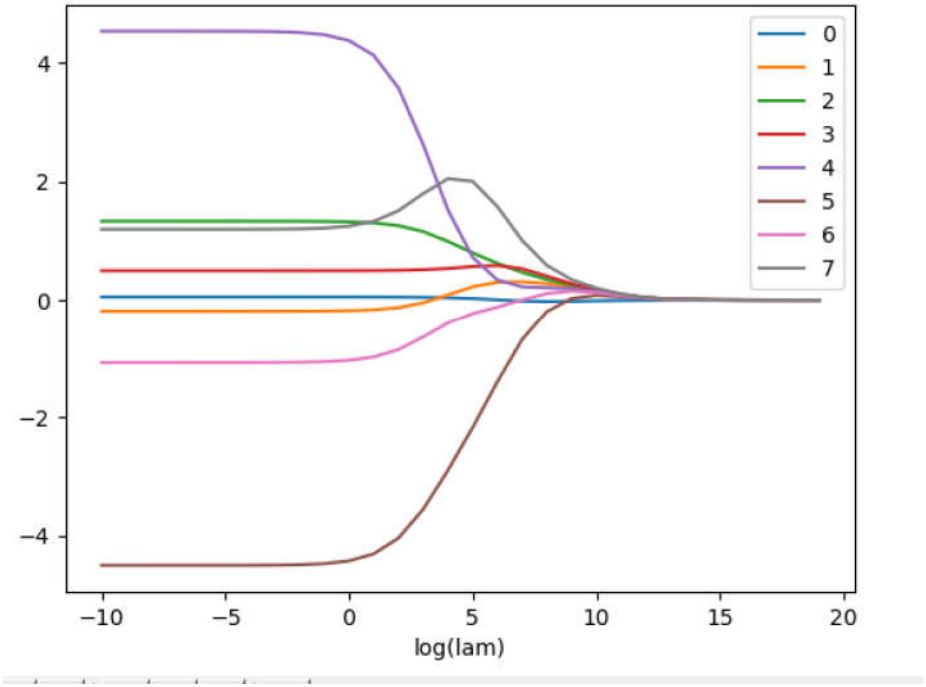


算法篇：岭回归

Ridge Regression

岭回归

在线性回归损失函数的基础上，增加了对权重的限制，作为正则化项

将**有限**的权重，放到**更重要的**特征维度上

$$L_{MSE}=\sum_{i=1}^{N}(y_i-\hat{y}_i)^2=(\mathbf{y}-\hat{\mathbf{y}})^T(\mathbf{y}-\hat{\mathbf{y}})$$ **线性回归**

$$L_{MSE}=\sum_{i=1}^{N}(y_i-\hat{y}_i)^2=(\mathbf{y}-\hat{\mathbf{y}})^T(\mathbf{y}-\hat{\mathbf{y}})+\lambda\|\mathbf{w}\|_2$$ **岭回归**

其中 $\|\mathbf{w}\|_2=\sum_{i=1}^{D}w_i^2=\mathbf{w}^{\mathrm{T}}\mathbf{w}$ **正则化项**

正则化项的作用：
使每个权重都不会过大
若 某个权重过大，则
自变量x一旦发生微小的改变
就会导致输出发生巨大改变
增大了过拟合的危险

牺牲了精度
提高了稳定性

根据拉格朗日方程:

$$L=(\mathbf{y}-\hat{\mathbf{y}})^T(\mathbf{y}-\hat{\mathbf{y}})+\lambda(\mathbf{w}^T\mathbf{w})$$

$$==\mathbf{y}^T\mathbf{y}-\mathbf{y}\mathbf{X}\mathbf{w}-\mathbf{w}^T\mathbf{X}^T\mathbf{y}+\mathbf{w}^T\mathbf{X}^T\mathbf{X}\mathbf{w}+\lambda\mathbf{w}^T\mathbf{w}$$

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

$$\mathbf{w}=\left(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I}\right)^{-1}\mathbf{X}^T\mathbf{y}$$

岭回归的权重计算方法

$$\mathbf{w}=\left(\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{X}^T\mathbf{y}$$

线性回归的权重计算方法

λ是一个可以选择的参数，
取值越大，数据对最终权重**w**的影响越小

随着λ 的增加，不重要的维度的权重会随之减少

避免了求拟不存在的现象

代码实现

```
import numpy as np
from numpy import matrix as mat
import matplotlib.pyplot as plt
```

```
def ridge_regress(X,Y,lam=0.2):
    # 数据转为mat格式
    N,D = np.shape(X)
    xMat = mat(X)
    yMat = mat(Y)
    xTx = xMat.T*xMat + mat(lam*np.eye(D))
    # 返回ws
    ws = xTx.I * (xMat.T * yMat)
    return np.array(ws)
```

$$\mathbf{w}=\left(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I}\right)^{-1}\mathbf{X}^T\mathbf{y}$$

