**《机器学习与数据挖掘》实验五**

实验题目： 编程实现误差逆传播算法（BP算法）

实验目的： 掌握误差逆传播算法（BP算法）的工作流程

实验环境（硬件和软件） Anaconda/Jupyter notebook/Pycharm

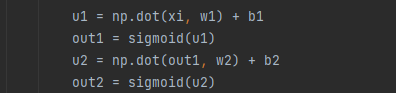
实验内容：

编码实现标准BP算法，在西瓜数据集3.0上用这个算法训练一个单隐层网络，并进行测试。

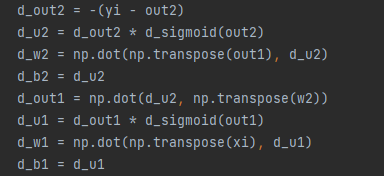
**要求：**

**一、**已经给定部分代码，补充完整的代码，需要补充代码的地方已经用红色字体标注，在第（2）部分，包括：

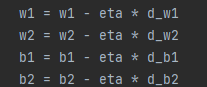
**#补充前向传播代码**



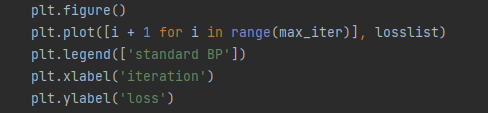
**#补充反向传播代码**



**#补充参数更新代码**



**#补充Loss可视化代码**



**二、**将补充完整的第（2）部分的代码提交，并提交实验结果；（**也可以自己重写这部分的代码提交**）

import pandas as pd

import numpy as np

from sklearn.preprocessing import LabelEncoder

from sklearn.preprocessing import StandardScaler

import matplotlib.pyplot as plt

seed = 2020

import random

np.random.seed(seed) # Numpy module.

random.seed(seed) # Python random module.

plt.rcParams['font.sans-serif'] = ['SimHei'] #用来正常显示中文标签

plt.rcParams['axes.unicode\_minus'] = False #用来正常显示负号

plt.close('all')

**（1）数据预处理**

def preprocess(data):

#将非数映射数字

for title in data.columns:

if data[title].dtype=='object':

encoder = LabelEncoder()

data[title] = encoder.fit\_transform(data[title])

#去均值和方差归一化

ss = StandardScaler()

X = data.drop('好瓜',axis=1)

Y = data['好瓜']

X = ss.fit\_transform(X)

x,y = np.array(X),np.array(Y).reshape(Y.shape[0],1)

return x,y

#定义Sigmoid

def sigmoid(x):

return 1/(1+np.exp(-x))

#求导

def d\_sigmoid(x):

return x\*(1-x)

**（2）标准BP算法**

def standard\_BP(x,y,dim=10,eta=0.8,max\_iter=500):

n\_samples = 1

w1 = np.random.random((x.shape[1],dim))

w2 = np.random.random((dim,1))

b1 = np.random.random((n\_samples,dim))

b2 = np.random.random((n\_samples,1))

losslist = []

for ite in range(max\_iter):

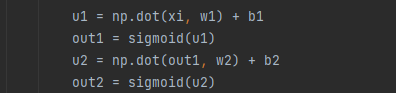
loss\_per\_ite = []

for m in range(x.shape[0]):

xi,yi = x[m,:],y[m,:]

xi,yi = xi.reshape(1,xi.shape[0]),yi.reshape(1,yi.shape[0])

#**#补充前向传播代码**



loss = np.square(yi - out2)/2

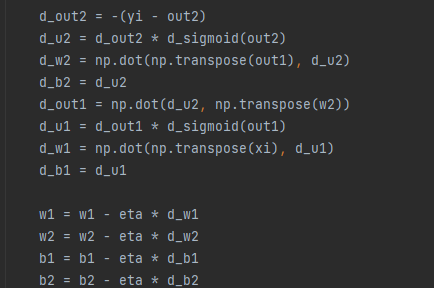
loss\_per\_ite.append(loss)

print('iter:%d loss:%.4f'%(ite,loss))

##反向传播

#**#补充反向传播代码**

#**#补充参数更新代码**

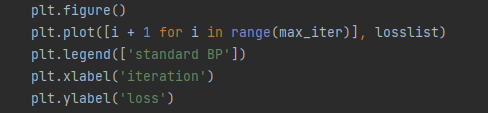


losslist.append(np.mean(loss\_per\_ite))

