



# 岭回归与Lasso回归模型

讲师：刘顺祥

1. 理解岭回归与Lasso回归的系数求解过程
2. 理解Lasso回归如何实现变量选择
3. 掌握岭回归与Lasso回归的实操

根据线性回归模型的参数估计公式 $\beta = (X'X)^{-1}X'y$ 可知，得到 $\beta$ 的前提是矩阵 $X'X$ 可逆，但在实际应用中，可能会出现自变量个数多于样本量或者自变量间存在多重共线性的情况，即 $X'X$ 的行列式为0。此时将无法根据公式计算回归系数的估计值 $\beta$ 。

## 当列数比行数多



构造矩阵

$$X = \begin{bmatrix} 1 & 2 & 5 \\ 6 & 1 & 3 \end{bmatrix}$$



计算 $X'X$

$$X'X = \begin{bmatrix} 1 & 6 \\ 2 & 1 \\ 5 & 3 \end{bmatrix} \begin{bmatrix} 1 & 2 & 5 \\ 6 & 1 & 3 \end{bmatrix} = \begin{bmatrix} 37 & 8 & 23 \\ 8 & 5 & 13 \\ 23 & 13 & 34 \end{bmatrix}$$



计算行列式

$$\begin{aligned} |X'X| &= 37 \times 5 \times 34 + 8 \times 13 \times 23 + 23 \times 8 \times 13 \\ &\quad - 37 \times 13 \times 13 - 8 \times 8 \times 34 - 23 \times 5 \times 23 = 0 \end{aligned}$$


当列之间存在多重共线性

 构造矩阵

$$X = \begin{bmatrix} 1 & 2 & 2 \\ 2 & 5 & 4 \\ 2 & 3 & 4 \end{bmatrix}$$

 计算 $X'X$

$$X'X = \begin{bmatrix} 1 & 2 & 2 \\ 2 & 5 & 3 \\ 2 & 4 & 4 \end{bmatrix} \begin{bmatrix} 1 & 2 & 2 \\ 2 & 5 & 4 \\ 2 & 3 & 4 \end{bmatrix} = \begin{bmatrix} 9 & 18 & 18 \\ 18 & 38 & 36 \\ 18 & 36 & 36 \end{bmatrix}$$

 计算行列式

$$\begin{aligned} |X'X| &= 9 \times 38 \times 36 + 18 \times 36 \times 18 + 18 \times 18 \times 36 \\ &\quad - 9 \times 36 \times 36 - 18 \times 18 \times 36 - 18 \times 38 \times 18 = 0 \end{aligned}$$

为解决多元线性回归模型中可能存在的不可逆问题，统计学家提出了岭回归模型。该模型解决问题的思路就是在线性回归模型的目标函数之上添加 $l_2$ 正则项（也称为惩罚项）。

$$J(\beta) = \sum (y - X\beta)^2 + \lambda \|\beta\|_2^2 = \sum (y - X\beta)^2 + \sum \lambda \beta^2$$

- 📌 在线性回归模型的目标函数之上添加 $l_2$ 正则项，其中 $\lambda$ 为非负数
- 📌 当 $\lambda=0$ 时，目标函数退化为线性回归模型的目标函数
- 📌 当 $\lambda \rightarrow +\infty$ 时，通过缩减回归系数使 $\beta$ 趋近于0
- 📌  $\lambda$ 是 $l_2$ 正则项平方的系数，用于平衡模型方差（回归系数的方差）和偏差

## 系数求解



展开岭回归模型中的平方项

$$\begin{aligned} J(\beta) &= (y - X\beta)'(y - X\beta) + \lambda\beta'\beta \\ &= (y' - \beta'X')(y - X\beta) + \lambda\beta'\beta \\ &= y'y - y'X\beta - \beta'X'y + \beta'X'X\beta + \lambda\beta'\beta \end{aligned}$$



计算导函数

$$\begin{aligned} \frac{\partial J(\beta)}{\partial \beta} &= 0 - X'y - X'y + 2X'X\beta + 2\lambda\beta \\ &= 2(X'X + \lambda I)\beta - 2X'y \end{aligned}$$



