**《机器学习与数据挖掘》实验四**

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算法和累积BP算法，在西瓜数据集3.0上分别用这两个算法训练一个单隐层网络，并进行比较。

**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，包括：

1. 在第（2）部分；

**#补充反向传播代码**

**#补充Loss可视化代码**

2. 在第（3）部分；

**#补充标准BP算法代码**

**#补充Loss可视化代码**

3. 在第（4）部分：

**#补充测试代码，根据当前的x，预测其类别；**

**二、**将补充完整的第（2）（3）（4）部分的代码提交，并提交实验结果；（**也可以自己重写这三部分的代码提交**）

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）累积BP算法**

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

n\_samples = x.shape[0]

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

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

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

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

losslist = []

for ite in range(max\_iter):

##前向传播

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

out1 = sigmoid(u1)

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

out2 = sigmoid(u2)

loss = np.mean(np.square(y - out2))/2

losslist.append(loss)

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

**#补充反向传播代码**

##更新

w1 = w1 - eta\*d\_w1

w2 = w2 - eta\*d\_w2

b1 = b1 - eta\*d\_b1

b2 = b2 - eta\*d\_b2

plt.figure()

**##补充Loss可视化代码**

plt.show()

return w1,w2,b1,b2

**（3）标准BP算法**

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

n\_samples = 1

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

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

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

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

losslist = []

**#补充标准BP算法代码**

**#补充Loss可视化代码**

return w1,w2,b1,b2

**（4）测试**

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

data.drop('编号',axis=1,inplace=True)

x,y = preprocess(data)

dim = 10

# \_,\_,\_,\_ = standard\_BP(x,y,dim)

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

**#补充测试代码，根据当前的x，预测其类别；**