##Loss可视化

plt.figure()

**##补充Loss可视化代码**



plt.show()

return w1,w2,b1,b2

**（3）测试**

data = pd.read\_table('watermelon30.txt',delimiter=',')

data.drop('编号',axis=1,inplace=True)

x,y = preprocess(data)

dim = 10

w1,w2,b1,b2 = standard\_BP(x,y,dim)

#根据当前的x，预测其类别；

u1 = np.dot(x,w1)+b1

out1 = sigmoid(u1)

u2 = np.dot(out1,w2)+b2

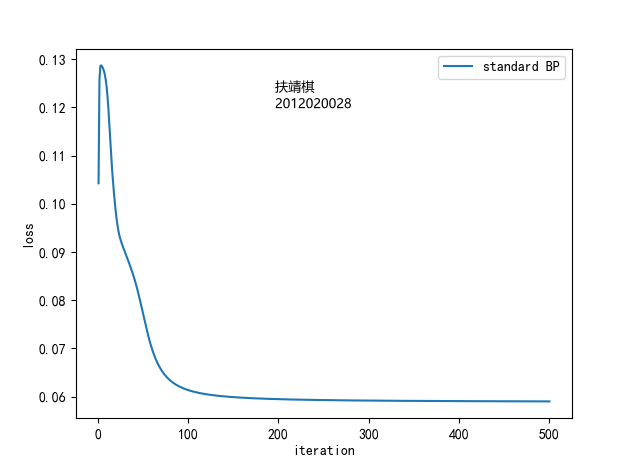
out2 = sigmoid(u2)

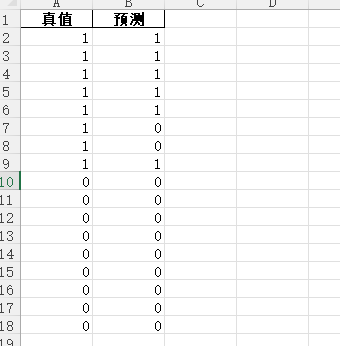
y\_pred = np.round(out2)

result = pd.DataFrame(np.hstack((y,y\_pred)),columns=['真值','预测'] )

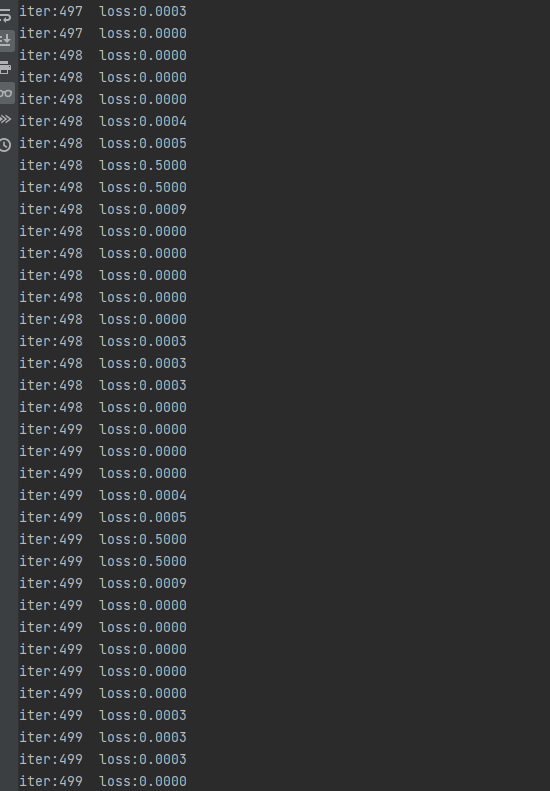
result.to\_excel('result.xlsx',index=False)

实验截图：









实验心得：

通过本次实验我学习到了如何运用BP算法来解决实际问题。BP算法的[非线性](https://so.csdn.net/so/search?q=%E9%9D%9E%E7%BA%BF%E6%80%A7&spm=1001.2101.3001.7020)的映射能力。该能力主要表现在复杂的权值上，而权值的变化就是BP的学习过程，因此可以存储大量的输入输出模式映射关系，而无需给出给出具体的映射方程，在解决实际问题有很好的作用。对BP算法的理解更加深刻，编写代码能力增强了，动手能力也得到提升。