《模式识别与机器学习A》实验报告

实验题目： 逻辑回归

班级： 2203601

学号： 2022113416

姓名： 刘子康

**实验报告内容**

**一、实验目的**

·理解逻辑回归模型的原理，掌握逻辑回归模型的参数估计算法；

·利用逻辑回归模型实现二分类算法，并生成数据集或利用实际数据加以验证。

**二、实验内容**

实现两种损失函数的参数估计（1.无惩罚项；2.加入对参数的惩罚），可以采用梯度下降、共轭梯度或者牛顿法等。

·手动生成两个类别数据（可以用高斯分布），验证逻辑回归模型算法。考察特征分布在满足和不满足朴素贝叶斯假设时，分别会得到什么样的结果。

·在UCI网站上找一些实际数据加以测试，验证逻辑回归模型在不同领域的广泛用处，例如广告预测。

**三、实验环境**

·操作系统：Windows 11

·编程语言：Python 3.10

·第三方库：Numpy 1.23.4，Matplotlib 3.8.2

·IDE：Pycharm 2022社区版

**四、实验过程、结果及分析**

**4.1实验原理**

逻辑回归（Logistic Regression）是一种常用于二分类问题的机器学习算法。它基于线性回归模型的思想，但通过引入一个Sigmoid函数（也称为逻辑函数），将输出限制在0到1之间，从而进行类别概率预测，达到分类效果。

逻辑回归的核心思想是建立一个线性模型来进行分类。对于一个输入特征向量 ，线性回归的预测结果可以表示为：，其中，是偏置项b，是权重（系数向量）w，这个公式表示了特征和目标之间的线性关系。在线性回归中，预测值y可以是任意实数，而在分类问题中，需要将预测值转换为一个概率值，表示样本属于某个类别的概率，这个转换是逻辑回归的关键步骤，通过一个非线性激活函数Sigmoid实现。

Sigmoid函数的公式为 ，其中，即线性回归的输出。Sigmoid 函数的值域在(0, 1)之间，当z趋近于正无穷时输出趋近于1，当z趋近于负无穷时输出趋近于0，适合用于表示概率。此时逻辑回归预测某样本属于类别1的后验概率为。

损失函数选择交叉熵函数，并使用梯度下降法进行优化。

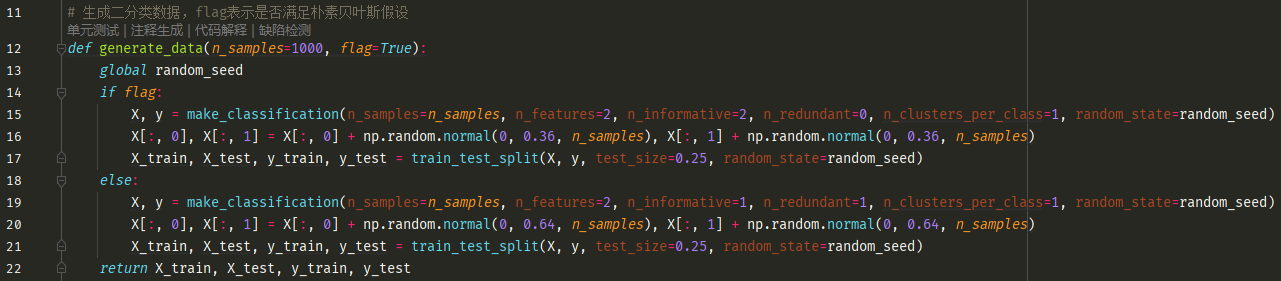
**4.2实验过程**

**4.2.1生成二分类数据集**

使用sklearn库生成并划分两个数据集，其特征分布分别满足和不满足朴素贝叶斯假设，并为两个数据集添加高斯噪声。

使用datasets模块的make\_classification函数生成1000个样本（默认类别数为2），为简化模型、方便绘图，设置特征数量为2、每个类别的簇数为1。对于特征分布满足朴素贝叶斯假设的数据集，设置信息特征数量为2、冗余特征数量为0；对于特征分布不满足朴素贝叶斯假设的数据集，设置信息特征数量为1、冗余特征数量为1，其中冗余特征由信息特征线性组合得到。

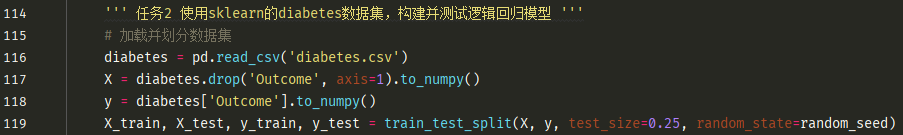
使用model\_selection模块的train\_test\_split函数随机将数据集75%划分为训练集，25%划分为测试集。



