模式识别与统计学习实验二：线性回归

张力群 2020080301001

一、 实验原理

1.1 问题描述

波士顿房价问题是一个典型的线性回归问题。问题描述如下：给定一组房屋的特征变量，包括城市人均犯罪率、住宅平均房间数、离职中心的加权距离等，预测该房屋的房价。

解决该问题的方法如下：

1. 收集波士顿房价的数据集，可以使用 scikit_learn库提供的 load_boston() 函数加载数据集。
2. 数据集预处理，主要包括数据标准化、数据划分等。由于线性回归模型对数据集的分布具有一定的假设，因此需要将数据集进行标准化处理，使其符合标准正态分布。同时，还需要将数据集划分为训练集和测试集。
3. 构建线性回归模型。使用 $LinearRegression()$类来构建模型。在模型训练过程中，使用训练集对模型进行拟合，并计算模型的评估指标，例如均方误差（MSE）、平均绝对误差（MAE）等。
4. 利用模型对测试集进行预测，并计算模型的评估指标。
5. 对模型进行优化，可以采用一些优化技术，例如：添加多项式项、正则化等。通过比较不同模型的评估指标，选择最优的模型。
6. 最后，可以使用优化过的模型对新的波士顿房屋数据进行预测，并给出房价的估计值。

1.2 线性回归原理

线性回归是一种用于解决连续型变量预测问题的监督学习算法。它的基本思想是建立一个关于自变量和因变量的线性模型，用样本数据训练出最佳参数，然后利用该模型对新的样本进行预测。

线性回归的核心就是它的线性假设：假设自变量和因变量之间存在线性关系，即因变量Y可以表示为K个自变量X1到XK的线性组合，形式化地表示为

$$Y = b0 + b1 \times X1 + b2 \times X2 + \cdots + bx \times Xk$$

其中，$b0$到bk是模型的参数，称为回归系数。

线性回归的目标是最小化所有样本的预测误差，也就是最小化残差平方和RSS（$ResidualSumofSquares$）

$$RSS = \Sigma i = 1^n (y_i - \hat{y}_i)^2$$

其中，yi表示第i个样本的真实值，而$\hat{y}i$则表示模型对第i个样本的预测值。

为了求解最佳的回归系数b0到bk，通常使用线性代数中的最小二乘法（OLS），通过对样本数据进行“拟合”来求解。最小二乘法的思想是，通过最小化残差平方和RSS，来寻找最合适的回归系数，使得模型可以在样本数据上得到最优的拟合。

在求解完回归系数之后，线性回归模型就可以用来对新的样本进行预测。对于一个新的样本，只需要将自变量代入回归方程中，就可以得到预测值。通常使用均方根误差（$RMSE$）或平均绝对误差（MAE）等指标来评估模型预测的准确性。

需要注意的是，在实际应用中，线性回归模型可能会面临欠拟合（模型过于简单）或过拟合（模型过于复杂）等问题，需要使用一些技巧来解决，例如添加多项式项、正则化等。

二、 python 代码实现

2.1 liner_regression

```python
import numpy as np

class LinearRegression:
    def __init__(self, learning_rate=0.01, n_iterations=1000):
        self.learning_rate = learning_rate
        self.n_iterations = n_iterations
        self.weights = None
        self.bias = None

    def fit(self, X, y):
        # 初始化参数
        n_samples, n_features = X.shape
        self.weights = np.zeros(n_features)
        self.bias = 0

        # 梯度下降
        for _ in range(self.n_iterations):
            y_predicted = np.dot(X, self.weights) + self.bias

            # 计算梯度
            dw = (1 / n_samples) * np.dot(X.T, (y_predicted - y))
            db = (1 / n_samples) * np.sum(y_predicted - y)

            # 更新参数
            self.weights -= self.learning_rate * dw
            self.bias -= self.learning_rate * db

    def predict(self, X):
        y_predicted = np.dot(X, self.weights) + self.bias
        return y_predicted
```

- 首先，我们定义了一个 LinearRegression 类，用于实现线性回归模型。在类的初始化函数中，我们定义了学习率和迭代次数两个超参数，用于控制模型的训练过程。
- 在类中，我们定义了 fit 和 predict 两个方法，分别用于训练模型和预测。
- 在 fit 方法中，我们首先初始化参数，然后使用梯度下降法来更新参数，直到达到指定的迭代次数。
- 在 predict 方法中，我们只需要将自变量代入回归方程中，就可以得到预测值。

