线性回归模型

线性回归

- 给定训练数据 $D = \{\mathbf{x}_i|y_i\}_{i=1}^N$ ，其中 $y \in \mathbb{R}$ 是连续值，一共有N个样本，线性回归的目标是学习一个从输入$\mathbf{x}$ 到输出 y 的映射 f
- 对新的测试数据 $\mathbf{x}$，用学习到的映射 f 对其进行预测：$\hat{y} = f(\mathbf{x})$
- 若假设映射 f 是一个线性函数，即

$$y = f(\mathbf{x}|\mathbf{w}) = \mathbf{w}^T\mathbf{x}$$

- 我们称之为线性回归模型。

目标函数

- 目标函数通常包含两项：**损失函数**和**正则项**，分别代表度量模型与训练数据的匹配程度（损失函数越小越匹配）和对模型的“惩罚”以避免过拟合。

$$J(\theta) = \frac{1}{N}\sum_{i=1}^{N} L(f(\mathbf{x}_i;\theta),y_i) + \Omega(\theta)$$

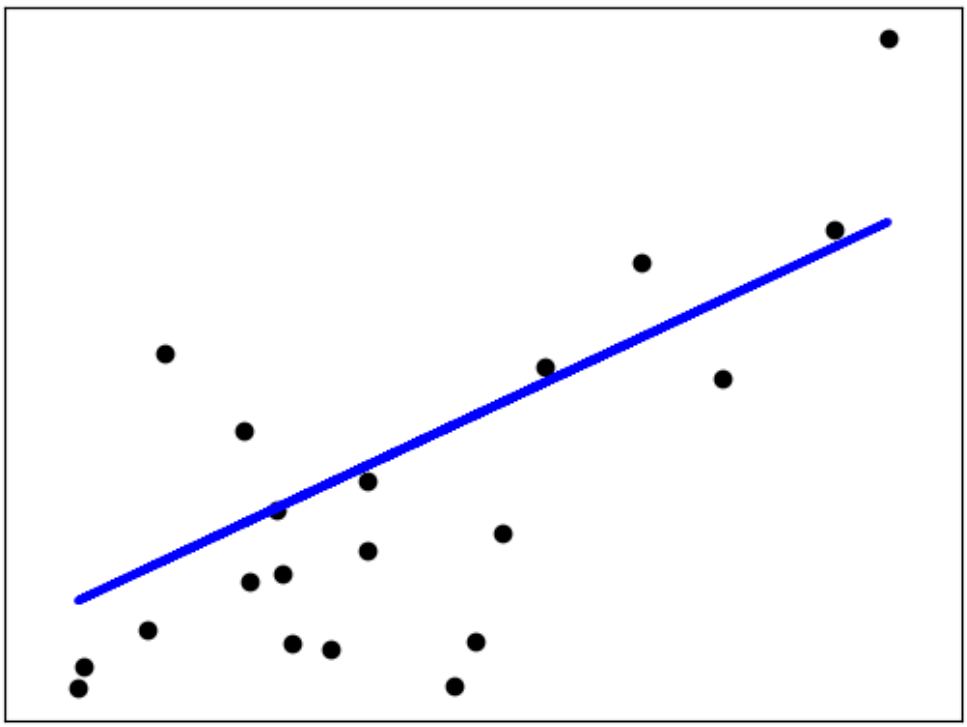
- 对回归问题，损失函数可以采用L2损失（也可以根据实际情况选择其他有意义的损失函数），得到

$$J(\theta) = \sum_{i=1}^{N} l(y_i,\hat{y}_i) = \sum_{i=1}^{N}(y_i-\hat{y}_i)^2 = \sum_{i=1}^{N}(y_i-\mathbf{w}^T\mathbf{x_i})^2$$

这就是**残差平方和**(residual sum of squares，RSS)

线性回归的正则项

- 由于线性模型比较简单，实际应用中有时正则项为空，得到最小二乘线性回归（Ordinary Least Square，OLS）



- 正则项可以为L2正则，得到岭回归（Ridge Regression）模型：

$$J(\mathbf{w}) = \sum_{i=1}^{N}(y_i-\mathbf{w}^T\mathbf{x_i})^2 + \lambda||\mathbf{w}||_2^2$$

- 正则项也可以选L1正则，得到Lasso模型：

$$J(\mathbf{w}) = \sum_{i=1}^{N}(y_i - \mathbf{w}^T\mathbf{x_i})^2 + \lambda|\mathbf{w}|$$