参数求解

$$\begin{aligned} 2(X'X + \lambda I)\beta - 2X'y &= 0 \\ \therefore \beta &= (X'X + \lambda I)^{-1}X'y \end{aligned}$$



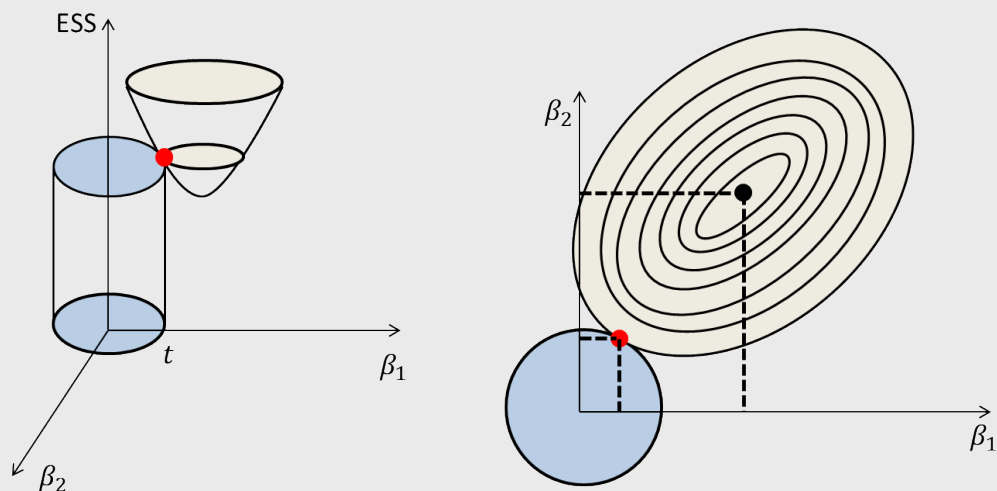
## 凸优化的等价命题

$$J(\beta) = \sum (y - X\beta)^2 + \lambda \|\beta\|_2^2 = \sum (y - X\beta)^2 + \sum \lambda \beta^2$$



$$\begin{cases} \operatorname{argmin} \left\{ \sum (y - X\beta)^2 \right\} \\ \text{附加约束 } \sum \beta^2 \leq t \end{cases}$$

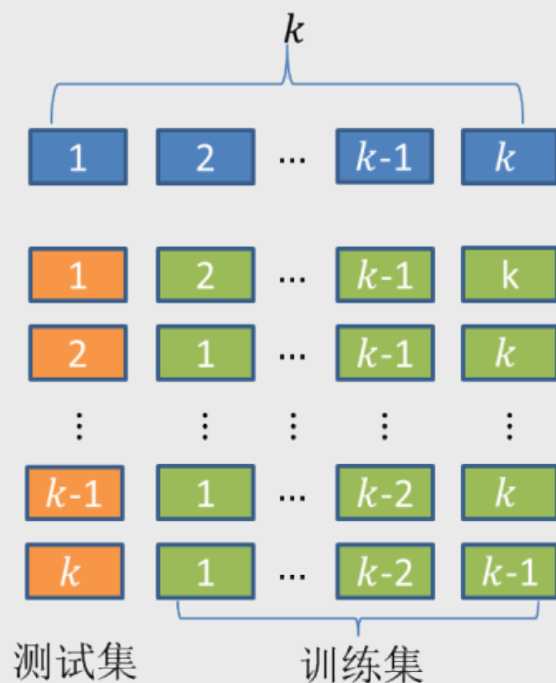
## 系数求解的几何意义



以二维空间为例（即自变量仅包含 $x_1$ 和 $x_2$ 两个），左半边的半椭圆体代表了 $\sum_{i=1}^n (y_i - \beta_0 - \sum_{j=1}^2 x_{ij}\beta_j)^2$ 的部分，它是关于两个系数的二次函数；圆柱体代表了 $\beta_1^2 + \beta_2^2 \leq t$ 的部分；

右半边为二维坐标下的映射图，对于线性回归模型来说，抛物面的中心黑点代表模型的最小二乘解，当附加 $\beta_1^2 + \beta_2^2 \leq t$ 时，抛物面与圆面构成的交点就是岭回归模型的系数解；