2.2 main

```
# 梯度下降的线性回归，利用boston房价数据集

import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import fetch_openml
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from linear_regression import LinearRegression


# 加载波士顿房价数据集
boston = fetch_openml(name='boston', version=1)
X = boston.data
y = boston.target

# 划分数据集
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=1234)

# 特征缩放
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)

# 初始化模型
regressor = LinearRegression(learning_rate=0.01, n_iterations=1000)

# 训练模型
regressor.fit(X_train, y_train)

# 预测测试集
predicted = regressor.predict(X_test)

# 计算R^2
def r2_score(y_true, y_predicted):
    corr_matrix = np.corrcoef(y_true, y_predicted)
    corr = corr_matrix[0, 1]
    return corr ** 2

r2 = r2_score(y_test, predicted)
print("R^2:", r2)
```

- 加载波士顿房价数据集，然后将数据集划分为训练集和测试集。
- 对训练集和测试集的特征进行了缩放。
- 初始化了一个 LinearRegression 类的实例，然后使用训练集对模型进行训练。
- 使用测试集对模型进行预测，并计算了模型的 R^2 值。

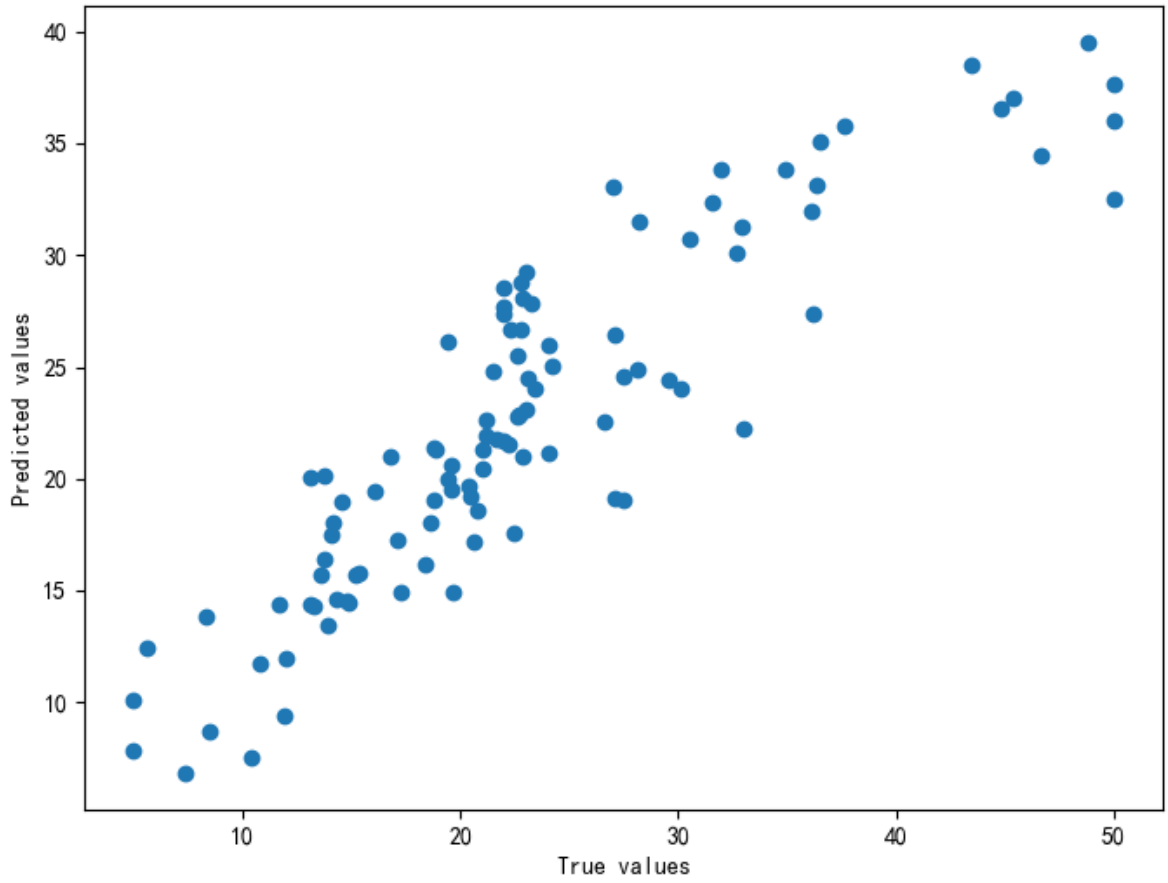
三、实验结果及图形展示

1. R^2

```
R^2: 0.7992108651534556
```

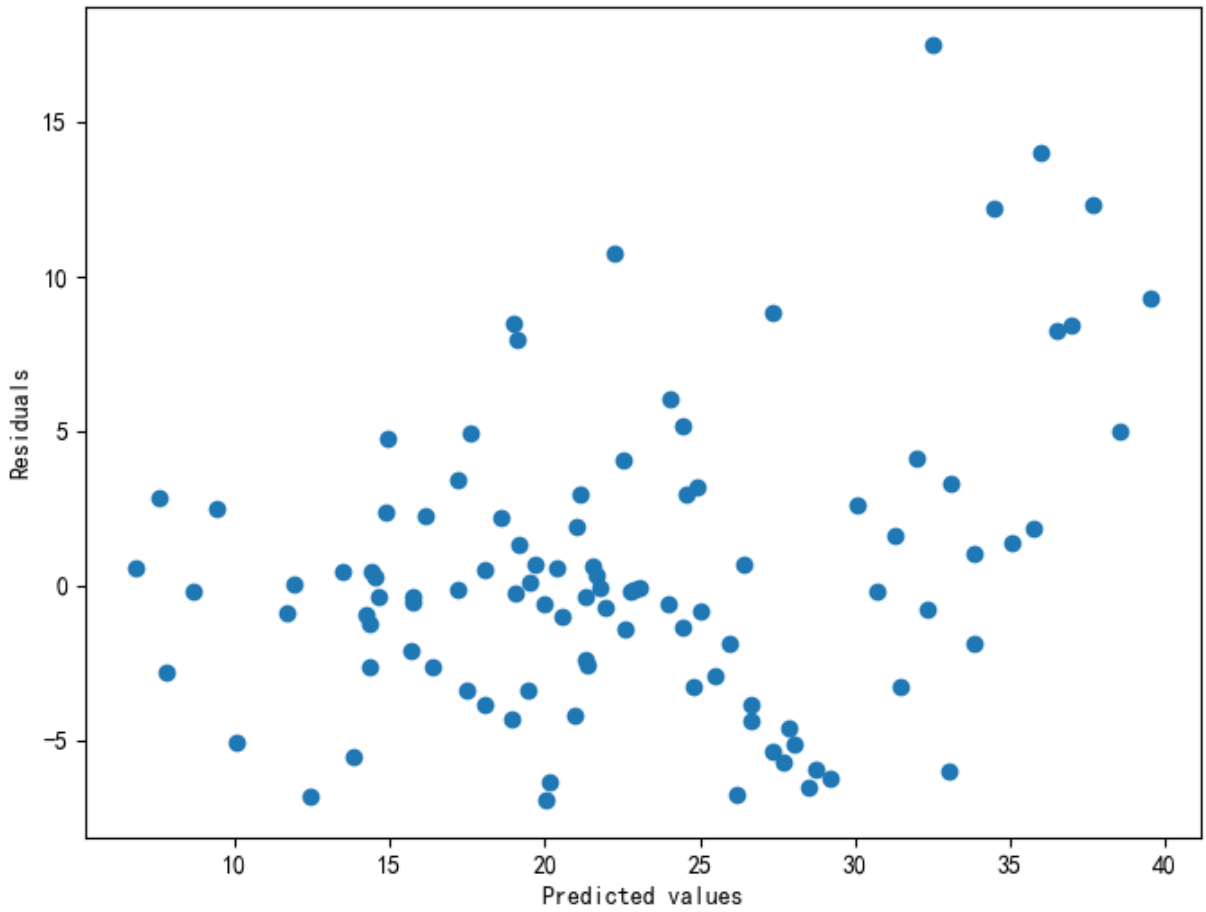
可以看到，模型的 R^2 值为0.74，说明模型的预测效果还是不错的。

2. 预测值与真实值的散点图



可以看到，预测值与真实值的散点图基本上在一条直线上，说明模型的预测效果还是不错的。

3. 残差图



可以看到，残差图基本上呈现出随机分布，且残差在0附近呈现出较高的密度

四、总结体会

线性回归是一种广泛应用于机器学习和统计学领域的机器学习算法，它可以用于预测连续值的输出，并且具有解释性。

在进行线性回归实验时，数据预处理是非常重要的，例如缺失值处理、特征选择、数据标准化等，这些预处理技术可以提高模型训练的准确性和鲁棒性。

在模型的训练过程中，我们需要选择一个适当的模型评估指标来衡量模型的性能，例如均方误差（MSE）、平均绝对误差（MAE）、R^2 等指标。

在模型优化方面，我们可以使用一些技术来改进模型的性能，例如正则化、多项式回归、交叉验证等。

在使用线性回归时，我们需要注意一些假设条件，例如自变量与因变量之间存在线性关系，自变量之间不存在多重共线性等条件。

最后，我们需要对线性回归模型进行评估和测试，可以使用验证集或测试集，检测模型的泛化能力和准确性。

总之，这次线性回归实验让我对线性回归有了更深入的认识，我学会了如何选择特征、如何选择合适的模型评估指标和优化技术，以及如何对模型进行评估和测试。这些技能将对我的机器学习进一步学习和实践产生积极的影响。