北京交通大學

信息网络综合专题研究课大作业

班级：通信 1706

姓名：曾俊三

学号：17211108

一．线性回归代码：

代码来源：GitHub yunjey

```
import torch
import torch.nn as nn
import numpy as np
import matplotlib.pyplot as plt


# Hyper-parameters
input_size = 1
output_size = 1
num_epochs = 60
learning_rate = 0.001

# Toy dataset
x_train = np.array([[3.3], [4.4], [5.5], [6.71], [6.93], [4.168],
                    [9.779], [6.182], [7.59], [2.167], [7.042],
                    [10.791], [5.313], [7.997], [3.1]], dtype=np.float32)

y_train = np.array([[1.7], [2.76], [2.09], [3.19], [1.694], [1.573],
                    [3.366], [2.596], [2.53], [1.221], [2.827],
                    [3.465], [1.65], [2.904], [1.3]], dtype=np.float32)

# Linear regression model
model = nn.Linear(input_size, output_size)

# Loss and optimizer
criterion = nn.MSELoss()
optimizer = torch.optim.SGD(model.parameters(), lr=learning_rate)

# Train the model
for epoch in range(num_epochs):
```

```
    # Convert numpy arrays to torch tensors
    inputs = torch.from_numpy(x_train)
    targets = torch.from_numpy(y_train)

    # Forward pass
    outputs = model(inputs)
    loss = criterion(outputs, targets)

    # Backward and optimize
    optimizer.zero_grad()
    loss.backward()
    optimizer.step()

    if (epoch+1) % 5 == 0:
        print ('Epoch [{}/{}], Loss: {:.4f}'.format(epoch+1, num_epochs,
loss.item()))

# Plot the graph
predicted = model(torch.from_numpy(x_train)).detach().numpy()
plt.plot(x_train, y_train, 'ro', label='Original data')
plt.plot(x_train, predicted, label='Fitted line')
plt.legend()
plt.show()

# Save the model checkpoint
torch.save(model.state_dict(), 'model.ckpt')
```

