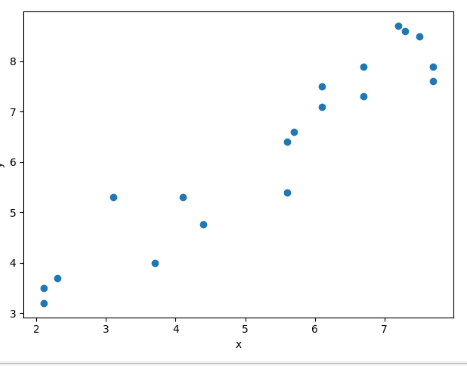
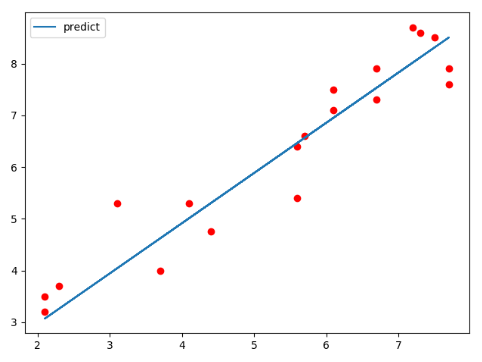
线性回归（简单的一元线性REGRESSION分析）

1. 自生成数据，二维



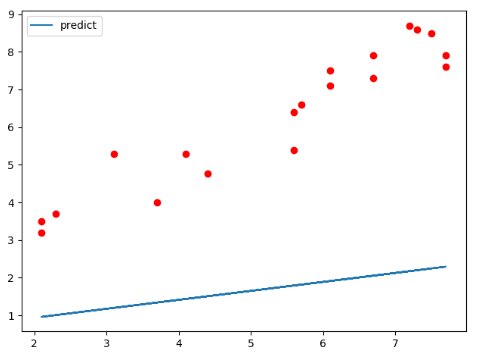
1. 优化函数选取梯度下降算法，学习率定义为0.001，训练轮数1000



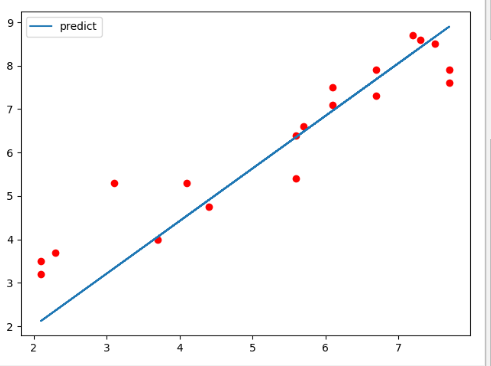
1. 固定一千轮迭代，调整学习率

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 0.00001 | 0.0001 | 0.005 | 0.001 | 0.01 | 0.1 |
| 0.12 | 0.13 | 0.15 | 1 | 1.001 | 1.002 |
|  |  |  |  |  |  |