  - 当λ取合适值时，Lasso（ least absolute shrinkage and selection operator）的结果是**稀疏**的（**w**的某些元素系数为0），起到**特征选择作用**。

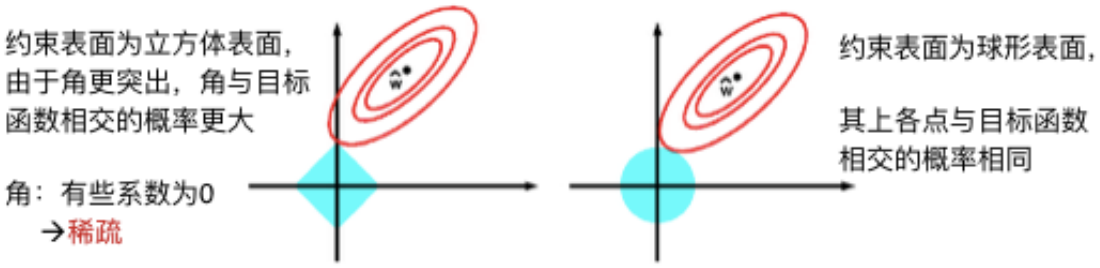
为什么L_1正则的解是稀疏的?

- 考虑两个优化问题：

$$\min_w RSS(\mathbf{w}) + \lambda||\mathbf{w}||_1$$
$$\min_w RSS(\mathbf{w}) + \lambda||\mathbf{w}||_2^2$$

- 分别等价于（连续的带约束的优化问题）

$$\min_w RSS(\mathbf{w}) \text{ s.t. } ||\mathbf{w}||_1 \leq B$$
$$\min_w RSS(\mathbf{w}) \text{ s.t. } ||\mathbf{w}||_2^2 \leq B$$



- 例：如 $\mathbf{w} = (1,0)^T, (1/\sqrt{2}, 1/\sqrt{2})^T$ 的L2 模相同(1)，但L1模分别为1和$\sqrt{2}$

线性回归模型的概率解释

- 最小二乘（线性）回归等价于极大似然估计
- 正则（线性）回归等价于高斯先验（L2正则）或Laplace先验（L1正则）下的贝叶斯估计

最小二乘线性回归等价于极大似然估计

- 假设：$y = f(\mathbf{x}) + \epsilon = \mathbf{w}^T\mathbf{x} + \epsilon$
- 其中ϵ为线性预测和真值之间的**残差**
- 我们通常假设残差的分布为 $\epsilon \sim N(0, \sigma^2)$，均值为0，方差$\sigma^2$。对该残差分布的基础上，加上y的分布，因此线性回归可写成：

$$p(y|\mathbf{x}, \theta) \sim N(y|\mathbf{w}^T\mathbf{x}, \sigma^2)$$

  其中 $\theta = (\mathbf{w}, \sigma^2)$。均值移动变化，方差没有变。

Recall：极大似然估计

- 极大似然估计（Maximize Likelihood Estimator, MLE）定义为（即给定参数θ的情况下，数据D出现的概率p，取出其中最大的参数θ）

$$\hat{\theta} = \arg\max_{\theta} \log p(D|\theta)$$

其中（log）似然函数为

$$l(\theta) = \log p(D|\theta) = \sum_{i=1}^{N} \log p(y_i|x_i, \theta)$$

- 表示在参数为 θ 的情况下，数据 $D = \{\mathbf{x_i}, y_i\}_{i=1}^N$ 出现的概率.
- 极大似然：选择数据出现概率最大的参数。

线性回归的MLE

$$p(y_i|\mathbf{x}_i, \mathbf{w}, \sigma^2) = (y_i|\mathbf{w}^T\mathbf{x}_i, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}}\exp(-\frac{1}{2\sigma^2}(y_i - \mathbf{w}^T\mathbf{x}_i)^2)$$

- OLS的似然函数为

$$l(\theta) = \log p(D, \theta) = \sum_{i=1}^{N} \log p(y_i|x_i, \theta)$$

- 极大似然可等价地写成极小负log似然损失(negative log likelihood， **NLL**)(在sklearn中，叫做logloss)

$$\begin{aligned} NLL(\theta) &= -\sum_{i=1}^{N} \log p(y_i|x_i, \theta) = -\sum_{i=1}^{N} \log[(\frac{1}{2\pi\sigma^2})^{1/2}\exp(-\frac{1}{2\sigma^2}(y_i - \mathbf{w}^T\mathbf{x}_i)^2)] \\ &= \frac{N}{2}\log(2\pi\sigma^2) + \frac{1}{2\sigma^2}\sum_{i=1}^{N}(y_i - \mathbf{w}^T x_i)^2 \end{aligned}$$

