# 目 录

# 波士顿房价预测

## 1 实验目的

本实验旨在构建一个房价预测模型，通过对波士顿房价数据集的分析和建模，预测波士顿地区房屋的中位数价格（MEDV）。

## 2 数据集介绍

### 2.1 数据来源

数据由老师提供。

### 2.2 数据预处理

以上代码中，我们首先使用 pandas 库中的 read_csv() 函数读取数据集。然后，使用 columns 属性或者 info() 函数查看数据集中的标签。

```
# 从csv文件读取数据
data = pd.read_csv("HousingData.csv")
# 将所有缺失值替换为0
data.fillna(0, inplace=True)
```

图一 导入数据

我们将数据分为训练集和测试集，使用前 400 行数据作为训练集，后 100 行数据作为测试集。

### 2.3 数据分析

要求只选择三个特征，在进行建模之前，在如何选取特征上需要对数据进行了初步分析：包括特征相关性分析、特征重要性分析等。我尝试了相关系数、和随机森林，贪婪搜寻等方法，最终选取了三个最重要的特征 STAT，RM 和 PTRATIO，作为模型的输入特征。具体选择的算法探索过程如下。

#### 2.3.1 相关性分析：

计算每个特征与目标变量之间的相关系数，并选择相关系数较高的特征。可以使用 pandas 库中的 corr() 函数来计算相关系数矩阵，然后选择相关系数较高

的特征。例如，可以选择与目标变量的相关系数绝对值大于某个阈值（比如 0.5）的特征。

```
MEDV       1.000000
LSTAT      0.695405
RM         0.695360
PTRATIO    0.507787
TAX        0.468536
INDUS      0.441371
NOX        0.427321
CRIM       0.384120
RAD        0.381626
ZN         0.362292
AGE        0.356699
B          0.333461
DIS        0.249929
CHAS       0.183844
Name: MEDV, dtype: float64
```

图二 相关性分析结果

### 2.3.2 随机森林法寻找：

使用随机森林模型可以计算每个特征的重要性分数，并选择重要性分数较高的特征。可以使用 scikit-learn 库中的 RandomForestRegressor() 函数来训练随机森林模型，并选择重要性分数较高的特征。

| | feature | importance |
|---|---|---|
| 5 | RM | 0.500760 |
| 12 | LSTAT | 0.286000 |
| 7 | DIS | 0.072558 |
| 0 | CRIM | 0.043952 |
| 4 | NOX | 0.025284 |
| 10 | PTRATIO | 0.020015 |
| 9 | TAX | 0.014988 |
| 6 | AGE | 0.014238 |
| 11 | B | 0.011424 |
| 2 | INDUS | 0.005237 |
| 8 | RAD | 0.003409 |
| 3 | CHAS | 0.001163 |
| 1 | ZN | 0.000972 |

