# 第1章作业

## 作业题目内容

使用以下商品房销售记录表数据，用梯度下降法，编程实现一个房价预测系统。



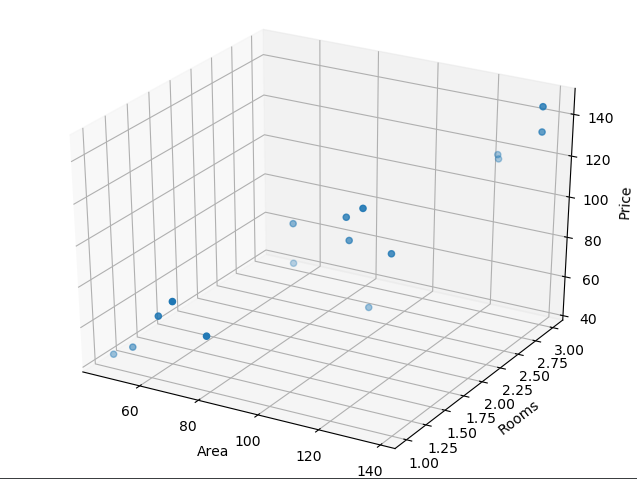
## 求解原理

**数据：**

**=**

**维度:（16，3）**

**数据可视化：**



[图1.商品房面积、房间数和价格的关系]

**目的：**

根据已有的数据，对数据进行拟合，实现对房价的预测

**求解方法：**

对于二元线性回归问题，可以设置假设函数，并通过损失函数进行参数的优化，达到对样本数据的拟合

1、假设函数设置为：

()

2、损失函数设置为：

(n为样本总量，)

3、采用梯度下降法最小化损失函数，对参数

梯度下降法各参数的更新表达式：

即：

多次迭代更新参数后的函数

## 编程求解

### 编程实验平台说明

开发平台：

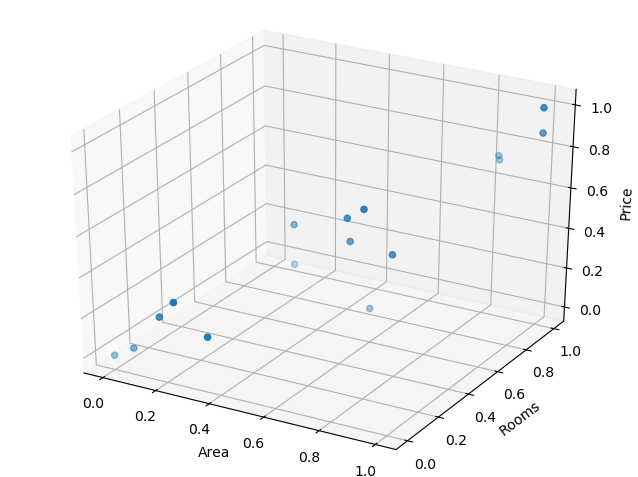
Linux version 5.4.0-40-generic (buildd@lcy01-amd64-011) (gcc version 9.3.0 (Ubuntu 9.3.0-10ubuntu2))

Python版本：3.7.6

开发工具：Pycharm+Xshell+Xming

### 实验方案

1. 对数据进行归一化处理

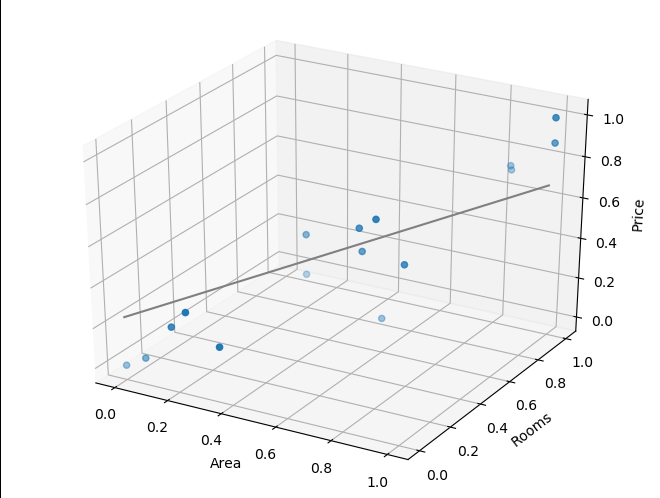


[图2.归一化数据后的函数图像（分布不改变）]

1. 定义期望函数（Loss Function）求Loss值
2. 定义梯度下降函数，对参数进行优化（求最小值）

### 程序及运行结果说明

import numpy as np  
import tensorflow as tf  
from matplotlib import pyplot as plt  
from mpl\_toolkits.mplot3d import Axes3D  
from sklearn.preprocessing import MinMaxScaler  
  