上式中，观察第二项即可得知OLS的RSS项与MLE是等价的关系。

正则回归等价于贝叶斯估计

- 假设残差的分布为 $\epsilon \sim N(0, \sigma^2)$，线性回归可写成：

$$\begin{aligned} p(y_i|\mathbf{x}_i, \theta) &\sim N(y_i|\mathbf{w}^T\mathbf{x}_i, \sigma^2) \\ p(\mathbf{y}_i|\mathbf{X}, \mathbf{w}, \sigma^2) = N(\mathbf{y}|\mathbf{X}\mathbf{w}, \sigma^2\mathbf{I}_N) &\propto \exp(-\frac{1}{2\sigma^2}[(\mathbf{y} - \mathbf{X}\mathbf{w})^T(\mathbf{y} - \mathbf{X}\mathbf{w})]) \end{aligned}$$

- 若假设参数w的先验分布为 $w_j \sim N(0, \tau^2)$
  - 偏向较小的系数值，从而得到的曲线也比较平滑

$$p(\mathbf{w}) = \Pi_{j=1}^{D} N(w_j|0,\tau^2) \propto \exp((-\frac{1}{2\tau^2}\sum_{j=1}^{D}\mathbf{w}_j^2)$$

$$= \exp(-\frac{1}{2\tau^2}[\mathbf{w}^T\mathbf{w}])$$

  - 其中 $1/\tau^2$ 控制先验的强度

- 根据贝叶斯公式，得到参数的后验分布为

$$p(\mathbf{w}|\mathbf{X},\mathbf{y},\sigma^2) \sim \exp(-\frac{1}{2\sigma^2}[(\mathbf{y}-\mathbf{X}\mathbf{w}^T)(\mathbf{y}-\mathbf{X}\mathbf{w}))] - \frac{1}{2\tau^2}[\mathbf{w}^T\mathbf{w}])$$

- 则最大后验估计（MAP）等价于最小目标函数

$$J(\mathbf{w}) = (\mathbf{y}-\mathbf{X}\mathbf{w})^T(\mathbf{y}-\mathbf{X}\mathbf{w}) + \frac{\sigma^2}{\tau^2}\mathbf{w}^T\mathbf{w}$$

- 对比岭回归的目标函数

$$J(\mathbf{w}) = \sum_{i=1}^{N}(y_i - \mathbf{w}^T\mathbf{x})^2 + \lambda||\mathbf{w}||_2^2$$

- 惊喜！

小结之目标函数

- 线性回归模型也可以放到机器学习一般框架
-
  - 损失函数：L2损失...
  - 正则：无正则、L2正则、L1正则...
- 正则回归模型可视为先验为正则、似然为高斯分布的贝叶斯估计

优化求解

- 线性回归的目标函数
  - 无正则的最小二乘线性回归（Ordinary Least Square，OLS）

$$J(\mathbf{w}) = \sum_{i=1}^{N}(y_i - \mathbf{w}^T\mathbf{x}_i)^2$$

  - L2正则的岭回归（Ridge Regression）模型：

$$J(\mathbf{w}) = \sum_{i=1}^{N}(y_i - \mathbf{w}^T\mathbf{x_i})^2 + \lambda||\mathbf{w}||_2^2$$

  - L1正则的Lasso模型：

$$J(\mathbf{w}) = \sum_{i=1}^{N}(y_i - \mathbf{w}^T\mathbf{x_i})^2 + \lambda|\mathbf{w}|$$

模型训练

- 模型训练是根据训练数据求解最优模型参数，即求目标函数取极小值的参数

$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w}} J(\mathbf{w})$$

- 根据模型的特点和问题复杂程度（数据、模型），可选择不同的优化算法
  - 一阶的导数为0：$\frac{\partial J(\mathbf{w})}{\partial \mathbf{w}} = \mathbf{0}$
  - 二阶导数>0：$\frac{\partial J^2(\mathbf{w})}{\partial \mathbf{w}^2} > \mathbf{0}$

OLS的优化求解

- 将OLS的目标函数写成矩阵形式

$$J(\mathbf{w}) = \sum_{i=1}^{N}(y_i - \mathbf{w}^T\mathbf{x}_i)^2 = (\mathbf{y} - \mathbf{X}\mathbf{w}^T)(\mathbf{y} - \mathbf{X}\mathbf{w})$$