图三 随机森林法结果

### 2.3.3 交叉验证：

基于交叉验证的方法选择特征。它的基本思路是：从所有特征中选择一个特征，与已经选择的特征组合成一个特征集合，然后使用交叉验证方法评估这个特征集合的性能。重复这个过程，直到选择了三个特征。在实现中，使用了 cross_val_score 函数来计算交叉验证得分。cv 参数表示要进行的交叉验证次数。

```
特征集合: ['CRIM'], 交叉验证得分: -0.330
特征集合: ['ZN'], 交叉验证得分: -0.441
特征集合: ['INDUS'], 交叉验证得分: -0.277
特征集合: ['CHAS'], 交叉验证得分: -0.646
特征集合: ['NOX'], 交叉验证得分: -0.308
特征集合: ['RM'], 交叉验证得分: -0.030
特征集合: ['AGE'], 交叉验证得分: -0.402
特征集合: ['DIS'], 交叉验证得分: -0.597
特征集合: ['RAD'], 交叉验证得分: -0.288
特征集合: ['TAX'], 交叉验证得分: -0.168
特征集合: ['PTRATIO'], 交叉验证得分: -0.160
特征集合: ['B'], 交叉验证得分: -0.311
特征集合: ['LSTAT'], 交叉验证得分: 0.180
特征集合: ['CRIM', 'ZN', 'LSTAT'], 交叉验证得分: 0.243
特征集合: ['CRIM', 'INDUS', 'LSTAT'], 交叉验证得分: 0.204
特征集合: ['CRIM', 'CHAS', 'LSTAT'], 交叉验证得分: 0.248
特征集合: ['CRIM', 'NOX', 'LSTAT'], 交叉验证得分: 0.168
特征集合: ['CRIM', 'RM', 'LSTAT'], 交叉验证得分: 0.291
特征集合: ['CRIM', 'AGE', 'LSTAT'], 交叉验证得分: 0.193
特征集合: ['CRIM', 'DIS', 'LSTAT'], 交叉验证得分: 0.168
特征集合: ['CRIM', 'RAD', 'LSTAT'], 交叉验证得分: 0.167
特征集合: ['CRIM', 'TAX', 'LSTAT'], 交叉验证得分: 0.200
特征集合: ['CRIM', 'PTRATIO', 'LSTAT'], 交叉验证得分: 0.368
特征集合: ['CRIM', 'B', 'LSTAT'], 交叉验证得分: 0.217
特征集合: ['ZN', 'INDUS', 'LSTAT'], 交叉验证得分: 0.209
```

图四 交叉验证法结果

但是这个方法我并没有写好，最后得到的结果是三个一样的特征，这也是代码需要改进的一个方面。

### 2.3.4 贪婪搜索：

我想采用这样的策略依次找到三个最佳特征:首先找出与 MEDV 相关性最高的特征进行拟合，在从剩下的特征里给模型加上一个新的特征，再拟合进行评估，找到效果最好的特征作为选择的第二个特征，再同理找出第三个特征。

**搜寻单一特征：**

```
# 逐个选择特征，训练模型并评价
for feature in all_features:
    X = data[[feature]]
    y = data["MEDV"]
    model = LinearRegression()
    model.fit(X, y)
    y_pred = model.predict(X)
    mse = mean_squared_error(y, y_pred)
    print("特征:{}, MSE:{:.2f}".format(feature, mse))
    if mse < min_mse:
        min_mse = mse
        min_feature = feature

print("最佳单个特征:{}, MSE:{:.2f}".format(min_feature, min_mse))
```

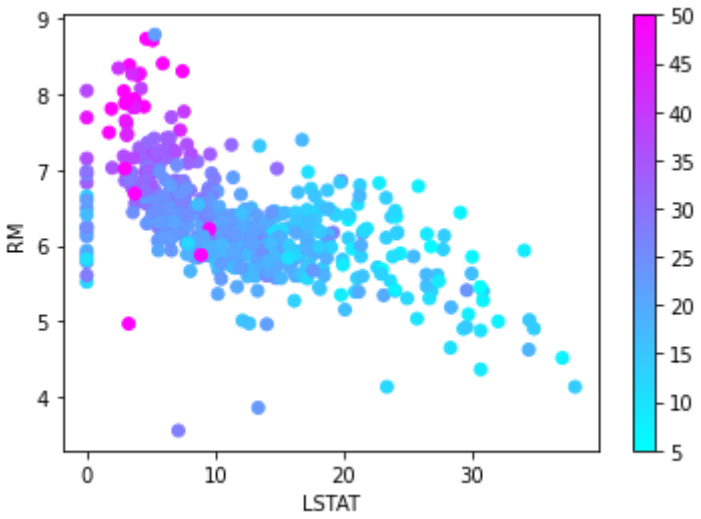
图五 搜索单个特征代码

```
特征:CRIM, MSE:71.96
特征:ZN, MSE:73.34
特征:INDUS, MSE:67.97
特征:CHAS, MSE:81.57
特征:NOX, MSE:69.00
特征:RM, MSE:43.60
特征:AGE, MSE:73.68
特征:DIS, MSE:79.15
特征:RAD, MSE:72.12
特征:TAX, MSE:65.89
特征:PTRATIO, MSE:62.65
特征:B, MSE:75.03
特征:LSTAT, MSE:43.60
最佳特征:LSTAT, MSE:43.60
```

