

波士顿房价数据集统计的是20世纪70年代中期波士顿郊区房价的中位数，统计了城镇人均犯罪率、不动产税等共计13个指标，506条房价数据，通过统计出的房价，试图能找到那些指标与房价的关系。

数据集变量代表的意义：

CRIM: 城镇人均犯罪率

ZN: 住宅用地所占比例

INDUS: 城镇中非住宅用地所占比例

CHAS: 虚拟变量,用于回归分析

NOX: 环保指数

RM: 每栋住宅的房间数

AGE: 1940 年以前建成的自住单位的比例

DIS: 距离 5 个波士顿的就业中心的加权距离

**RAD**: 距离高速公路的便利指数  
**TAX**: 每一万美元的不动产税率  
**PTRATIO**: 城镇中的教师学生比例  
**B**: 城镇中的黑人比例  
**LSTAT**: 地区中有多少房东属于低收入人群  
**MEDV**: 自住房屋房价中位数（也就是均价）

**实验目的**：直接调用scikit-learn中的相关函数，实现波士顿房价简单预测的代码如下，主要步骤包括：

1.加载数据

2.划分训练集和测试集

3.创建线性回归模型

4.拟合训练数据

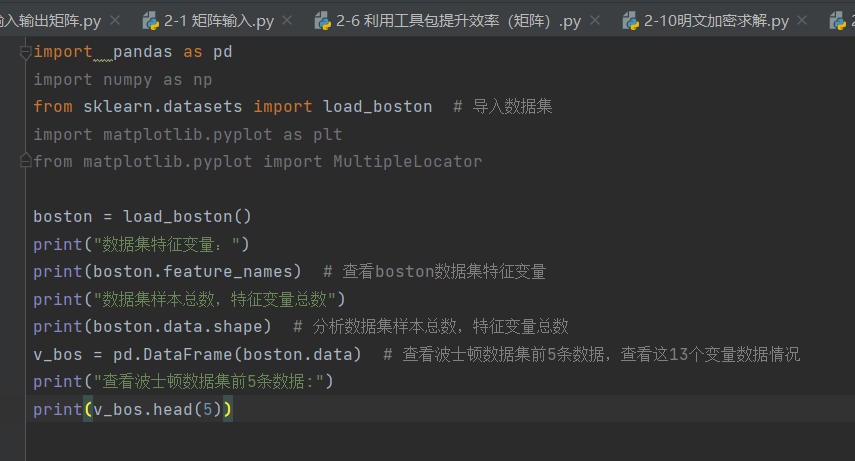
5.得到预测结果

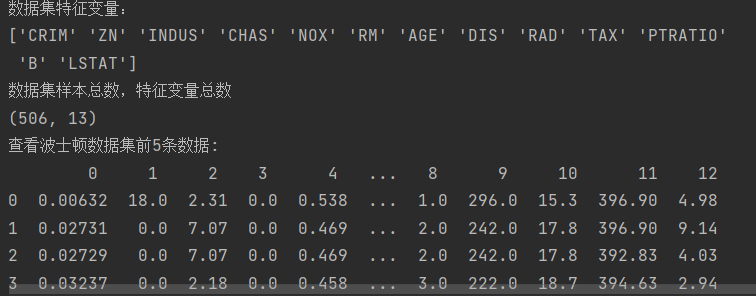
6.计算相应的评测指标

1. 数据分析：

首先导入数据集，对数据进行分析

根据程序输出结果，查看数据集数据样本总数，与特征变量个数；以及通过数据集前5条数据，查看13个特征变量数据情况。

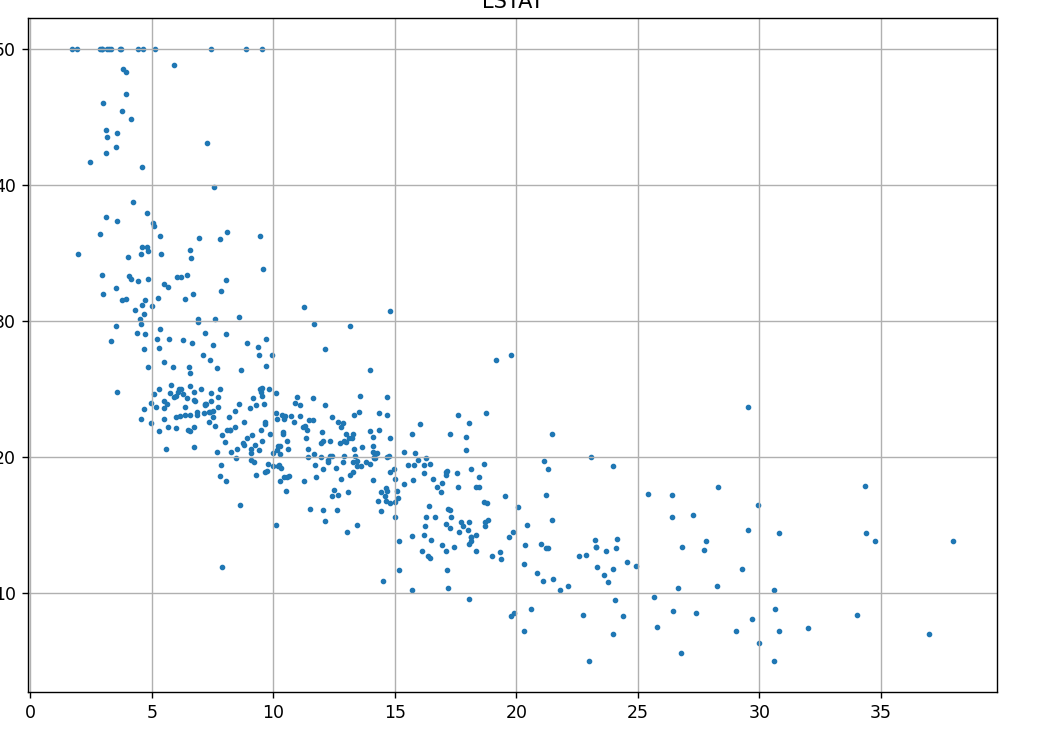
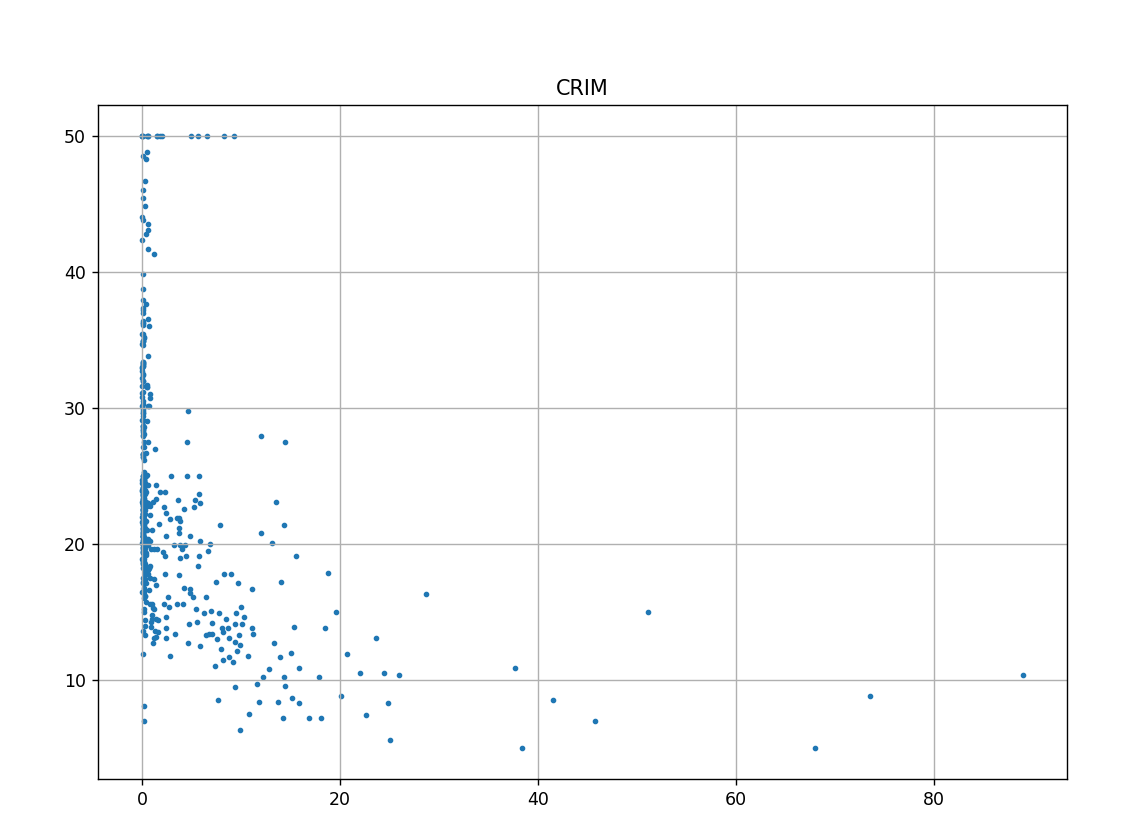