- 只取与**w**有关的项，得到

$$J(\mathbf{w}) = \mathbf{w}^T(\mathbf{X}^T\mathbf{X})\mathbf{w} - 2\mathbf{w}^T(\mathbf{X}^T\mathbf{y})$$

- 求导

$$\frac{\partial}{\partial \mathbf{w}} J(\mathbf{w}) = 2\mathbf{X}^T\mathbf{X}\mathbf{w} - 2\mathbf{X}^T\mathbf{y} = 0 \rightarrow \mathbf{X}^T\mathbf{X}\mathbf{w} = \mathbf{X}^T\mathbf{y}$$

$$\frac{\partial}{\partial y}(\mathbf{y}^T\mathbf{A}\mathbf{y}) = (\mathbf{X} + \mathbf{X}^T)\mathbf{y}$$

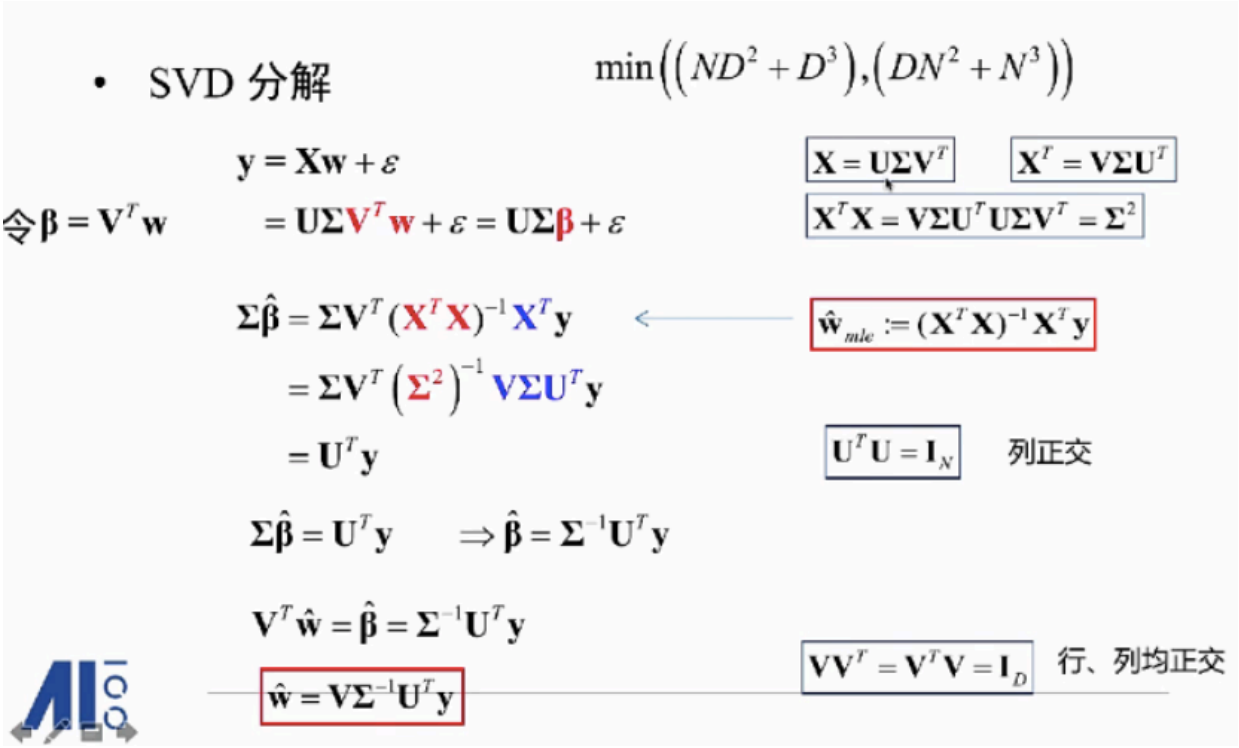
$$\frac{\partial}{\partial \mathbf{a}}(\mathbf{b}^T\mathbf{a}) = \mathbf{b}$$