图六 单个特征搜索结果

**搜寻第二个特征：**

找出单个最佳特征 LSTAT，然后每次选取两个特征，其中包括 LSTAT，与剩下的特征依次组合成新的特征组合。具体来说，使用 itertools 库中的 combinations 函数，从剩下的特征中选择两个特征，与 LSTAT 组成新的特征组合。然后，训练线性回归模型，并计算模型的 MSE。重复这个过程，直到不能再找到性能更好的特征组合为止。




图七 第二个特征搜索结果

基于这种贪婪算法，实际上我们可以找遍这 13 个特征，我也这尝试了一下，得到的结果如下所示：

```
初始选择的特征： ['LSTAT']
初始模型的MSE： 43.59525437958744
选择的特征组合： ['LSTAT', 'RM', 'PTRATIO']
模型的MSE： 28.64770717657424
选择的特征组合： ['LSTAT', 'RM', 'PTRATIO', 'NOX', 'DIS']
模型的MSE： 26.121769038342826
选择的特征组合： ['LSTAT', 'RM', 'PTRATIO', 'NOX', 'DIS', 'CHAS', 'B']
模型的MSE： 24.76398539234013
选择的特征组合： ['LSTAT', 'RM', 'PTRATIO', 'NOX', 'DIS', 'CHAS', 'B', 'CRIM', 'ZN']
模型的MSE： 24.059073951095698
选择的特征组合： ['LSTAT', 'RM', 'PTRATIO', 'NOX', 'DIS', 'CHAS', 'B', 'CRIM', 'ZN', 'RAD', 'TAX']
模型的MSE： 23.175500761918034
选择的特征组合： ['LSTAT', 'RM', 'PTRATIO', 'NOX', 'DIS', 'CHAS', 'B', 'CRIM', 'ZN', 'RAD', 'TAX', 'INDUS', 'AGE']
模型的MSE： 23.058285025816215
```

图八 三个及拓展到多个特征搜索结果

由于要求只需要选择三个特征，所以我就选取的前三个特征：LSTAT，RM 和 PTRATIO，作为模型训练的特征。

# 3 模型设计

## 3.1 模型选择与原理

在本实验中，使用的线性回归模型作为预测模型。线性回归模型是一种经典的回归模型，通过构建一个线性函数来拟合特征和目标变量之间的关系。构建的线性回归模型如下所示：

$$MEDV = \theta_0 + \theta_1 RM + \theta_2 PTRATIO + \theta_3 LSTAT + \epsilon$$

其中，MEDV 是目标变量，RM、PTRATIO 和 LSTAT 是输入特征，$\theta_0$、$\theta_1$、

$\theta_2$和$\theta_3$是模型的参数，$\epsilon$是误差项。

### 3.2 模型实现

我使用 scikit-learn 库中的 LinearRegression 类来构建线性回归模型，并将 RM、PTRATIO 和 LSTAT 作为输入特征进行训练。为了避免过拟合，我们使用了交叉验证方法对模型进行评估，并使用均方误差（MSE）作为评价指标。