**4.2.2加载实际数据**

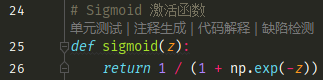
UCI数据集是一个常用的机器学习标准测试数据集，选择其中的Diabetes数据集进行模型测试。Diabetes数据集用于糖尿病领域的研究和预测，其包含442个样本、8个特征和2个输出类别，可用于二分类算法的测试。

从Github上下载Diabetes数据集的表格格式diabetes.csv，放到代码文件同一目录下，使用Pandas库加载并转化为Numpy数组格式，之后划分数据集。

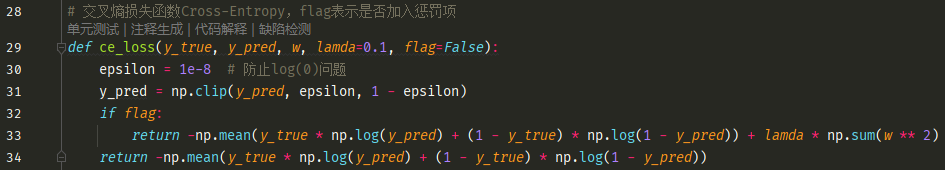


**4.2.3定义Sigmoid函数、损失函数和梯度计算函数**

Sigmoid函数输入为，输出为预测值。



损失函数ce\_loss()，输入为真实值y\_true、预测值y\_pred、权重w，lamda为惩罚项系数（默认0.1），flag代表是否加入惩罚项（默认False）。



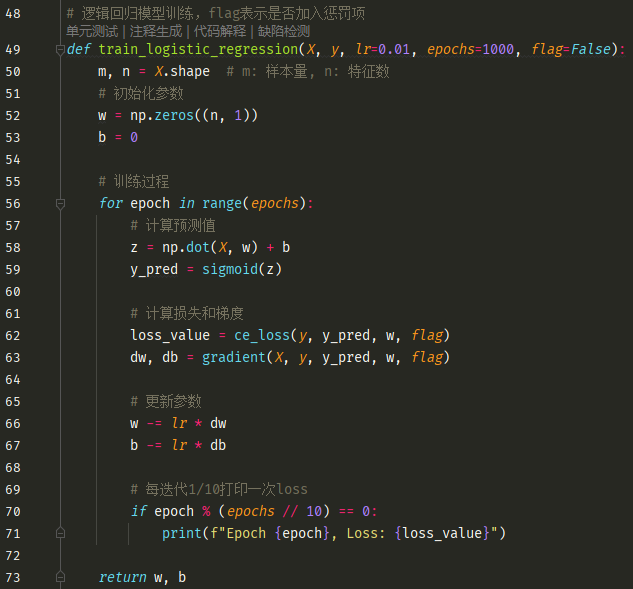
梯度计算函数gradient()，输入为特征矩阵X、真实值y\_true、预测值y\_pred、权重w，lamda为惩罚项系数（默认0.1），flag代表是否加入惩罚项（默认False）。每次迭代过程的下降梯度由损失函数对权重w和偏置b求偏导得到。



**4.2.4训练过程**

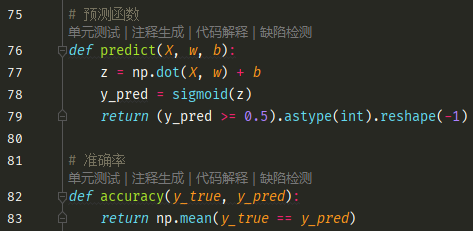
设置迭代次数epochs（默认1000）和学习率lr（默认0.01），初始化权重w和偏置b，然后开始训练。在每次迭代中，特征矩阵X与权重w相乘再加上偏置b，并通过Sigmoid函数的到预测值y\_pred，之后通过ce\_loss()函数和gradient()函数计算当前loss值和下降梯度，更新参数并进入下一次迭代。

可以选择有惩罚项或无惩罚项，若选择有惩罚项，则在计算loss和梯度时加入正则化项，loss为交叉熵加上，其中λ为惩罚项系数，为权重w的L2范数的平方；权重梯度为损失函数对权重w的偏导数加上，偏置梯度不变。