所以，得到 (ordinary least squares, OLS)

$$\hat{\mathbf{W}}_{OLS} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

通常通过奇异值分解（singular value decomposition，SVD）求解。

OLS的计算



岭回归的优化求解

- 岭回归的目标函数与OLS只相差一个正则项（也是**w**的二次函数），所以类似可得：

$$J(\mathbf{w})=\sum_{i=1}^{N}(y_i-\mathbf{w}^T\mathbf{x_i})^2+\lambda||\mathbf{w}||_2^2=(\mathbf{y}-\mathbf{X}\mathbf{w}^T)(\mathbf{y}-\mathbf{X}\mathbf{w})+\lambda\mathbf{w}^T\mathbf{w}$$

- 求导，得到

$$\frac{\partial}{\partial\mathbf{w}}J(\mathbf{w})=2\mathbf{X}^T\mathbf{X}\mathbf{w}-2\mathbf{X}^T\mathbf{y}+2\lambda\mathbf{w}^T=0$$

$$\hat{\mathbf{W}}_{ridge}=(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I}_D)^{-1}\mathbf{X}^T\mathbf{y}$$

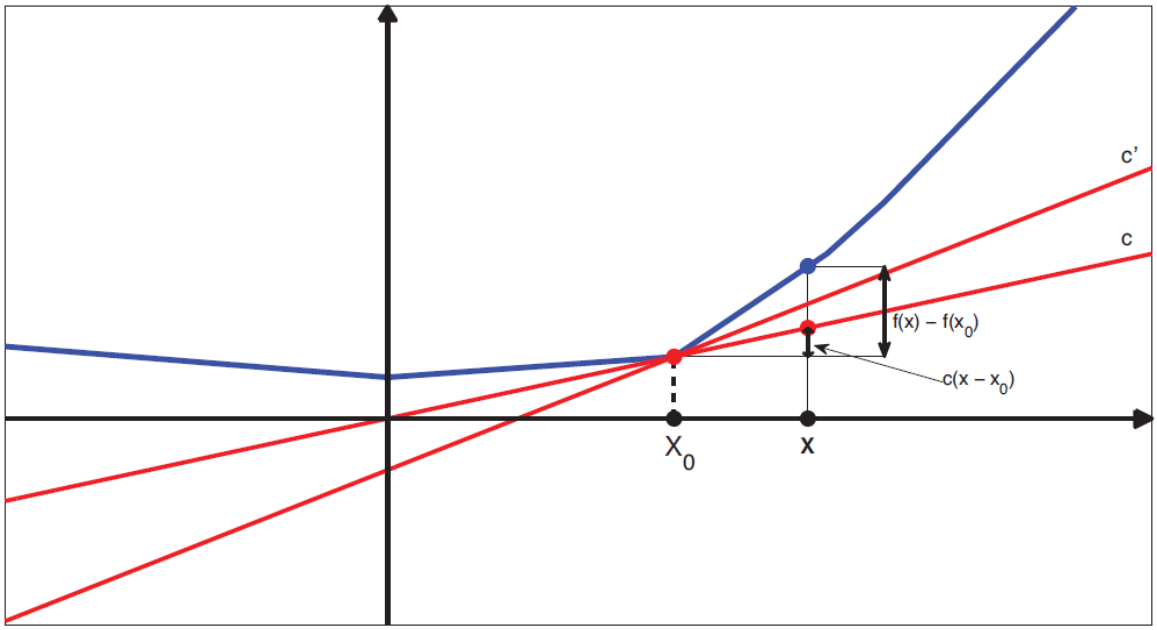
Lasso的优化条件

- Lasso的目标函数为 $J(\mathbf{w},\lambda)=RSS(\mathbf{w})+\lambda||\mathbf{w}||_1$
- 但项 $||\mathbf{w}||_1$ 在 $\mathbf{w}_j=0$ 处不可微（不平滑优化问题）
- 为了处理不平滑函数，扩展导数的表示，定义一个 (凸)函数 f 在点处 x_0 的次梯度(subgradient)或次导数(subderivative)为一个标量g，使得

$$f(x)-f(x_0)\geq g(x-x_0),\forall x\in I$$

- 其中I为包含x_0的某个区间。
- 定义区间[a,b]的子梯度集合为

$$a=\lim_{x\to x_0^-}\frac{f(x)-f(x_0)}{x-x_0},b=\lim_{x\to x_0^+}\frac{f(x)-f(x_0)}{x-x_0}$$



- 所有次梯度的区间称为函数f在处x_0的次微分(subdifferential)，用 $\partial f(x)|_{x_0}$ 表示
- 例：绝对值函数 $f(x) = |x|$，其次梯度为

$$\partial f(x) = \begin{cases} \{-1\} & \text{if } x < 0 \\ [-1, +1] & \text{if } x = 0 \\ \{+1\} & \text{if } x > 0 \end{cases}$$