## $\lambda$ 值的确定--交叉验证法



### 交叉验证的思想

首先将数据集拆分成 $k$ 个样本量大体相当的数据组（如图中的第一行），并且每个数据组与其他组都没有重叠的观测；

然后从 $k$ 组数据中挑选 $k - 1$ 组数据用于模型的训练，剩下的一组数据用于模型的测试（如图中的第二行）；

以此类推，将会得到 $k$ 种训练集和测试集。在每一种训练集和测试集下，都会对应一个模型及模型得分（如均方误差）。

## $\lambda$ 值的确定--交叉验证法

```
RidgeCV(alphas=(0.1, 1.0, 10.0), fit_intercept=True, normalize=False,  
         scoring=None, cv=None)
```

**alphas** : 用于指定多个lambda值的元组或数组对象，默认该参数包含0.1、1和10三种值。

**fit\_intercept** : bool类型参数，是否需要拟合截距项，默认为True。

**normalize** : bool类型参数，建模时是否需要数据集做标准化处理，默认为False。

**scoring** : 指定用于评估模型的度量方法。

**cv** : 指定交叉验证的重数。

## 糖尿病数据的预测

```
# 构造不同的Lambda值
Lambdas = np.logspace(-5, 2, 200)
# 设置交叉验证的参数，对于每一个Lambda值，都执行10重交叉验证
ridge_cv = RidgeCV(alphas = Lambdas, normalize=True,
                    scoring='neg_mean_squared_error',
                    cv = 10)

# 模型拟合
ridge_cv.fit(X_train, y_train)
# 返回最佳的lambda值
ridge_best_Lambda = ridge_cv.alpha_
ridge_best_Lambda

out:
0.013509935211980266
```

## 糖尿病数据的预测

```
# 基于最佳的Lambda值建模
ridge = Ridge(alpha = ridge_best_alpha, normalize=True)
ridge.fit(X_train, y_train)
```

```
# 返回岭回归系数
pd.Series(index = ['Intercept'] + X_train.columns.tolist(),
          data = [ridge.intercept_] + ridge.coef_.tolist())
```

$$\begin{aligned} Y &= -323.11 + 6.21BMI + 0.93BP - 0.49S1 + 0.21S2 \\ &+ 0.03S3 + 4.21S4 + 51.69S5 + 0.38S6 \end{aligned}$$

```
out:
Intercept    -323.114952
BMI           6.208205
BP            0.927412
S1           -0.489199
S2            0.210826
S3            0.028643
S4            4.211265
S5           51.688690
S6            0.384038
```

岭回归模型解决线性回归模型中矩阵 $X'X$ 不可逆的办法是添加 $l_2$ 正则的惩罚项，但缺陷在于始终保留建模时的所有变量，无法降低模型的复杂度。对于此，Lasso回归采用了 $l_1$ 正则的惩罚项。

$$J(\beta) = \sum (y - X\beta)^2 + \lambda \|\beta\|_1 = \sum (y - X\beta)^2 + \sum \lambda |\beta|$$