#归一化处理数据  
mm = MinMaxScaler()  
x\_1 = mm.fit\_transform(np.array([137.97,104.50,100.00,124.32,79.20,99.00,124.00,114.00,106.69,138.05,53.75,46.91,68.00,63.02,81.26,81.21]).reshape(16,-1))  
x\_2 = mm.fit\_transform(np.array([3.00,2.00,2.00,3.00,1.00,2.00,3.00,2.00,2.00,3.00,1.00,1.00,1.00,1.00,2.00,2.00]).reshape(16,-1))  
y = mm.fit\_transform(np.array([145.00,110.00,93.00,116.00,65.32,104.00,118.00,91.00,62.00,133.00,51.00,45.00,78.50,69.65,75.69,95.30]).reshape(16,-1))  
  
#未归一化的数据  
# x\_1 = np.array([137.97,104.50,100.00,124.32,79.20,99.00,124.00,114.00,106.69,138.05,53.75,46.91,68.00,63.02,81.26,81.21])  
# x\_2 = np.array([3.00,2.00,2.00,3.00,1.00,2.00,3.00,2.00,2.00,3.00,1.00,1.00,1.00,1.00,2.00,2.00])  
# y = np.array([145.00,110.00,93.00,116.00,65.32,104.00,118.00,91.00,62.00,133.00,51.00,45.00,78.50,69.65,75.69,95.30])  
  
#对数据进行可视化  
fig = plt.figure()  
ax = Axes3D(fig)  
ax.set\_xlabel("Area")  
ax.set\_ylabel("Rooms")  
ax.set\_zlabel("Price")  
# plt.show()  
  
  
#Loss calculate  
data = np.stack((x\_1,x\_2,y),axis=1) #转为3D数组  
# print("data=",data)  
# print(data.shape)  
# print(data.size)  
# print(len(data))  
  
#print function  
def Print\_Feature(data):  
 for i in range(0,len(data)):  
 x\_1 = data[i,0]  
 x\_2 = data[i,1]  
 y = data[i,2]  
 print(x\_1,x\_2,y)  
  
#Loss Function  
def mse(b,w1,w2,data):  
 totalError = 0  
 for i in range(0,len(data)):  
 x\_1 = data[i,0]  
 x\_2 = data[i,1]  
 y = data[i,2]  
 totalError +=((w1\*x\_1+w2\*x\_2+b)-y)\*\*2  
 return totalError/float((len(data))\*2)  
  
#定义一步梯度下降运算  
def step\_gradient(b\_current,w1\_current,w2\_current,data,lr):  
 b\_gradient = 0  
 w1\_gradient = 0  
 w2\_gradient = 0  
 M = 16  
  
 for i in range(0,len(data)):  
 x\_1 = data[i,0]  
 x\_2 = data[i,1]  
 y = data[i,2]  
  
 w1\_gradient += (2/M)\*x\_1\*((w1\_current\*x\_1+w2\_current\*x\_2+b\_current)-y)  
 w2\_gradient += (2/M)\*x\_2\*((w1\_current\*x\_1+w2\_current\*x\_2+b\_current)-y)  
 b\_gradient += (2/M)\*((w1\_current\*x\_1+w2\_current\*x\_2+b\_current)-y)  
  
 new\_b = b\_current-(lr\*b\_gradient)  
 new\_w1 = w1\_current-(lr\*w1\_gradient)  
 new\_w2 = w2\_current-(lr\*w2\_gradient)  
  
 return [new\_b,new\_w1,new\_w2]  
  
#迭代num\_iterations次梯度下降运算，优化参数w1，w2，b  
def gradient\_descent(data,star\_b,star\_w1,star\_w2,lr,num\_iterations):  
 b = star\_b  
 w1 = star\_w1  
 w2 = star\_w2  
 for step in range(num\_iterations):  
 b,w1,w2 = step\_gradient(b,w1,w2,data,lr)  
 loss = mse(b,w1,w2,data)  
 if step%5 == 0:  
 print(f"iterations:{step},loss:{loss},w1:{w1},w2:{w2},b:{b}")  
 return [b,w1,w2]  
  
  
def main():  
 lr = 0.0001  
 init\_b = 0  
 init\_w1 = 0  
 init\_w2 = 0  
 num\_iterations = 10000  
 Print\_Feature(data)  
 [b,w1,w2] = gradient\_descent(data,init\_b,init\_w1,init\_w2,lr,num\_iterations)  
 loss = mse(b,w1,w2,data)  
 print(f"Final Loss:{loss},w1:{w1},w2:{w2},b:{b}")  
 xx = np.arange(0,1,0.01)  
 yy = np.arange(0,1,0.01)  
 Z = w1\*xx+w2\*yy+b  
 ax.scatter(x\_1, x\_2, y) # 绘制散点图  
 ax.plot3D(xx,yy,Z,'gray') #绘制最后更新参数后的预测房价函数  
 plt.show()  
 # plt.savefig('result.png',bbox\_inches='tight')  
if \_\_name\_\_ == '\_\_main\_\_':  
 main()



[图3.拟合后的函数图像，灰色直线为预测房价的函数模型]

## 讨论及收获

1. 深刻理解线性回归问题
2. 深刻理解梯度下降，并推到梯度下降算法
3. Python函数库的运用，matplotlib