- 如果函数处处可微，$\partial f(x) = \frac{df(x)}{dx}$
- 同标准的微积分类似，可以证明当且仅当 $0 \in \partial f(x)|_{\hat{x}}$ 时，为 f 的局部极值点。
- 将上述结论带入Lasso问题
- 目标函数为：

$$J(\mathbf{w}, \lambda) = RSS(\mathbf{w}) + \lambda||\mathbf{w}||_1 = \sum_{i=1}^{N}(y_i - \mathbf{w}^T\mathbf{x_i})^2 + \lambda||\mathbf{w}||_1$$

- 可微项的梯度：

$$\frac{\partial}{\partial w_j} RSS(\mathbf{w}) = \frac{\partial}{\partial w_j}\sum_{i=1}^{N}\left(y_i - \left(\mathbf{w}_{-j}^T\mathbf{x}_{i,-j} + w_j x_{ij}\right)\right)^2$$

$$= -2\sum_{i=1}^{N}\left(y_i - \mathbf{w}_{-j}^T\mathbf{x}_{i,-j} - w_j x_{ij}\right)x_{ij}$$

$$= 2\sum_{i=1}^{N} x_{ij}^2 w_j - 2\sum_{i=1}^{N}\left(y_i - \mathbf{w}_{-j}^T\mathbf{x}_{i,-j}\right)x_{ij}$$

$$= a_j w_j - c_j$$

$$a_j = 2\sum_{i=1}^{N} x_{ij}^2$$

$$c_j = 2\sum_{i=1}^{N} x_{ij}\left(y_i - \mathbf{w}_{-j}^T\mathbf{x}_{i,-j}\right)$$

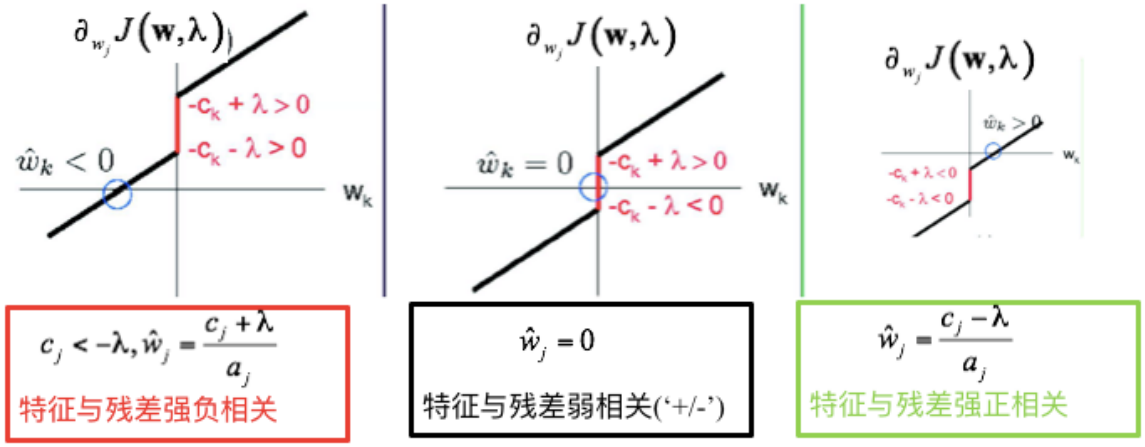
第j维特征与残差的相关性

利用D-j 维特征得到的预测的残差

- 目标函数的子梯度(subgradient)

$$\partial_{w_j} J(\mathbf{w}, \lambda) = (a_j w_j - c_j) + \lambda \partial_{w_j} ||\mathbf{w}||_1 = \begin{cases} \{a_j w_j - c_j - \lambda\} & \text{if } w_j < 0 \\ [c_j - \lambda, c_j + \lambda] & \text{if } w_j = 0 \\ \{a_j w_j - c_j + \lambda\} & \text{if } w_j > 0 \end{cases}$$

- 根据 c_j 的不同，$\partial_{w_j} J(\mathbf{w}, \lambda) = 0$有三种情况



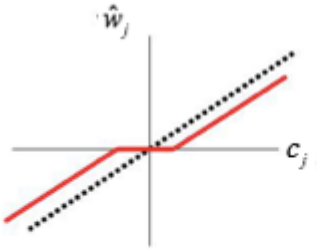
$c_j < -\lambda, \hat{w}_j = \frac{c_j + \lambda}{a_j}$
特征与残差强负相关