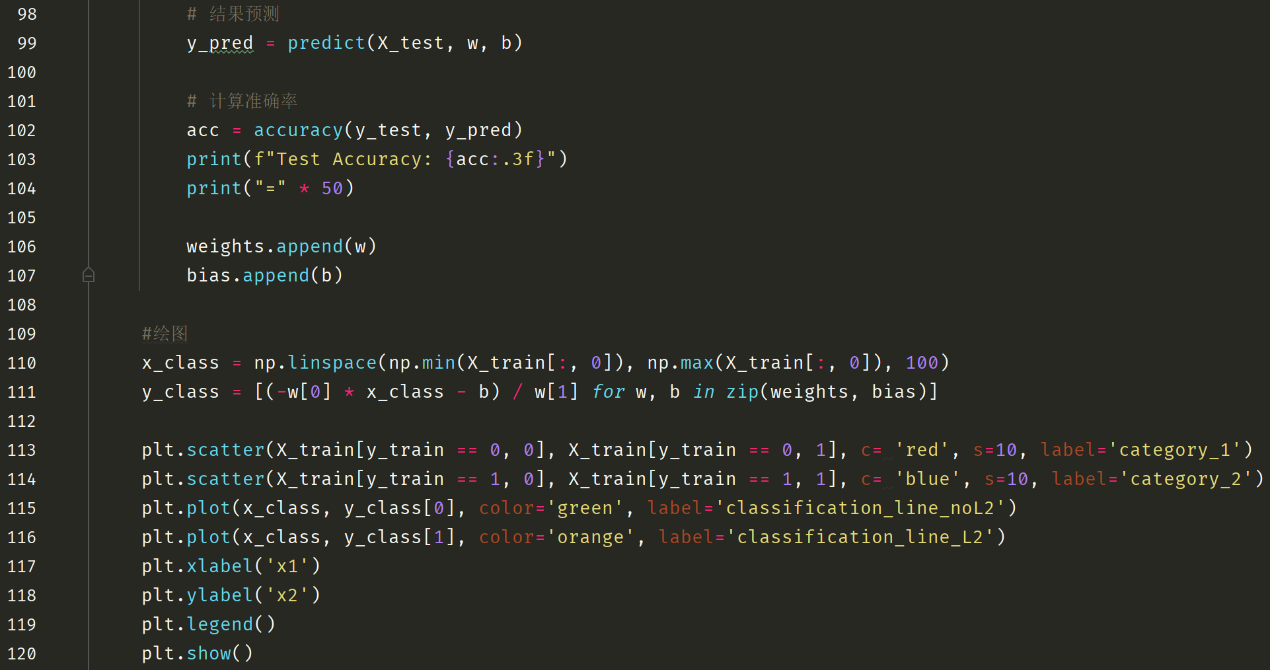


**4.2.5输出结果并绘图**

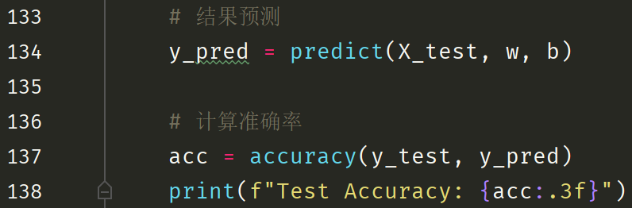
predict()函数使用得到的权重w和偏置b对测试集进行预测，并返回预测值。accuracy()函数比较真实值和预测值，计算准确率。



