**课后作业一：多项式回归模型**

任务：多项式回归是一种回归分析形式，在这种形式中，自变量 ( x ) 和因变量 ( y ) 之间的关系被建模为 ( n ) 阶多项式。使用机器学习的方法来创建一个多项式回归模型，该模型可以根据给定的数据集预测结果。数据集由自变量 ( x ) 和因变量 ( y ) 组成，你的任务是找到一个多项式，能最好地描述 ( x ) 和 ( y ) 之间的关系。

数据集：本实验选取数据集包含125个样本点，每个样本具有一个自变量( x )和一个因变量( y )。数据集根据4:1的比例划分为训练集和测试集。

**实现：**

1.导入必要的库

import pandas as pd #用来分析结构化数据

import numpy as np #提供高性能的矩阵运算

import matplotlib.pyplot as plt #可视化工具

import csv #读写文件的库

2.函数定义

（1）函数 f(x, theta)：用于计算输入 x 和参数 theta 的多项式回归函数值。其中x 是一个矩阵，每一行对应一个输入样本；theta 是一个参数向量。

def f(x,theta):

return np.dot(x,theta)

1. 函数 standardize(x)：用于对输入数据 x 进行标准化，通过减去均值并除以标准差。

def standardize(x):

    return (x - np.mean(x)) / np.std(x)

1. 函数 to\_matrix(x)：用于将输入向量 x 转换为一个矩阵，包含三列：一列全是1（用于截距），x 和 x^2。

def to\_matrix(x):

    return np.vstack([np.ones(x.shape[0]),x,x\*\*2]).T

1. 函数 E(x, y, theta)：用于计算预测值与实际值 y 之间的误差（损失）。使用均方误差（MSE）作为误差指标。

def E(x,y,theta):

    return 0.5 \* np.sum((f(x,theta)) - data\_y)\*\*2

1. 函数 regression(x, y, theta)：使用梯度下降法拟合多项式模型到数据上。迭代更新 theta 以最小化误差。

def regression(x,y,theta):

    count=0

    diff=1

    new\_x= to\_matrix(standardize(x))

    error = E(new\_x,y,theta)

    ETA = 1e-3

    while diff > 1e-4:

        theta = theta - ETA\*np.dot(f(new\_x,theta)-y,new\_x)

        count+=1

        new\_error = E(new\_x,y,theta)

        diff = abs(new\_error-error)

        error = new\_error

        print('第{}次,theat={},差值={:.4f}'.format(count,theta,diff))

    return theta

1. 函数 getdata(path)：从train\_dataset.csv文件路径加载训练数据。返回对应于特征和标签的数组 data\_x 和 data\_y。（为了区分train和test，我写了两个数据读取函数）

def getdata(path):

    data\_frame = pd.read\_csv(r'D:\dataenclorse\first\train\_dataset.csv')  # skiprows=14

    data\_x,data\_y = np.array(data\_frame['x']), np.array(data\_frame['y'])

    return data\_x,data\_y

1. 函数 getdata2(path)：从test\_dataset.csv文件路径加载测试数据。返回测试特征数组 test\_data\_x 和 y。

def getdata2(path):

    data\_frame = pd.read\_csv(r'D:\dataenclorse\first\test\_dataset.csv')  # skiprows=14

    test\_data\_x,y= np.array([data\_frame['x'], np.array(data\_frame['y'])])

    return test\_data\_x,y

3.具体实现

（1）初始化参数 theta，随机生成三个参数。

if \_\_name\_\_ == '\_\_main\_\_':

    theta = np.random.randn(3)

1. 加载训练数据，使用梯度下降法拟合模型。

 data\_x,data\_y=getdata('train\_dataset.csv')

    new\_theta = regression(data\_x,data\_y,theta)

1. 标准化训练数据并绘制散点图，生成预测曲线并绘制。

  x = standardize(data\_x)

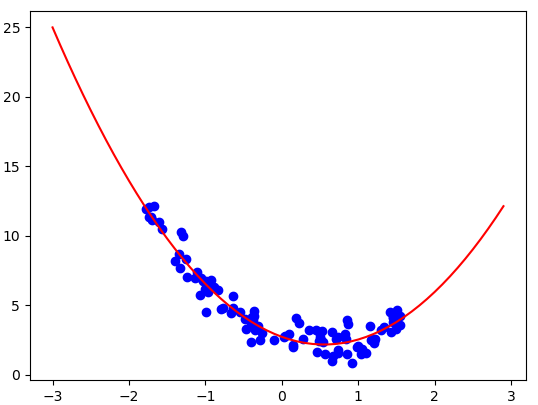
    plt.scatter(x,data\_y,c='blue')

    xx = np.arange(-3,3,0.1)

    plt.plot(xx,f(to\_matrix(xx),new\_theta),c='red')

    plt.show()

散点图如下：近似拟合为二次函数：



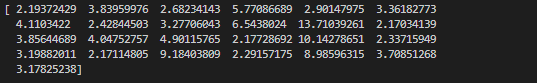
1. 加载测试数据，使用拟合的模型进行预测。

    test\_data\_x,y=getdata2('test\_dataset.csv')

    test\_data\_y=f(to\_matrix(standardize(test\_data\_x)),new\_theta)

    print(test\_data\_y)

展示结果：



4.将预测结果保存到指定路径的CSV文件中

    file\_path=r'D:\dataenclorse\first\t.csv'

    with open(file\_path,'w',newline='',encoding='utf-8') as f:

        fieldnames=['x','y']

        f\_csv = csv.DictWriter(f, fieldnames=fieldnames)

        f\_csv.writeheader()

        for i in range(0,len(test\_data\_y)):

            f\_csv.writerow({'x':test\_data\_x[i],'y':test\_data\_y[i]})

    pass

