线性回归作业

1. 说明线性回归的特点、过程

简单、适用于数据呈现线性分布

1. 先给一个初始的线性模型y=kx+b
2. 构建一个损失函数
3. 利用梯度下降的方法来优化线性模型的参数k,b
4. 利用优化好的模型来执行预测

答：特点是，假设因变量与自变量之间存在线性关系，并通过最小化损失函数来拟合出一条最合适的直线，它可以用来预测和诠释变量之间的关系。

过程：①收集数据->②定义模型->③模型训练->④模型预测

1. 列举生活中常见的线性回归应用场景

答：经济领域：预测物品的价格等。

房地产：预测房价与房屋面积、地理位置等因素之间的关系。

人文社科：预测睡眠和作息、身体状况、遗传因素之间的关系。

医学研究：分析药物剂量与治疗效果的关系。

1. 什么是梯度下降？梯度下降的作用是什么？为什么要使用梯度下降？

梯度下降：沿着损失函数梯度的反方向，调整参数的方式叫做梯度下降。

通过寻找损失函数最小值的方式来确定最优的参数。

答：梯度下降是一种优化算法，用于求解最小化目标函数的参数。它通过迭代的方式逐步调整参数，使目标函数的值逐渐趋于最小值。

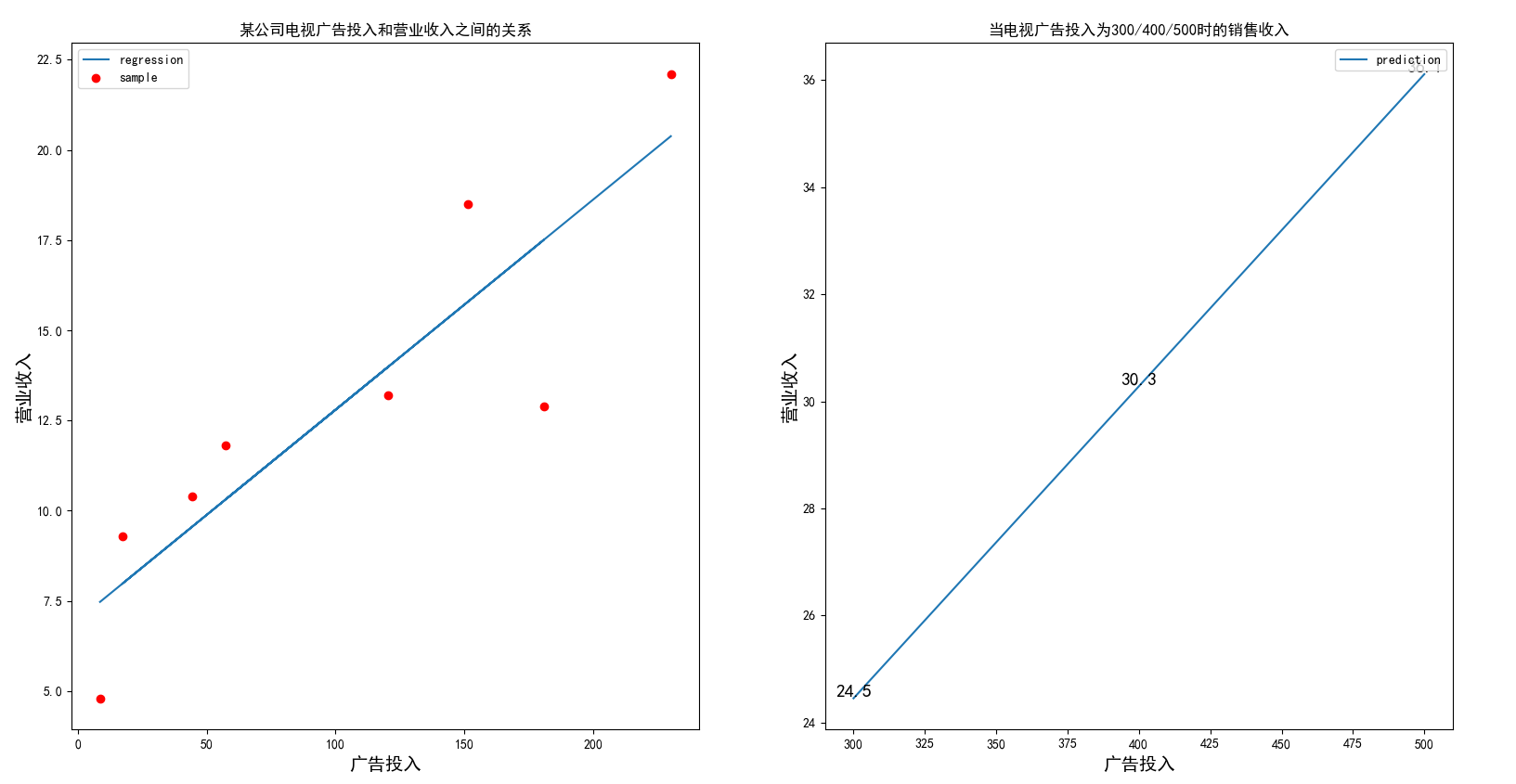
梯度下降是一种高效的优化算法，能够在大规模数据集上进行计算；拥有广泛适用性，可以应用于各种机器学习算法中，如线性回归、逻辑回归和神经网络等；梯度下降是一种高效的优化算法，能够在大规模数据集上进行计算。