$\hat{w}_j = 0$
特征与残差弱相关('+/-')

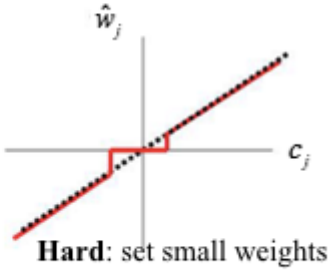
$\hat{w}_j = \frac{c_j - \lambda}{a_j}$
特征与残差强正相关

- 软 & 硬阈

$$\hat{w}_j(c_j) = \begin{cases} (c_j + \lambda)/a_j & \text{if } c_j < -\lambda \\ 0 & \text{if } c_j \in [-\lambda, \lambda] \\ (c_j - \lambda)/a_j & \text{if } c_j > \lambda \end{cases} = \text{soft}\left(\frac{c_j}{a_j}; \frac{\lambda}{a_j}\right) \quad \text{soft}(a;\delta) = \text{sign}(a)(|a| - \delta)_+$$



Soft: set small weights to zero and "shrink" all other weights.
Biased estimator



Hard: set small weights to zero without "shrinking" all other weights.
Unbiased estimator

- 预计算

$$a_j = 2 \sum_{i=1}^{N} x_j^2$$

- 初始化参数**w**（全0或随机） 循环直到收敛:
- - for *j* = 1,...*D*
  - - 计算

$$c_j = 2 \sum_{i=1}^{N} x_j (y_i - \mathbf{w}_{-j}^T \mathbf{x}_{i,-j})$$

    - 更新w_j:

$$\hat{w}_j(c_j) = \begin{cases} (c_j + \lambda)/a_j & \text{if } c_j < -\lambda \\ 0 & \text{if } c_j \in [-\lambda, \lambda] = \text{soft}(\frac{c_j}{a_j}; \frac{\lambda}{a_j}) \\ (c_j - \lambda)/a_j & \text{if } fc_i > \lambda \end{cases}$$

- 选择变化幅度最大的维度进行更新

坐标轴下降法

- 为了找到一个函数的局部极小值，在每次迭代中可以在当前点处沿一个坐标方向进行一维搜索。
- 整个过程中循环使用不同的坐标方向。一个周期的一维搜索迭代过程相当于一个梯度迭代。
- 注意：
-
  - 梯度下降方法是利用目标函数的导数（梯度）来确定搜索方向的，而该梯度方向可能不与任何坐标轴平行。
  - 而坐标轴下降法是利用当前坐标系统进行搜索，不需要求目标函数的导数，只按照某一坐标方向进行搜索最小值。（在稀疏矩阵上的计算速度非常快，同时也是Lasso回归最快的解法）

梯度下降

- 在线性回归中，目标函数中的训练误差部分为

$$J(\mathbf{w}) = \sum_{i=1}^{N}(f(\mathbf{x}_i) - y_i)^2$$

- 梯度为

$$g(\mathbf{w}) = \frac{\partial}{\partial \mathbf{w}} J(\mathbf{w}) = \sum_{i=1}^{N} 2(f(\mathbf{x}_i) - y_i)\mathbf{x}_i$$

- L2正则的梯度计算简单（L1正则需要计算次梯度）

$$\Omega(\mathbf{w}) = \lambda||\mathbf{w}||_2^2 = \lambda \mathbf{w}^T\mathbf{w}, \frac{\partial \Omega(\mathbf{w})}{\partial \mathbf{w}} = 2\lambda\mathbf{w}$$

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

- 在上述梯度下降算法中，梯度 $g(\mathbf{w}) = \sum_{i=1}^{N} 2(f(\mathbf{x}_i) - y_i)\mathbf{x}_i$
  - 利用所有的样本，因此被称为“**批处理**梯度下降”
- 随机梯度下降：每次只用一个样本 $(\mathbf{x}_t, y_t)$

$$g(\mathbf{w}) = 2(f(\mathbf{x}_t), y_t)\mathbf{x}_t$$

  - 通常收敛会更快，且不太容易陷入局部极值
  - 亦被称为**在线学习**(Online Learning)
  - 对大样本数据集尤其有效

- SGD的变形：
  - 可用于离线学习：每次看一个样本，对所有样本可重复多次循环使用（一次循环称为一个epoch）
  - 每次可以不止看一个样本，而是看一些样本（mini-batch）