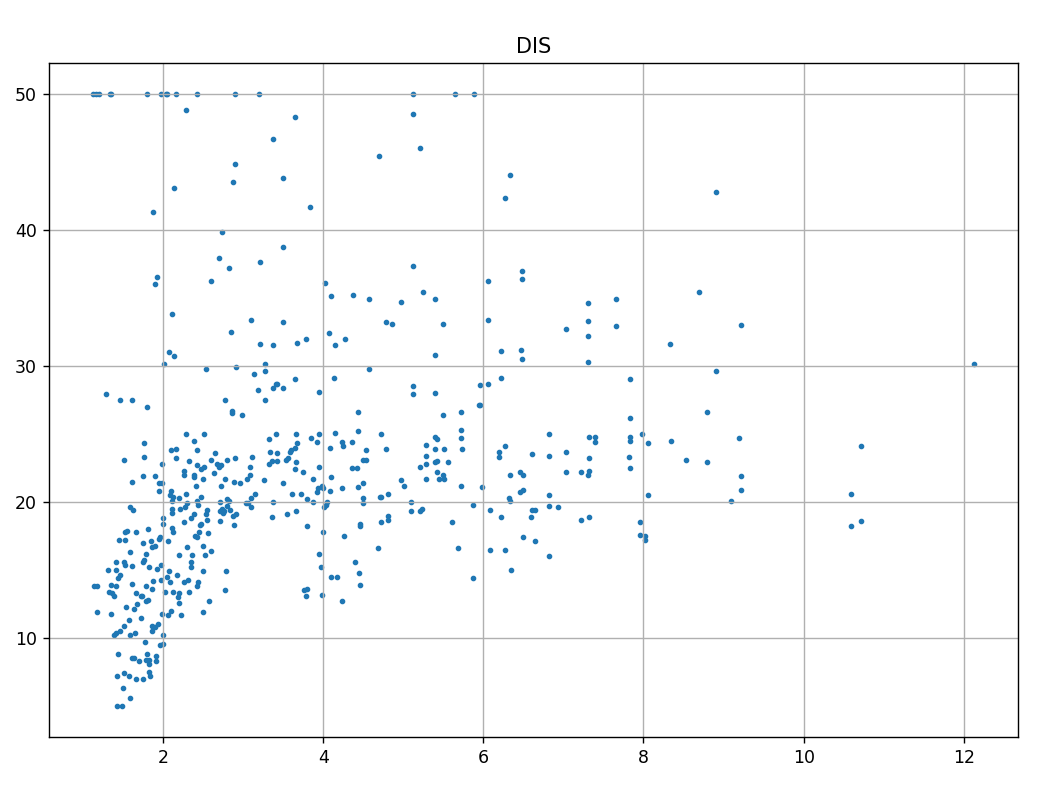


1. 数据处理

对自变量进行特征分析，并画出[散点图](https://so.csdn.net/so/search?q=%E6%95%A3%E7%82%B9%E5%9B%BE&spm=1001.2101.3001.7020)，分析特征变量与房价之间的相关性

例：







**数据处理以及可视化代码：**

import pandas as pd

import numpy as np

from sklearn.datasets import load\_boston # 导入数据集

import matplotlib.pyplot as plt

from matplotlib.pyplot import MultipleLocator

"""

第一步：首先认识波士顿数据集，分析查看数据集样本总数，特征变量总数。

第二步：然后画出波士顿数据集所有特征变量的散点图，并分析特征变量与房价的影响关系。

"""

boston = load\_boston()

print(boston.feature\_names) # 查看boston数据集特征变量

print(boston.data.shape) # 分析数据集样本总数，特征变量总数

v\_bos = pd.DataFrame(boston.data) # 查看波士顿数据集前5条数据，查看这13个变量数据情况

print(v\_bos.head(5))

x = boston['data'] # 导入特征变量

y = boston['target'] # 导入目标变量房价

student = input('房价特征信息图--0；各个特征信息图--1: ') # 输入0代表查看影响房价特征信息图，输入1代表查看各个特征信息图

if str.isdigit(student):

b = int(student)

if (b <= 1):

print('开始画图咯...', end='\t')

if (b == 0):

plt.figure(figsize=(20, 15))

y\_major\_locator = MultipleLocator(5) # 把y轴的刻度间隔设置为10，并存在变量里

ax = plt.gca() # ax为两条坐标轴的实例

ax.yaxis.set\_major\_locator(y\_major\_locator) # 把y轴的主刻度设置为5的倍数

plt.ylim(0, 51)

plt.grid()

for i in range(len(y)):

plt.scatter(i, y[i], s=20)

plt.show()

else:

name = boston['feature\_names']

for i in range(13):

plt.figure(figsize=(10, 7))

plt.grid()

plt.scatter(x[:, i], y, s=5) # 横纵坐标和点的大小