对于任务1，输出测试准确率，并使用matplotlib库进行绘图，将数据集两个类别可视化，并分别绘制未加入惩罚项和加入惩罚项后的分界曲线。



对于任务2，由于使用的糖尿病数据集特征数较多，不便于绘图，故只输出测试准确率。



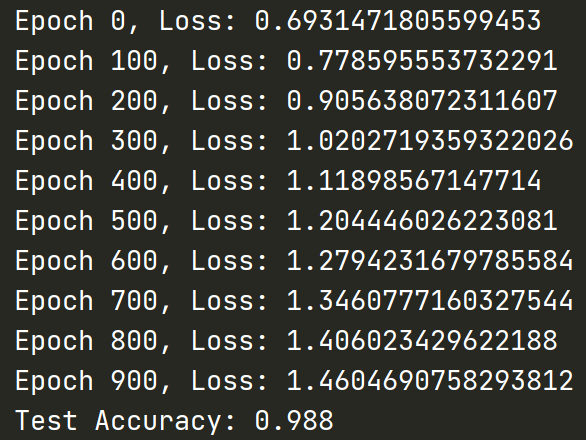
**4.3实验结果及分析**

**4.3.1任务1**

·特征分布满足朴素贝叶斯假设

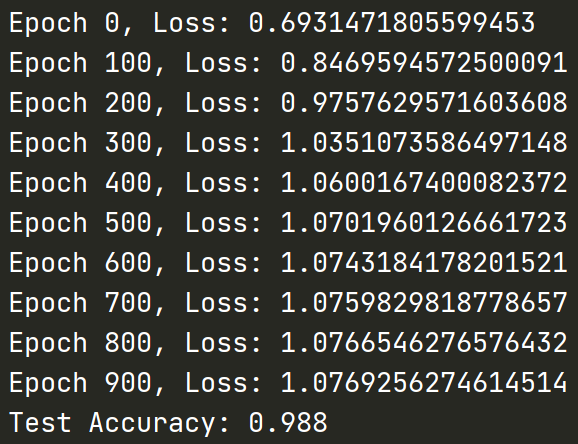
·无惩罚项

训练过程loss值变化及测试准确率如下：

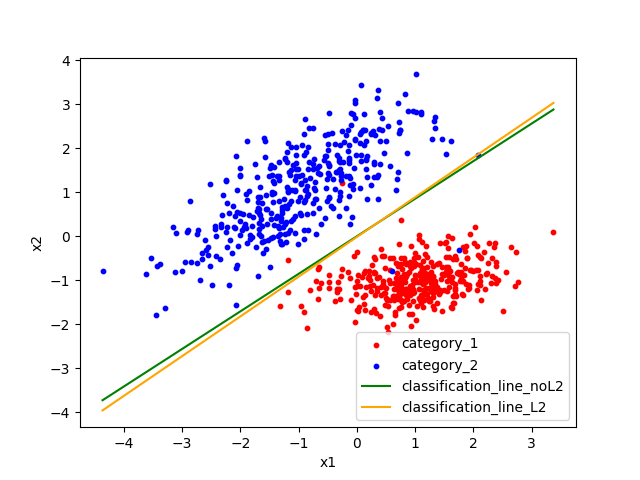


·有惩罚项

训练过程loss值变化及测试准确率如下：



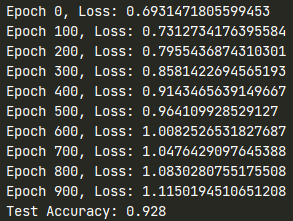
总体数据绘图如下：



·特征分布不满足朴素贝叶斯假设

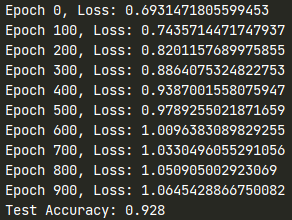
·无惩罚项

训练过程loss值变化及测试准确率如下：

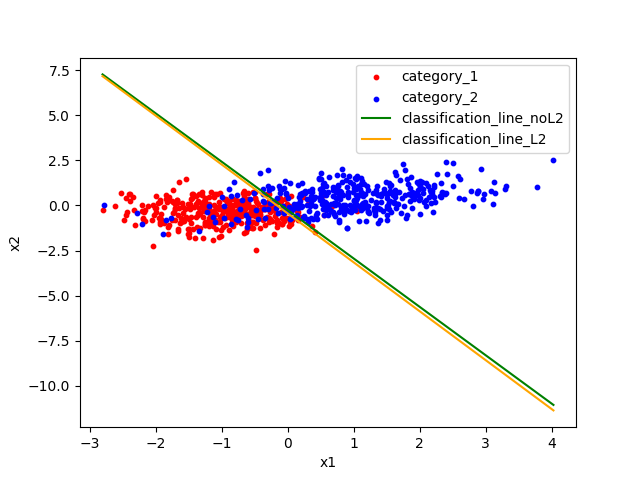


·有惩罚项

训练过程loss值变化及测试准确率如下：



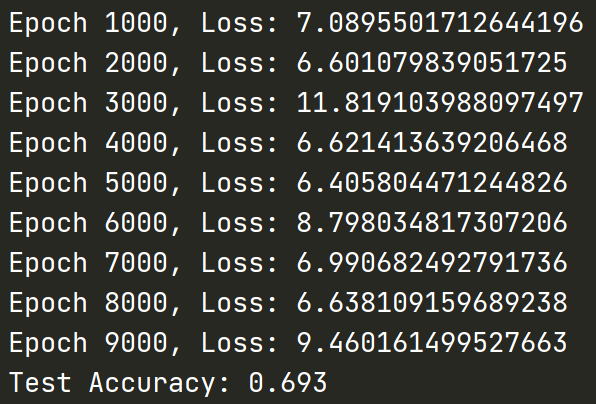
总体数据绘图如下：



**4.3.2任务2**

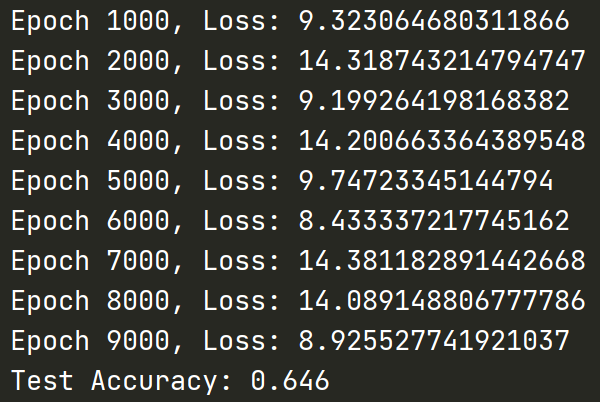
·无惩罚项

训练过程loss值变化及测试准确率如下：



·有惩罚项

训练过程loss值变化及测试准确率如下：



**五、实验总体结论**

逻辑回归是一种广义的线性回归分析模型，属于机器学习中的监督学习，主要用于解决二分类问题。其关键点在于Sigmoid函数，将计算值映射到(0, 1)之间，可以表示类别预测概率。逻辑回归具有简单高效、鲁棒性强、输出概率应用广泛的优点，但也存在不适用于非线性分布的数据、假设特征独立、需改进才能用于多分类问题的缺点。

使用交叉熵作为损失函数，梯度下降法作为优化方法，实现了逻辑回归模型在手动生成的二分类数据集和UCI中的糖尿病数据集上的训练和预测，并在测试集上达到了较好的分类效果，但也存在一些误分类点，可能与数据本身特点和模型较为简单有关，后续可使用其他分类模型或优化算法来提高性能。

在不满足朴素贝叶斯假设的数据集上，由于特征之间不是相互独立，导致有效特征变少，逻辑回归模型性能略有下降。

手动生成的二分类数据集，分类任务较为简单，迭代次数较多，模型出现过拟合现象，在加入惩罚项后过拟合现象得到一定程度的抑制。

**六、完整实验代码**

1. import numpy as np
2. import pandas as pd
3. import matplotlib.pyplot as plt
4. from sklearn.model\_selection import train\_test\_split
5. from sklearn.datasets import make\_classification
