体求解** $\boldsymbol{w}_{k+1} = \underset{\boldsymbol{w}}{\operatorname{argmin}} \frac{L}{2}\|\boldsymbol{w}-\boldsymbol{z}\|_2^2 + \lambda\|\boldsymbol{w}\|_1$ 迭代式
  - 对迭代计算迭代式1和迭代式2，即可得到结果

迭代式2如何求解?

L_1正则化的求解

- **近端映射**

  近端算子(Proximal Operator)

$$\mathrm{prox}_{\lambda,h(\boldsymbol{w})}(\boldsymbol{z}) = \underset{\boldsymbol{w}}{\mathrm{argmin}}\frac{L}{2}\|\boldsymbol{w}-\boldsymbol{z}\|_2^2 + \lambda\cdot h(\boldsymbol{w})$$

➢ **在L_1正则化中，也可称为软阈值函数**

$$S_\lambda(\boldsymbol{z}) = \underset{\boldsymbol{w}}{\mathrm{argmin}}\frac{L}{2}\|\boldsymbol{w}-\boldsymbol{z}\|_2^2 + \lambda\|\boldsymbol{w}\|_1$$

➢ **对迭代式2，它的w各个分量互不影响，w各个分量可以用下式求解**

$$w_{k+1}^i = \begin{cases} z^i - \lambda/L, & \lambda/L < z^i \\ 0, & \lambda/L \geq |z^i| \\ z^i + \lambda/L, & -\lambda/L > z^i \end{cases} \tag{5.3.6}$$

(4.3.6)与迭代式2等价性证明：《南瓜书》11.3.10

L_1正则化的求解

- **其它L_1正则化求解方法**
  - **次梯度法(Subgradient Method)**
    - **通过使用目标函数在不可微处的次梯度代替传统梯度，进行迭代优化以求解不可微的优化问题**
  - **坐标下降法 (Coordinate Descent)**
    - **将多变量优化问题分解为多个单变量问题，逐变量沿坐标轴方向上进行优化，直到收敛**
  - **近似法 (Approximation Methods)**
    - **通过将不可微问题近似为可微问题，利用传统的优化方法进行求解**
  - **.......**

L_1正则化的示例

- **示例： Lasso回归**
  - **Lasso回归**

$$\min_{\boldsymbol{w}} \frac{1}{2}\|\boldsymbol{X}\boldsymbol{w}-\boldsymbol{y}\|_2^2+\lambda\|\boldsymbol{w}\|_1$$