plt.title(name[i])

print(name[i], np.corrcoef(x[:i]), y)

plt.show()

else:

print('同学请选择0或者1')

else:

print('同学请选择0或者1')

1. 通过导入线性回归模型搭建波士顿房价预测模型

import numpy as np

from skimage.metrics import mean\_squared\_error

from sklearn import linear\_model

from sklearn.linear\_model import LinearRegression # 导入线性模型

from sklearn.datasets import load\_boston # 导入数据集

from sklearn.metrics import r2\_score # 使用r2\_score对模型评估

from sklearn.model\_selection import train\_test\_split # 导入数据集划分模块

import matplotlib.pyplot as plt

import matplotlib.pyplot as plt2

boston = load\_boston()

x = boston['data'] # 影响房价的特征信息数据

y = boston['target'] # 房价

name = boston['feature\_names']

# 数据处理

unsF = [] # 次要特征下标

for i in range(len(name)):

if name[i] == 'RM' or name[i] == 'PTRATIO' or name[i] == 'LSTAT' or name[i] == 'AGE' or name[i] == 'NOX' or name[i] == 'DIS' or name[i] == 'INDUS':

continue

unsF.append(i)

x = np.delete(x, unsF, axis=1) # 删除次要特征

unsT = [] # 房价异常值下标

for i in range(len(y)):

if y[i] > 50: # 对房价影响较小的特征信息进行剔除

unsT.append(i)

x = np.delete(x, unsT, axis=0) # 删除样本异常值数据

y = np.delete(y, unsT, axis=0) # 删除异常房价

# 将数据进行拆分，一份用于训练，一份用于测试和验证

# 测试集大小为30%,防止过拟合

# 这里的random\_state就是为了保证程序每次运行都分割一样的训练集和测试集。

# 否则，同样的算法模型在不同的训练集和测试集上的效果不一样。

x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size=0.3, random\_state=0)