小结：线性回归之优化求解

- 线性回归模型比较简单
- 当数据规模较小时，可直接解析求解
  - scikit learn中的实现采用SVD分解实现
  - 当数据规模较大时，可采用随机梯度下降
  - Scikit learn提供一个SGDRegression类
- 岭回归求解类似OLS，采用SVD分解实现
- Lasso优化求解采用坐标轴下降法

模型选择

- 线性回归的目标函数
  - 无正则的最小二乘线性回归（Ordinary Least Square，OLS）

$$J(\mathbf{w}) = \sum_{i=1}^{N}(y_i - \mathbf{w}^T\mathbf{x}_i)^2$$

  - L2正则的岭回归（Ridge Regression）模型：

$$J(\mathbf{w}) = \sum_{i=1}^{N}(y_i - \mathbf{w}^T\mathbf{x_i})^2 + \lambda||\mathbf{w}||_2^2$$

  - L1正则的Lasso模型：

$$J(\mathbf{w}) = \sum_{i=1}^{N}(y_i - \mathbf{w}^T\mathbf{x_i})^2 + \lambda|\mathbf{w}|$$

模型评估与模型选择

- 模型训练好后，需要在校验集上采用一些度量准则检查模型预测的效果
  - 校验集划分（train_test_split、交叉验证）
  - 评价指标（sklearn.metrics）
- 模型选择：选择预测性能最好的模型
  - 模型中通常有一些超参数，需要通过模型选择来确定
  - 参数搜索范围：网格搜索（GridSearch）
- Scikit learn将交叉验证与网格搜索合并为一个函数：sklearn.model_selection.GridSearchCV

评价准则

- 模型训练好后，可用一些度量准则检查模型拟合的效果
  - 开方均方误差（rooted mean squared error，RMSE）：

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

  - 平均绝对误差（mean absolute error，MAE）：

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|\hat{y}_i - y_i|$$

  - R2 score：既考虑了预测值与真值之间的差异，也考虑了问题本身真值之间的差异（ scikit learn 线性回归模型的缺省评价准则）

$$SS_{res} = \sum_{i=1}^{N}(\hat{y}_i - y_i)^2, SS_{tot} = \sum_{i=1}^{N}(y_i - \bar{y})^2, R^2 = 1 - \frac{SS_{res}}{SS_{tot}}$$

- 也可以检查残差的分布
- 还可以打印预测值与真值的散点图

Scikit learn中的回归评价指标

Regression	
'explained_variance'	metrics.explained_variance_score
'neg_mean_absolute_error'	metrics.mean_absolute_error
'neg_mean_squared_error'	metrics.mean_squared_error
'neg_mean_squared_log_error'	metrics.mean_squared_log_error
'neg_median_absolute_error'	metrics.median_absolute_error
'r2'	metrics.r2_score

线性回归中的模型选择

sklearn.model_selection

- Scikit learn中的model selection模块提供模型选择功能
-
  - 对于线性模型，留一交叉验证（*N*折交叉验证，亦称为leave-one-out cross-validation，LOOCV）有更简便的计算方式，因此Scikit learn提供了RidgeCV类和LassoCV类
  - 后续课程将讲述一般模型的交叉验证和参数调优GridSearchCV

RidgeCV

- RidgeCV中超参数λ用alpha表示

- RidgeCV(*alphas**=(0.1, 1.0, 10.0), fit_intercept**=True, normalize**=False, scoring**=None, cv**=None, gcv_mode**=None, s**tore_cv_values**=False)*

```
from sklearn.linear_model import  RidgeCV

alphas = [0.01, 0.1, 1, 10,20, 30, 50, 60, 80,100]

reg = RidgeCV(alphas=alphas, store_cv_values=True)
reg.fit(X_train, y_train)
```

LassoCV

- LassoCV的使用与RidgeCV类似
- Scikit learn还提供一个与Lasso类似的LARS（least angle regression，最小角回归），二者仅仅是优化方法不同，目标函数相同。
- 当数据集中特征维数很多且存在共线性时，LassoCV更合适。

小结：线性回归之模型选择

- 采用交叉验证评估模型预测性能，从而选择最佳模型
-
    - 回归性能的评价指标
    - 线性模型的交叉验证通常直接采用广义线性模型的留一交叉验证进行快速模型评估
    -
        - Scikit learn中对RidgeCV和LassoCV实现该功能