问题：使用卧室数量、房屋面积等8个特征对房价预测，对比训练后两个模型的参数情况

求解：使用近端梯度下降求解

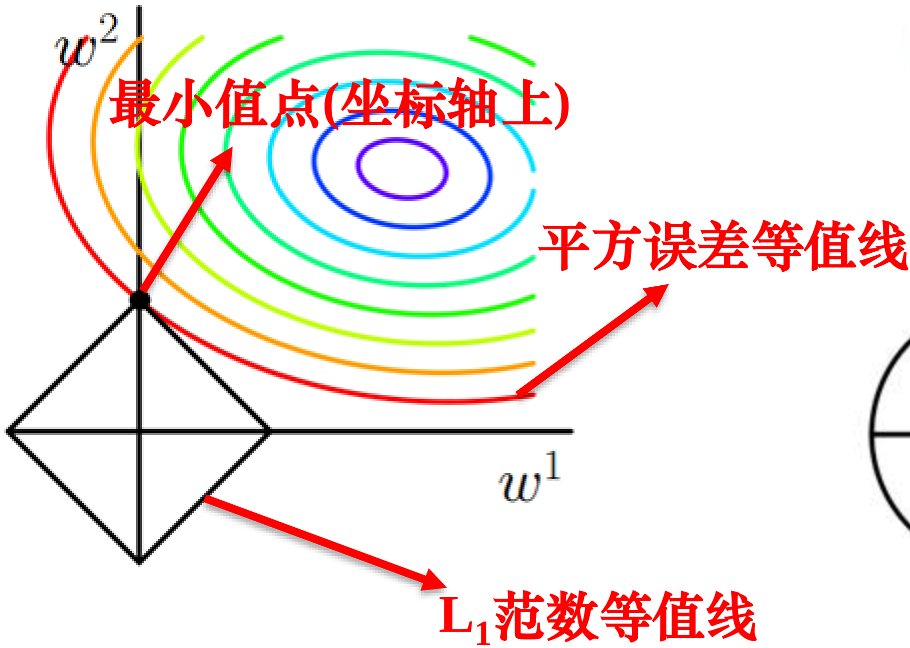
性质特点：可以看出Lasso回归训练后能使部分参数为零，即具有求取稀疏解的能力

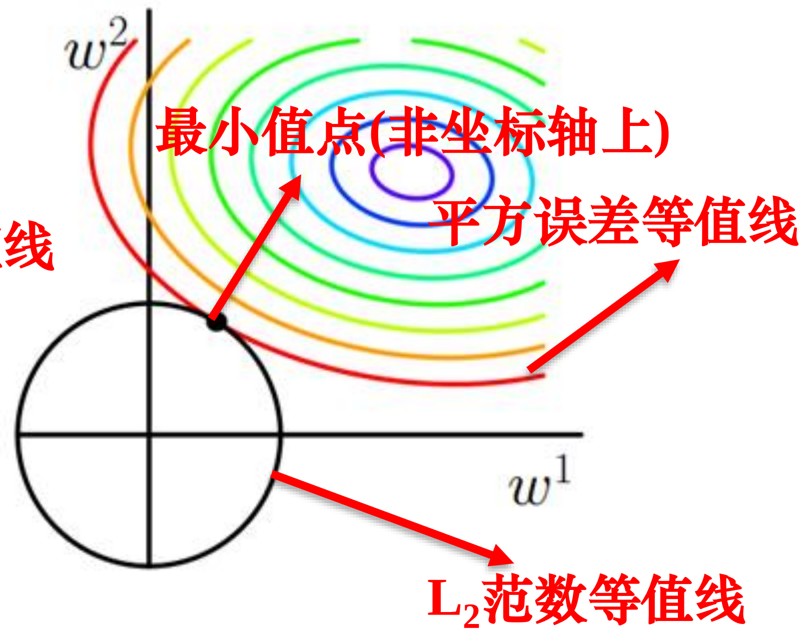
	线性回归	Lasso回归 λ=0.01	Lasso回归 λ=0.1	Lasso回归 λ=1
$\boldsymbol{w}_0^*$	-0.1084	-0.1052	-0.0921	0
$\boldsymbol{w}_1^*$	0.0464	0.0399	0	0
$\boldsymbol{w}_2^*$	-0.0537	-0.0412	0	0
$\boldsymbol{w}_3^*$	0.0712	0.0689	0.0254	0
$\boldsymbol{w}_4^*$	-0.5957	-0.5801	-0.3821	0
$\boldsymbol{w}_5^*$	0.6248	0.6179	0.5738	0.4826
$\boldsymbol{w}_6^*$	-0.0012	0	0	0
$\boldsymbol{w}_7^*$	-0.4792	-0.4663	-0.3967	-0.1532

数据源： https://archive.ics.uci.edu/ml/machine-learning-databases/housing/housing.data

L_1正则化与稀疏解

- **L_1正则化比L_2正则化更易获得稀疏解**
  - **L_1正则化在L_1等值线角点上更容易取到最优解(最小值)，在角点上取值，会使权值w分量取零值。因此，L_1正则化可以使模型参数稀疏化**





L_1正则化与稀疏解关系具体说明：《机器学习》周志华著 P253

稀疏学习（*拓展）

- 基本概念
- 字典学习

模型的稀疏性

- **回顾LASSO回归**
  - ➢ LASSO回归：L1范数的约束下，模型权重向稀疏化方向发展。使得模型进行**特征选择**，即对数据**按行选择**置零，引入了**稀疏性**
  - ➢ 模型的**稀疏性**：模型中**大部分参数都是零**或接近零

模型稀疏性的好处

1. **特征选择**：自动压缩不重要特征权重，保留关键特征，简化模型

2. **降低过拟合**：限制模型复杂度，提高泛化能力

数据表示的稀疏性

- **回顾LASSO回归**

模型的稀疏性

$$\min_{\boldsymbol{w}} \frac{1}{2}\|\boxed{\boldsymbol{X}}\boxed{\boldsymbol{w}} - \boldsymbol{y}\|_2^2 + \lambda\|\boldsymbol{w}\|_1 \qquad (5.4.1)$$

数据表示的稀疏性

➢ **数据表示的稀疏性：即稀疏表示，数据表示中只含有少量非零元素**

稀疏表示的好处

1. **提升存储计算效率：数据易于存储，模型计算量减少**
2. **提取关键特征：学习数据的稀疏表示，实现对数据的高效分析**

如何为普通稠密表达的数据找到合适的“稀疏表示”形式？字典学习是一个代表性方法

字典学习的基本概念

- **字典学习**

  ➢ 基本概念：通过为**普通稠密表达的数据**学习一组过完备的基向量（原子）来作为**合适的字典**，在表示中尽可能使用较少的原子将样本转化为**稀疏表达形式**。这个过程通常称之为**“字典学习”或是“稀疏编码”**