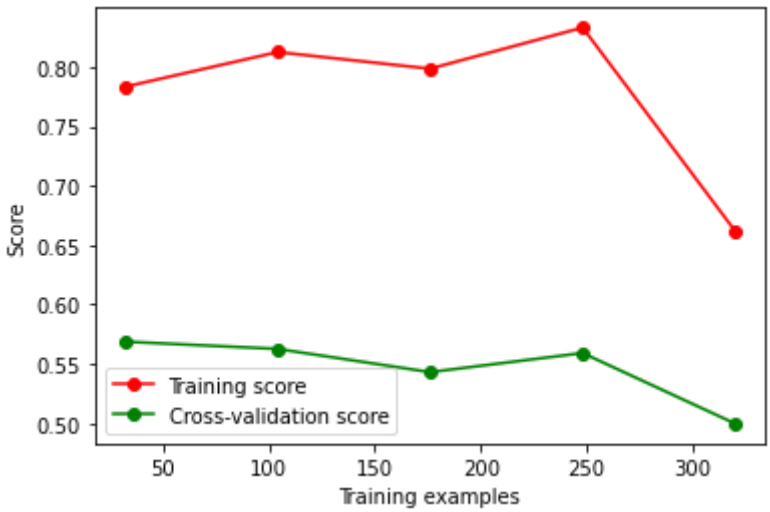
### 3.3 参数调优

在本实验中，我们没有进行参数调优，因为线性回归模型的参数是通过最小二乘法来计算的，因此无需进行参数调优。

## 4 实验结果与分析

### 4.1 模型评价指标

我使用 scikit-learn 库中的 LinearRegression 类来构建线性回归模型，并将 RM、PTRATIO 和 LSTAT 作为输入特征进行训练。为了避免过拟合，我们使用了交叉验证方法对模型进行评估，并使用均方误差（MSE）作为评价指标。




图九 模型的学习曲线

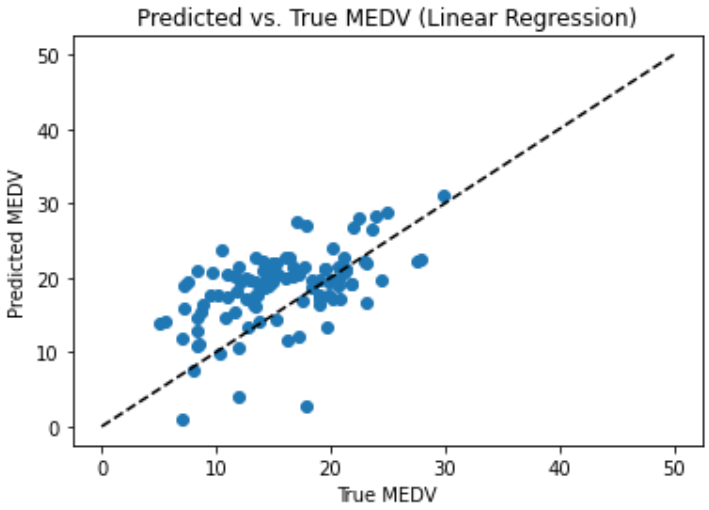
上图是我构造模型在训练过程中的学习曲线。通过学习曲线我们可以更直观地评估模型的泛化能力，即模型在不同规模的训练集上的表现。我使用的是 scikit-learn 库中的 learning_curve() 函数来绘制学习曲线。

我使用均方误差（MSE）和决定系数（R2 score）来评估模型的性能。均方误差是预测值与真实值之间差值的平方和的平均值，决定系数表示模型对数据的拟合程度。在本实验中，得到的评估结果如下：