二.运行结果

结果 1：

Epoch [5/60], Loss: 0.7007

Epoch [10/60], Loss: 0.3863

Epoch [15/60], Loss: 0.2589

Epoch [20/60], Loss: 0.2073

Epoch [25/60], Loss: 0.1864

Epoch [30/60], Loss: 0.1779

Epoch [35/60], Loss: 0.1745

Epoch [40/60], Loss: 0.1731

Epoch [45/60], Loss: 0.1725

Epoch [50/60], Loss: 0.1723

Epoch [55/60], Loss: 0.1722

Epoch [60/60], Loss: 0.1721

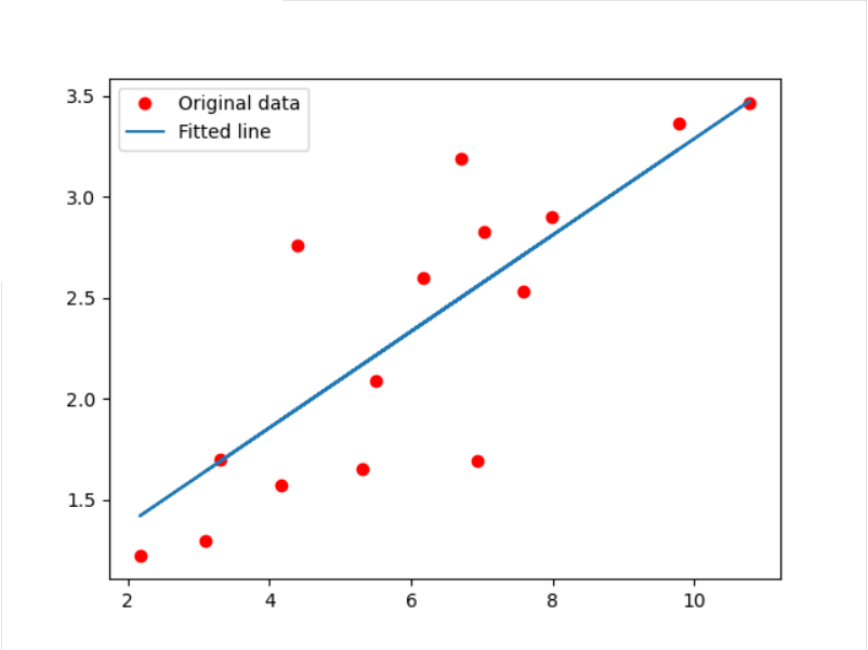

图 1

结果 2：

Epoch [5/60], Loss: 5.0693

Epoch [10/60], Loss: 2.3013

Epoch [15/60], Loss: 1.1796

Epoch [20/60], Loss: 0.7248

Epoch [25/60], Loss: 0.5401

Epoch [30/60], Loss: 0.4650

Epoch [35/60], Loss: 0.4341

Epoch [40/60], Loss: 0.4213

Epoch [45/60], Loss: 0.4157

Epoch [50/60], Loss: 0.4131

Epoch [55/60], Loss: 0.4116

Epoch [60/60], Loss: 0.4107

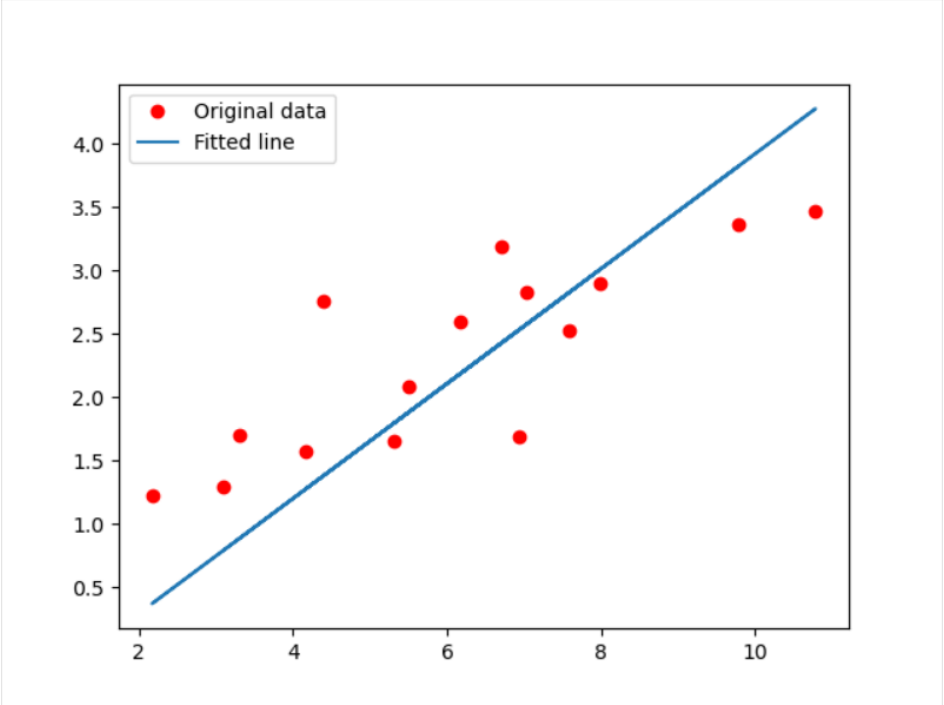

图 2

三．代码分析

代码思路:

1. 设置参数;
2. 设置数据库数据，即线性回归点数据;
3. 构建线性回归模型;
4. 建立优化器;
5. 构建训练模型
6. 调用库函数，绘制图表。

该代码思路明确清晰，算法过程也比较简单易懂，算法中调用了 matplotlib 函数进行绘图。输出的结果上来看，两此结果有较大差异，结果符合预期，运行正确。

四．心得体会

这次大作业，我本来计划完成第一个大作业，但是由于遇到诸多困难，话费很多时间未得到成果，请教同学无果，考虑到截止时间压力和能力问题，以及考研的压力，中途放弃了第一个作业，选择了老师上课讲的较为简单的方式，在 GitHub 网站上下载了线性回归代码，学习了很久如何调试运行 python 代码，并学习了 python 基础语言，以及库函数的安装和调用方式，学到了很多使用的技巧，收获颇丰，以后如果还有时间和机会，希望可以再次尝试学习该部分内容，作更深入的了解。