# 线性回归模型

lf = LinearRegression()

lf.fit(x\_train, y\_train) # 训练数据,学习模型参数

y\_predict = lf.predict(x\_test) # 预测

# 与验证值作比较

error = mean\_squared\_error(y\_test, y\_predict).round(5) # 平方差

score = r2\_score(y\_test, y\_predict).round(5) # 相关系数

# 绘制真实值和预测值的对比图

fig = plt.figure(figsize=(13, 7))

plt.rcParams['font.family'] = "sans-serif"

plt.rcParams['font.sans-serif'] = "SimHei"

plt.rcParams['axes.unicode\_minus'] = False # 绘图

plt.plot(range(y\_test.shape[0]), y\_test, color='red', linewidth=1, linestyle='-')

plt.plot(range(y\_test.shape[0]), y\_predict, color='blue', linewidth=1, linestyle='dashdot')

plt.legend(['真实值', '预测值'])

plt.title("学号", fontsize=20)

error = "标准差d=" + str(error)+"\n"+"相关指数R^2="+str(score)

plt.xlabel(error, size=18, color="black")

plt.grid()

plt.show()

plt2.rcParams['font.family'] = "sans-serif"

plt2.rcParams['font.sans-serif'] = "SimHei"

plt2.title('学号', fontsize=24)

xx = np.arange(0, 40)

yy = xx

plt2.xlabel('\* truth \*', fontsize=14)

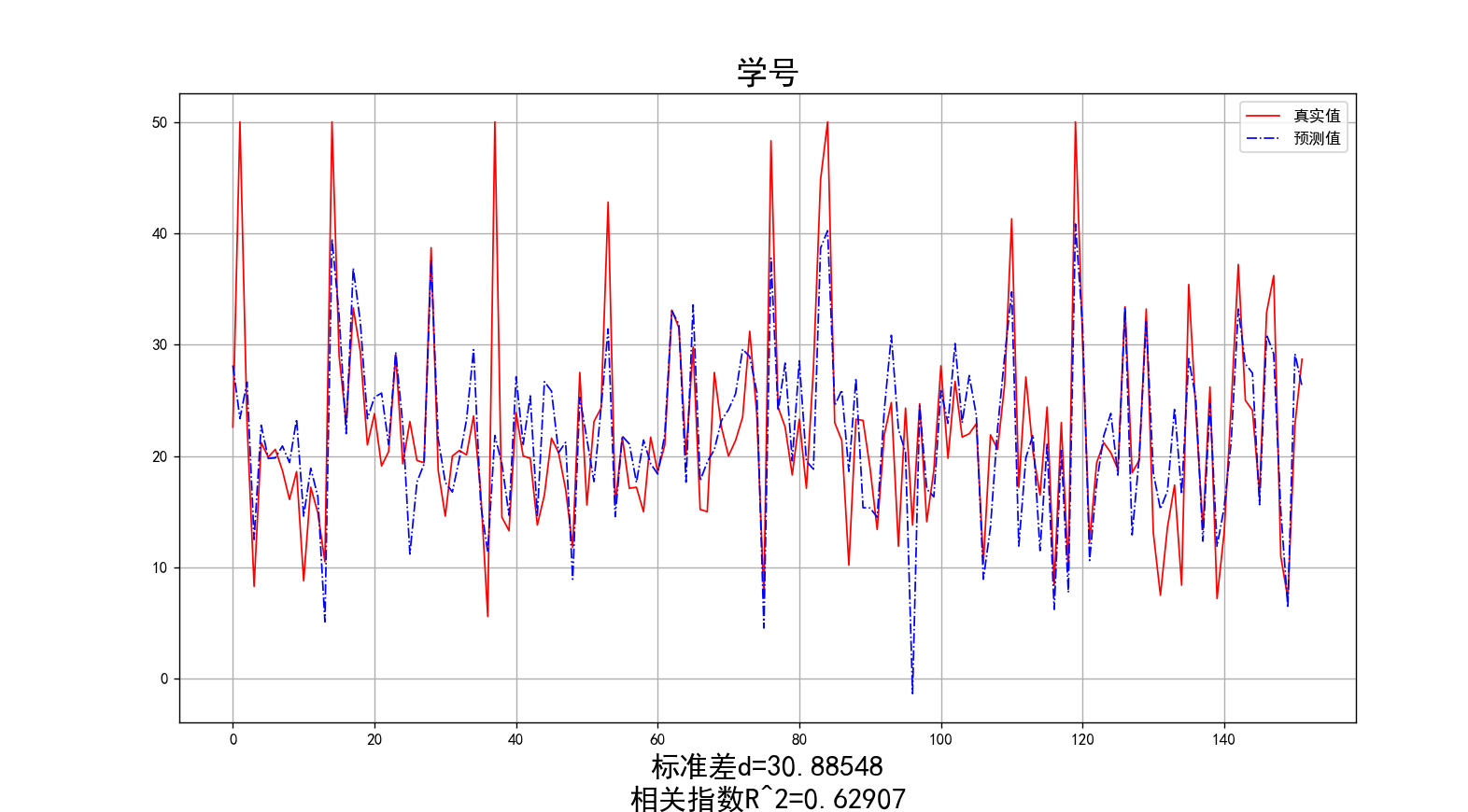
plt2.ylabel('\* predict \*', fontsize=14)