  ➢ 例子：文档的字典表示

字典学习的基本概念

- **基本定义：对于给定的数据集$X=\{x_1,\ldots,x_n\}$，通过由原子b_i构成的字典矩阵B和由稀疏向量α_i构成的稀疏矩阵A对数据集样本进行近似表达**

$$\underbrace{(\mathbf{X})}_{\mathbf{X}\in\mathbb{R}^{m\times n}} \approx \underbrace{(\boldsymbol{b}_1|\boldsymbol{b}_2|\cdots|\boldsymbol{b}_k)}_{\mathbf{B}\in\mathbb{R}^{m\times k}} \underbrace{\begin{pmatrix}\boldsymbol{\alpha}_1\\ \vdots \\ \boldsymbol{\alpha}_k\end{pmatrix}}_{\mathbf{A}\in\mathbb{R}^{k\times n},\ \text{稀疏}} \tag{5.4.2}$$

字典学习的求解

- 字典学习的求解

  - 优化目标：$\min_{\mathbf{B},\boldsymbol{\alpha}_i} \sum_{i=1}^{m} \|\boldsymbol{x}_i - \mathbf{B}\boldsymbol{\alpha}_i\|_2^2 + \lambda\|\boldsymbol{\alpha}_i\|_1$，**准确表示**样本且$\alpha$**尽量稀疏**
  - 算法基本思路：式子中包含字典B和稀疏向量α_i两项，直接求解较为困难。考虑采用变量**交替优化的策略**来求解
  - 算法流程

步骤1：初始化字典矩阵B

步骤2：通过固定字典B，转化为LASSO形式进行求解，更新稀疏矩阵A

步骤3：通过固定稀疏矩阵A，更新字典B

步骤4：反复迭代上述两步，最终可求得字典B和稀疏矩阵A

字典学习的求解

- **步骤2**
  - ➢ **原优化目标：** $$\min_{\mathbf{B},\alpha_i} \sum_{i=1}^{m} \|\boldsymbol{x}_i - \mathbf{B}\boldsymbol{\alpha}_i\|_2^2 + \lambda\|\boldsymbol{\alpha}_i\|_1 \qquad (5.4.3)$$
  - ➢ **通过固定字典B，更新稀疏向量α，将上式转化成如下形式**

$$\min_{\boldsymbol{\alpha}_i} \|\boldsymbol{x}_i - \mathbf{B}\boldsymbol{\alpha}_i\|_2^2 + \lambda\|\boldsymbol{\alpha}_i\|_1 \qquad (5.4.4)$$

符合先前LASSO求解的形式，可以直接求解

$$\boxed{\min_{\boldsymbol{w}} \frac{1}{2}\|\boldsymbol{X}\,\boldsymbol{w} - \boldsymbol{y}\|_2^2 + \lambda\|\boldsymbol{w}\|_1}$$

Lasso回归的损失函数

字典学习的求解

- **步骤3**

  - **原优化目标：** $\min\limits_{\mathbf{B},\boldsymbol{\alpha}_i} \sum_{i=1}^{m} \|\boldsymbol{x}_i - \mathbf{B}\boldsymbol{\alpha}_i\|_2^2 + \lambda\|\boldsymbol{\alpha}_i\|_1$，**通过固定稀疏矩阵A** $\in \mathbb{R}^{k\times n}$**，更新字典B，将优化目标转化成下式**

$$\min_{\mathbf{B}} \|\mathbf{X} - \mathbf{BA}\|_2^2 \tag{5.4.5}$$

  - **求解思路：基于逐列更新策略，在字典B更新的过程中，每次只更新一个原子b_i和对应的稀疏向量a_i，最终通过多次迭代取得满意的近似解**

字典学习的求解

- **逐列更新策略**

  - 优化目标 $\min_{\mathbf{B}}\|\mathbf{X}-\mathbf{BA}\|_2^2$
  - 通过对B和A进行按行、按列拆分，上式重写为

$$\min_{\mathbf{B}}\|\mathbf{X}-\mathbf{BA}\|_2^2=\min_{\boldsymbol{b}_i}\left\|\mathbf{X}-\sum_{j=1}^{k}\boldsymbol{b}_i\boldsymbol{\alpha}_i\right\|_2^2 \tag{5.4.6}$$

$$=\min_{\boldsymbol{b}_i}\left\|\left(\mathbf{X}-\sum_{j\neq i}\boldsymbol{b}_j\boldsymbol{\alpha}_j\right)-\boldsymbol{b}_i\boldsymbol{\alpha}_i\right\|_2^2$$

$$=\min_{\boldsymbol{b}_i}\|\mathbf{E}_i-\boldsymbol{b}_i\boldsymbol{\alpha}_i\|_2^2$$