（4）

源代码：

1. import numpy as np
2. import matplotlib.pyplot as plt
3. import sklearn.linear\_model as lm
4. plt.rcParams["font.sans-serif"]="SimHei"  *#显示中文*
5. train\_x=np.array([[230.1],[44.5],[17.2],[151.5],[180.8],[57.5],[120.2],[8.6]])
6. train\_y=np.array([22.1,10.4,9.3,18.5,12.9,11.8,13.2,4.8])
7. *#定义模型*
8. model=lm.LinearRegression()
9. *#模型训练*
10. model.fit(train\_x,train\_y)
11. *#模型预测*
12. pred\_y=model.predict(train\_x)
13. print("预测的：",pred\_y)
14. pred\_x\_given=np.array([[300],[400],[500]])
15. pred\_y\_given=model.predict(pred\_x\_given)
16. print("系数：",model.coef\_)
17. print("截距：",model.intercept\_)
18. k=model.coef\_[-1]
19. b=model.intercept\_
20. plt.subplot(1,2,1)
21. plt.title("某公司电视广告投入和营业收入之间的关系")
22. plt.xlabel("广告投入",fontsize=14)
23. plt.ylabel("营业收入",fontsize=14)
24. plt.scatter(train\_x,train\_y,c="r",label="sample")
25. plt.plot(train\_x,pred\_y,label="regression")
26. plt.legend()
27. plt.subplot(1,2,2)
28. plt.title("当电视广告投入为300/400/500时的销售收入")
29. plt.xlabel("广告投入",fontsize=14)
30. plt.ylabel("营业收入",fontsize=14)
31. plt.plot(pred\_x\_given,pred\_y\_given,label="prediction")
32. for a,b in zip(pred\_x\_given,pred\_y\_given) :
33. plt.text(x=a,y=b,s=round(b,1),fontsize=14,ha="center",va="bottom")
34. plt.legend()
35. plt.show()

运行结果:



(5)

源代码：

1. import numpy as np
2. import random
3. import matplotlib.pyplot as plt
4. import sklearn.linear\_model as lm
5. plt.rcParams["font.sans-serif"]="SimHei"  #显示中文
6. train\_x=np.random.randint(1,10,(10,1))
7. #print(train\_x)
8. calculate\_y=2\*train\_x
9. #print(calculate\_y)
10. noise=np.random.randint(1,3,(10,1))
11. train\_y=calculate\_y+noise
12. print(train\_y)
13. model=lm.LinearRegression()
14. model.fit(train\_x,train\_y)
15. pred\_y=model.predict(train\_x)
16. print(pred\_y)
17. plt.figure()
18. plt.title("线性回归")
19. plt.xlabel("X",fontsize=14)
20. plt.ylabel("Y",fontsize=14)
21. plt.scatter(train\_x,train\_y,c="r",label="sample")
22. plt.plot(train\_x,pred\_y,label="regression")
23. plt.show()

运行结果：