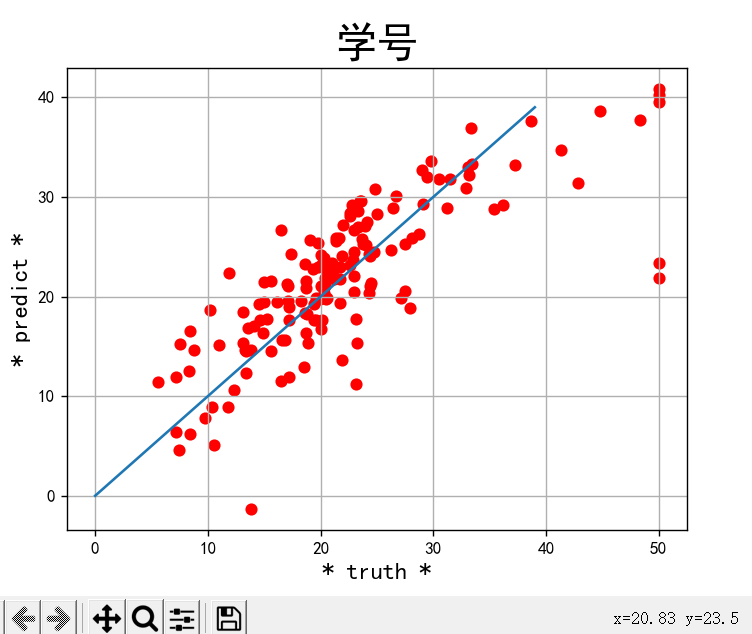
plt2.plot(xx, yy)

plt2.scatter(y\_test, y\_predict, color='red')

plt2.grid()

plt2.show()





1. 总结：利用sklearn工具包，利用多种算法以及图表方式进行整合。

问题在于，做不来，上课就算听课了，和我做不做得来有啥关系呢

解决方案:

仔细观看B站机器学习视频、教程，波士顿预测相关视频

认真研读CSDN上波士顿预测相关的文章