## 凸优化的等价命题

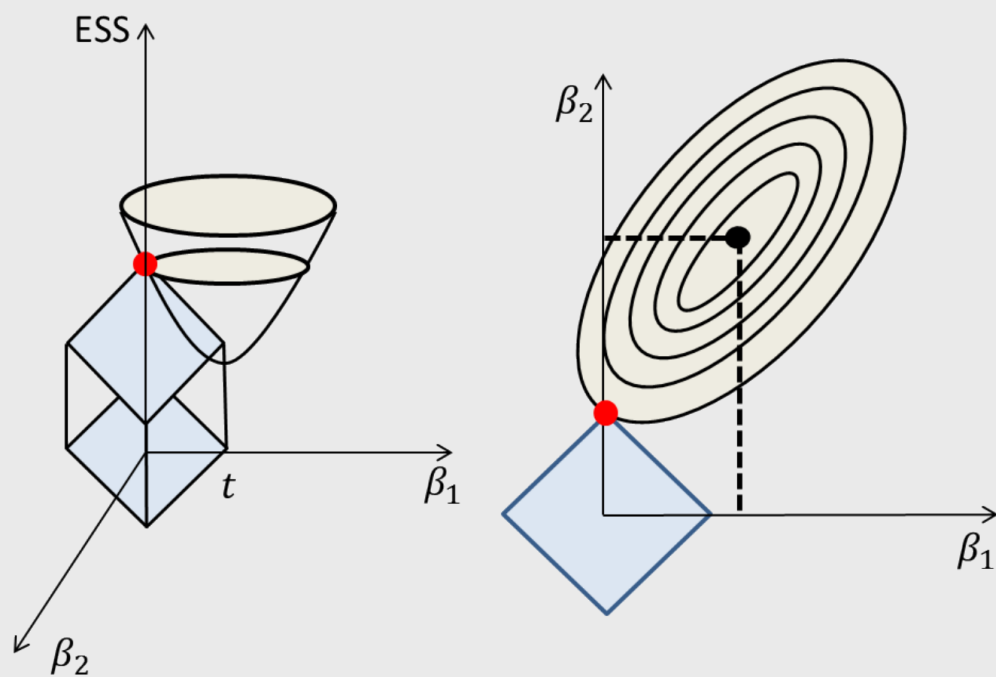
$$J(\beta) = \sum (y - X\beta)^2 + \lambda \|\beta\|_1 = \sum (y - X\beta)^2 + \sum \lambda |\beta|$$



$$\begin{cases} \operatorname{argmin} \left\{ \sum (y - X\beta)^2 \right\} \\ \text{附加约束 } \sum |\beta| \leq t \end{cases}$$



## 系数求解的几何意义



以二维空间为例，左半边的半椭圆体代表 $\sum_{i=1}^n (y_i - \beta_0 - \sum_{j=1}^2 x_{ij}\beta_j)^2$ 的部分，它是关于两个系数的二次函数；正方形则代表了 $|\beta_1| + |\beta_2| \leq t$ 的部分；

将LASSO回归的惩罚项映射到二维空间的话，就会形成“角”，一旦“角”与抛物面相交，就会导致 $\beta_1$ 为0，进而实现变量的删除。而且相比于圆面， $l_1$ 正则项的方框顶点更容易与抛物面相交，起到变量筛选的效果；

## $\lambda$ 值的确定--交叉验证法

```
LassoCV(alphas=None, fit_intercept=True, normalize=False,  
        max_iter=1000, tol=0.0001)
```

**alphas** : 指定具体的Lambda值列表用于模型的运算

**fit\_intercept** : bool类型参数, 是否需要拟合截距项, 默认为True

**normalize** : bool类型参数, 建模时是否需要对数据集做标准化处理, 默认为False

**max\_iter** : 指定模型最大的迭代次数, 默认为1000次

## 糖尿病数据的预测

```
# LASSO回归模型的交叉验证
lasso_cv = LassoCV(alphas = Lambdas, normalize=True, cv = 10, max_iter=10000)
lasso_cv.fit(X_train, y_train)

# 输出最佳的lambda值
lasso_best_alpha = lasso_cv.alpha_
lasso_best_alpha
```

**out:**  
0.062949889902218878

## 糖尿病数据的预测

```
# 基于最佳的lambda值建模
lasso = Lasso(alpha = lasso_best_alpha, normalize=True, max_iter=10000)
# 对“类”加以数据实体，执行回归系数的运算
lasso.fit(X_train, y_train)
# 返回LASSO回归的系数
pd.Series(index = ['Intercept'] + X_train.columns.tolist(),
          data = [lasso.intercept_] + lasso.coef_.tolist())
```

$$Y = -280.13 + 6.20BMI + 0.87BP - 0.14S1 - 0.49S3 + 44.82S5 + 0.33S6$$

**out:**

Intercept	-280.130832
BMI	6.199077
BP	0.865640
S1	-0.135059
S2	-0.000000
S3	-0.485755
S4	0.000000
S5	44.816615
S6	0.330531

# EDU

CSDN学院 IT实战派

