监督学习-回归

ML18

礼欣
www.python123.org

岭回归及其应用实例

线性回归

对于一般地线性回归问题，参数的求解采用的是最小二乘法，其目标函数如下：

$$\boldsymbol{argmin}\|\boldsymbol{Xw}-\boldsymbol{y}\|^{\boldsymbol{2}}$$

参数w的求解，也可以使用如下矩阵方法进行：

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

线性回归

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

对于矩阵X，若某些列线性相关性较大（即训练样本中某些属性线性相关），就会导致$\boldsymbol{X}^{\boldsymbol{T}}\boldsymbol{X}$的值接近0，在计算$\left(\boldsymbol{X}^{\boldsymbol{T}}\boldsymbol{X}\right)^{-\mathbf{1}}$时就会出现不稳定性：

结论：传统的基于最小二乘的线性回归法缺乏稳定性。

岭回归

岭回归的优化目标：

$$\boldsymbol{argmin}\|\boldsymbol{Xw}-\boldsymbol{y}\|^{\boldsymbol{2}}+\boldsymbol{\alpha}\|\boldsymbol{w}\|^{\boldsymbol{2}}$$

对应的矩阵求解方法为：

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

岭回归

- 岭回归(ridge regression)是一种专用于共线性数据分析的有偏估计回归方法
- 是一种改良的最小二乘估计法，对某些数据的拟合要强于最小二乘法。

sklearn中的岭回归

在sklearn库中，可以使用sklearn.linear_model.Ridge调用岭回归模型，其主要参数有：

- alpha：正则化因子，对应于损失函数中的α
- fit_intercept：表示是否计算截距，
- solver：设置计算参数的方法，可选参数‘auto’、‘svd’、‘sag’等

交通流量预测实例

数据介绍：

数据为某路口的交通流量监测数据，记录全年小时级别的车流量。

实验目的：

根据已有的数据创建多项式特征，使用岭回归模型代替一般的线性模型，对车流量的信息进行多项式回归。

技术路线：`sklearn.linear_model.Ridgefrom sklearn.preprocessing.PolynomialFeatures`

数据实例

数据特征如下：

HR：一天中的第几个小时（0-23）

WEEK_DAY：一周中的第几天（0-6）

DAY_OF_YEAR：一年中的第几天（1-365）

WEEK_OF_YEAR：一年中的第几周（1-53）

TRAFFIC_COUNT：交通流量

全部数据集包含2万条以上数据（21626）

ID	HR	WEEK_DAY	DAY_OF_YEAR	WEEK_OF_YEAR	TRAFFIC_COUNT
79	0	3	1	1	31
79	1	3	1	1	20
79	2	3	1	1	21
79	3	3	1	1	7
79	4	3	1	1	7
79	5	3	1	1	12
79	6	3	1	1	5
79	7	3	1	1	11
79	8	3	1	1	10
79	9	3	1	1	12
79	10	3	1	1	5
79	11	3	1	1	12
79	12	3	1	1	9
79	13	3	1	1	10
79	14	3	1	1	24
79	15	3	1	1	26
79	16	3	1	1	20
79	17	3	1	1	21
79	18	3	1	1	27

实例程序编写

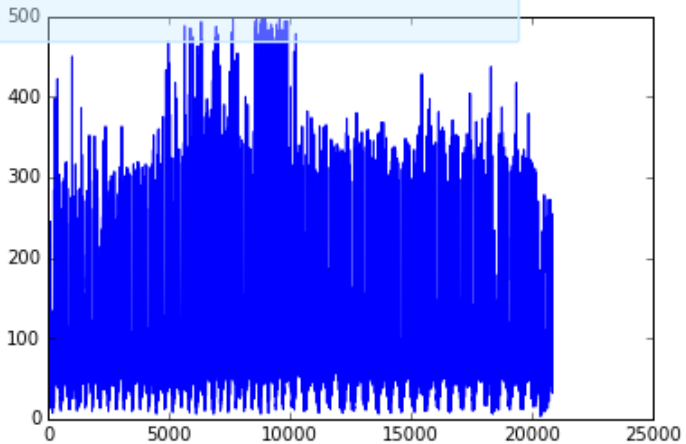
1. 建立工程，导入sklearn相关工具包：

```
>>> import numpy as np
>>> from sklearn.linear_model import Ridge
#通过sklearn.linermodel加载岭回归方法
>>> from sklearn import cross_validation
#加载交叉验证模块，加载matplotilib模块
>>> import matplotlib.pyplot as plt
>>> from sklearn.preprocessing import PolynomialFeatures
#通过。。加载。。。用于创建多项式特征，如ab、a²、b²
```

实例程序编写

2. 数据加载：

```
>>> data=np.genfromtxt('data.txt')
#使用numpy的方法从txt文件中加载数据
>>> plt.plot(data[:,4])
#使用plt展示车流量信息，如右图
```



实例程序编写

3. 数据处理：

```
>>> X=data[:,:4]
#X用于保存0-3维数据，即属性
>>> y=data[:,4]
#y用于保存第4维数据，即车流量
>>> poly=PolynomialFeatures(6)
#用于创建最高次数6次方的的多项式特征，多次试验后决定采用6次
>>> X=poly.fit_transform(X)
#X为创建的多项式特征
```

实例程序编写

4. 划分训练集和测试集：

```
>>> train_set_X, test_set_X , train_set_y, test_set_y =
      cross_validation.train_test_split(X,y,test_size=0.3,
                                        random_state=0)
#将所有数据划分为训练集和测试集，test_size表示测试集的比例，
#random_state是随机数种子
```

实例程序编写

5. 创建回归器，并进行训练：

```
>>> clf=Ridge(alpha=1.0,fit_intercept = True)
#接下来我们创建岭回归实例
>>> clf.fit(train_set_X,train_set_y)
#调用fit函数使用训练集训练回归器
>>> clf.score(test_set_X,test_set_Y)
#利用测试集计算回归曲线的拟合优度，clf.score返回值为0.7375
#拟合优度，用于评价拟合好坏，最大为1，无最小值，当对所有输入都输出同一个值时，拟合优度为0。
```

结果分析

6. 画出拟合曲线：

```
>>> start=200  #接下来我们画一段200到300范围内的拟合曲线
>>> end=300
>>> y_pre=clf.predict(X) #是调用predict函数的拟合值
>>> time=np.arange(start,end)
>>> plt.plot(time,y[start:end],'b', label="real")
>>> plt.plot(time,y_pre[start:end],'r', label='predict')
#展示真实数据（蓝色）以及拟合的曲线（红色）
>>> plt.legend(loc='upper left') #设置图例的位置
>>> plt.show()
```

结果分析

分析结论：预测值和实际值的走势大致相同