根据不同的 λ 计算不同的 $\mathbf{w}$

```
def get_ws_by_lams(train_X,train_Y,lams):
    # 数据正则化
    mean_X= np.mean(train_X,axis=0,keepdims=True)
    std_X = np.std(train_X,axis=0,keepdims=True)
    X = (train_X-mean_X)/std_X

    mean_Y = np.mean(train_Y,axis=0,keepdims=True)
    Y= train_Y-mean_Y

    # 针对不同的lam求取不同的ws
    N,D = np.shape(train_X)
    N_ws = len(lams)
    save_ws = np.zeros((D,N_ws))
    for i,lam in enumerate(lams):
        ws = ridge_regress(X,Y,lam=lam)
        save_ws[:,i]=ws[:,0]
    return  save_ws
```

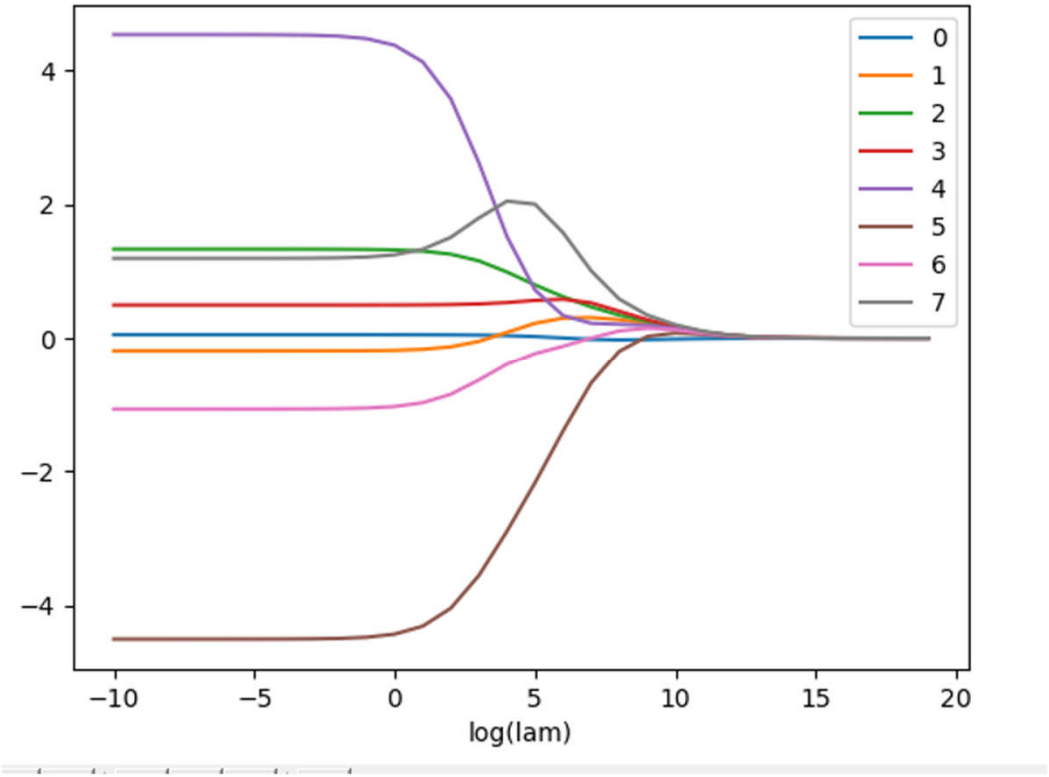
```python
if __name__=="__main__":
    # 真实数据测试:
    X,Y= load_DataSet('鲍鱼.txt',col_X=(0,1,2,3,4,5,6,7),col_Y=(8))

    # 测试1
    train_X = X[:99,:]
    train_Y = Y[:99,:]

    N_lam = 30
    lams = [ np.exp(i-10)  for i in range(N_lam)]

    save_ws = get_ws_by_lams(X,Y,lams)

    # 绘图
    fig = plt.figure() # 创建绘图对象
    ax = fig.add_subplot(1,1,1)
    N,D = np.shape(train_X)
    for i in range(D):
        ax.plot([(i-10)  for i in range(N_lam)],save_ws[i,:],label=str(i))
    ax.set_xlabel('log(lam)')
    plt.legend(loc='best')
    plt.show()
```



例子： 8维特征

随着 λ 的增加，每个特征维度所对应的权重逐渐减少

特征中1、3维的权重一直较低说明其不太重要

特征中4、5维的权重较大说明其比较重要

且特征4 起到正作用 特征5 起到反作用