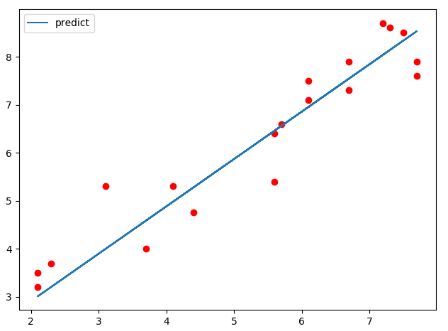
顺次结果：



0.00001

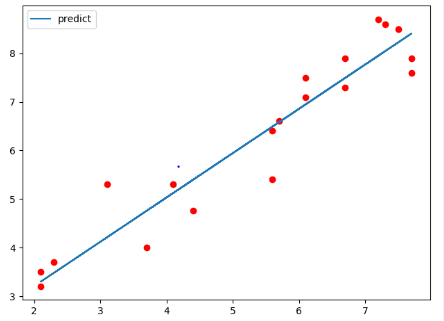


0.0001



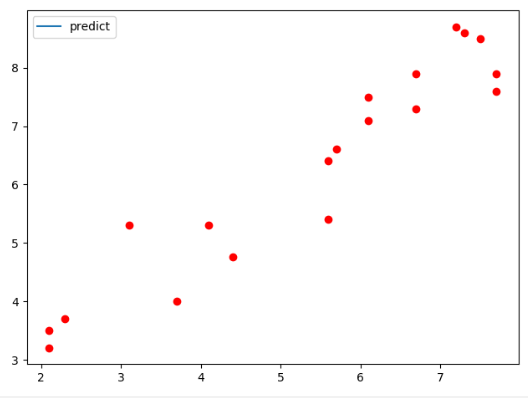
0.005

Time: 1.1049277782440186 s



0.01

Time: 1s

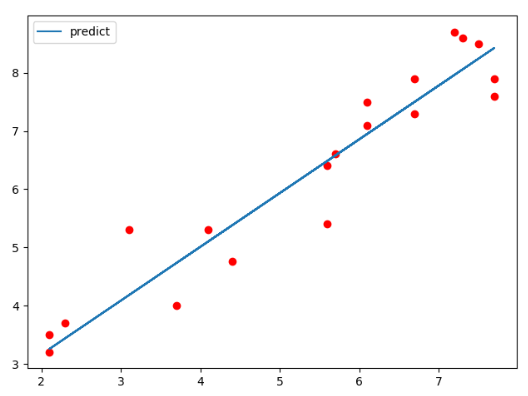


0.1

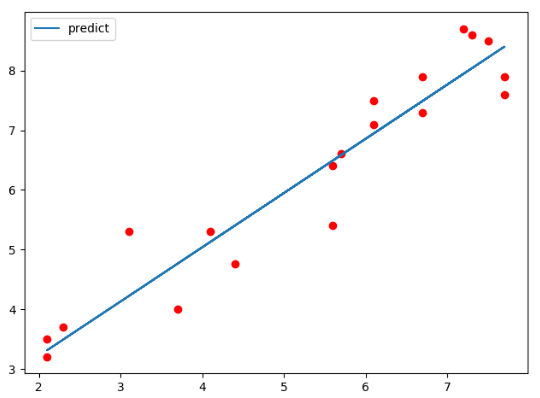
无法拟合

类似，超过0.1，无法收敛并拟合出结果

1. 提高迭代次数，选取LR=0.001(1e-3)

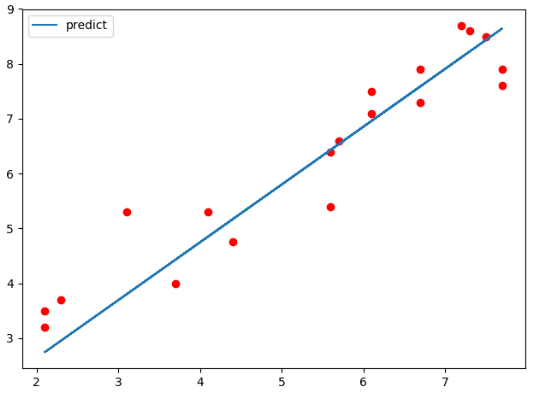


5000轮



8000轮

1. 减少迭代次数，固定LR=0.001



500轮

1. 思考【思考结果本人总结完会即时上传到本文件所在文件夹】

LEARNING-RATE 如何影响拟合过程，其作用何在？

SGD与MSE数学原理？

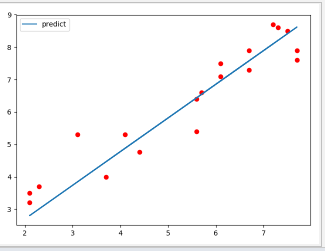
loss.backward()是如何计算梯度的？

optimizer.zero\_grad()为何在每次方向传播的时候需要将参数的梯度归零？

实际问题如何界定最优，对现实帮助最大的拟合？是否能够很好拟合以往数据（大多数点落到拟合曲线/直线上）就是最好的？

如果不是这样的，使用MSE损失，拟合出来的结果对未来数据能否做出很好的预测？它的上限是什么？怎么界定准确范围？

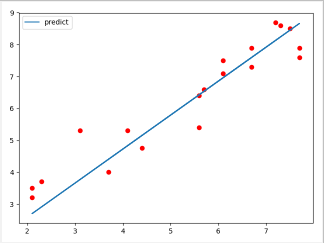
1. 提供几组参考数据



Epoch [2000/2000], Loss: 0.3936

LR=0.001

epochs=2000



Epoch [1000/1000], Loss: 0.4277

LR=0.001

epochs=1000

1. 修正

criterion = nn.MSELoss()

loss = criterion(outputs, targets)

PyTorch1.1版本 使用 loss.item()

替换 loss.data[0]