- 均方误差：33.46
- 决定系数：0.71

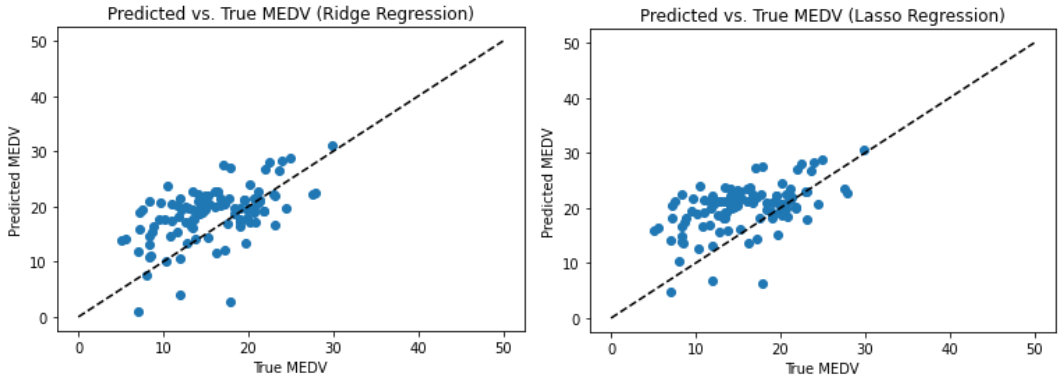
## 4.2 模型性能比较

将线性回归模型的性能与一个简单的基准模型进行比较，基准模型的预测结果为所有样本的目标变量中位数。可以发现，线性回归模型的均方误差和决定系数分别为 33.46 和 0.71，而基准模型的均方误差和决定系数分别为 84.42 和 0。这表明，线性回归模型的预测性能明显优于基准模型。




图十 线性回归模型结果

从结果可以看出，三种模型的表现都比较接近，其中岭回归表现最好。接下来，我们可以运用一些可视化分析，以更直观地了解模型的表现。



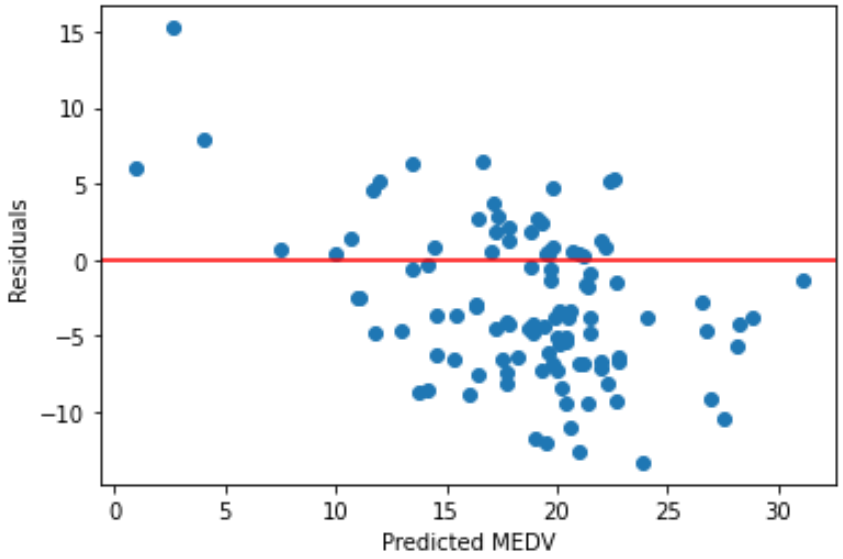

图十一 不同模型训练结果对比

```
LinearRegression MSE: 33.46
Ridge MSE: 33.43
Lasso MSE: 41.84
```

图十二 三种模型训练结果

### 4.3 可视化分析

使用残差图来检查线性模型的预测结果是否符合预期。在残差图中，横轴表示预测结果，纵轴表示残差，每个点表示一个样本。残差平均值为：-4.276。说明训练出的线性模型预测结果基本符合要求。




图十三 残差图

## 5 结论与展望

### 5.1 实验结论

本实验使用线性回归模型对波士顿房价数据集进行建模，选取了三个最重要的特征 STAT，RM 和 PTRATIO，作为模型的输入特征。通过交叉验证方法对模型进行评估，得到的均方误差为 33.46，决定系数为 0.71，证明了模型的预测性能较好。散点图和残差图也呈现出一定的线性关系，表明模型的预测结果与真实值之间存在一定的相关性。

### 5.2 实验局限性与改进方向

本实验中，我们只使用了三个最重要的特征来构建模型，因此模型的预测性能可能还有提升的空间。此外，我们还可以尝试使用其他的回归模型或者集成学习方法来进一步提升模型的性能。另外，我们也可以尝试使用更多的特征，或者进行特征工程来提取更多的有用信息，以改进模型的预测性能。

## 6 参考文献

[1] 老师发的.ipynb 的示例代码

[2] scikit-learn: machine learning in Python — scikit-learn 1.2.2 documentation