逐列更新时其他列固定，则E_i矩阵已知

该项含义为**最小化矩阵重建误差**，可采用矩阵的奇异值分解进行求解

字典学习

- **逐列更新策略**

  - **通过最小化E_i重建误差用SVD求解得：**

$$\boldsymbol{b}_i = \mathrm{U}_{\cdot,1}, \boldsymbol{\alpha}_i = \Sigma_{1,1}\mathrm{V}_{\cdot,1}{}^{\mathrm{T}}$$

> - U，V为E_i的左右奇异矩阵，$\mathrm{E}_i = \mathrm{U}\Sigma\mathrm{V}^{\mathrm{T}}$
> - b_i和α_i分别为最大奇异值对应的U的第一行和V第一列的两个向量

  - **稀疏性保持：通过保留α_i中已有的0值来保留已有系数的稀疏性，仅取出α_i中非零的元素记为α_i'，保留E_i中对应b_i和α_i'的乘积项为E_i'。再进行上述步骤最小化误差**

奇异值分解：https://zh.wikipedia.org/wiki/%E5%A5%87%E5%BC%82%E5%80%BC%E5%88%86%E8%A7%A3

字典学习的应用

● 图像去噪

- ➢ 图像去噪：从受到噪声干扰的图像中**恢复出原始图像**

带噪图像

图像去噪

无噪图像

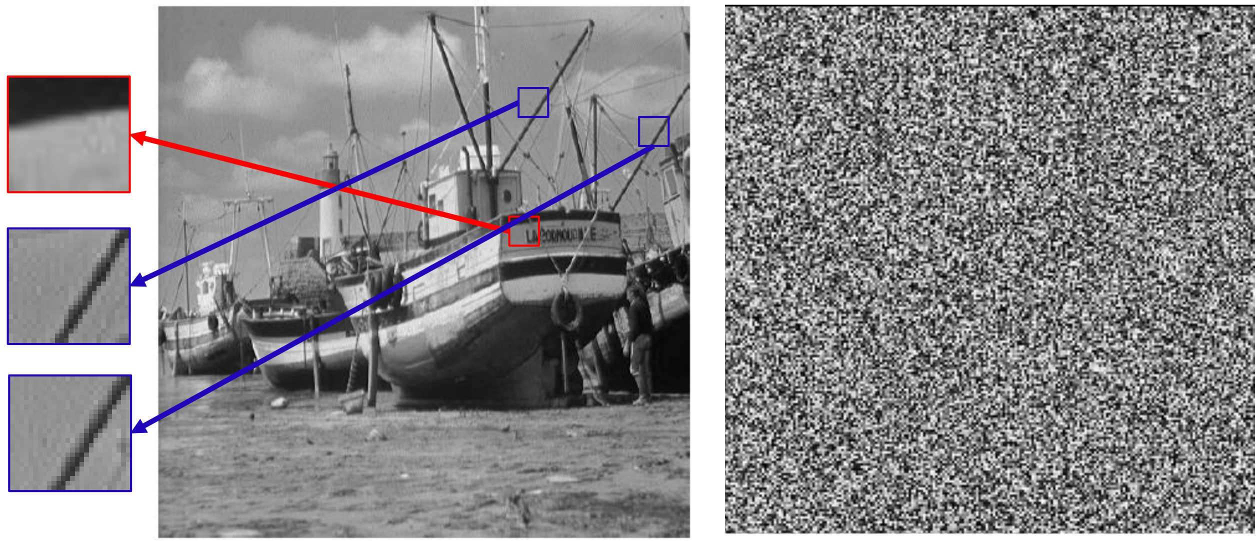
- ➢ 字典学习是一种常用的图像去噪方法，它通过学习自然图像的稀疏表示，实现对图像的去噪处理

字典学习的应用

- **图像去噪**
  - ➢ **字典学习重建图像去噪原理**

局部冗余性

相似结构和纹理



随机、高频

易被稀疏表示　　难以稀疏表示

二者可通过字典学习分离

字典学习的应用

- **图像去噪**
  - ➢ **字典学习步骤**

$$\underbrace{\min_{\mathbf{B},\boldsymbol{\alpha}_i} \sum_{i=1}^{m} \| \mathbf{x}_i - \mathbf{B}\boldsymbol{\alpha}_i \|_2^2}_{\text{重建}} + \underbrace{\lambda \sum_{i=1}^{m} \| \boldsymbol{\alpha}_i \|_1}_{\text{稀疏性}}$$

  - ➢ **去噪步骤**