6. random\_seed = 43
7. np.random.seed(random\_seed)
8. *# 生成二分类数据，flag表示是否满足朴素贝叶斯假设*
9. def generate\_data(n\_samples=1000, flag=True):
10. global random\_seed
11. if flag:
12. X, y = make\_classification(n\_samples=n\_samples, n\_features=2, n\_informative=2, n\_redundant=0, n\_clusters\_per\_class=1, random\_state=random\_seed)
13. X[:, 0], X[:, 1] = X[:, 0] + np.random.normal(0, 0.36, n\_samples), X[:, 1] + np.random.normal(0, 0.36, n\_samples)
14. X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.25, random\_state=random\_seed)
15. else:
16. X, y = make\_classification(n\_samples=n\_samples, n\_features=2, n\_informative=1, n\_redundant=1, n\_clusters\_per\_class=1, random\_state=random\_seed)
17. X[:, 0], X[:, 1] = X[:, 0] + np.random.normal(0, 0.64, n\_samples), X[:, 1] + np.random.normal(0, 0.64, n\_samples)
18. X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.25, random\_state=random\_seed)
19. return X\_train, X\_test, y\_train, y\_test
20. *# Sigmoid 激活函数*
21. def sigmoid(z):
22. return 1 / (1 + np.exp(-z))
23. *# 交叉熵损失函数Cross-Entropy，flag表示是否加入惩罚项*
24. def ce\_loss(y\_true, y\_pred, w, lamda=0.5, flag=False):
25. epsilon = 1e-8  *# 防止log(0)问题*
26. y\_pred = np.clip(y\_pred, epsilon, 1 - epsilon)
27. if flag:
28. return -np.mean(y\_true \* np.log(y\_pred) + (1 - y\_true) \* np.log(1 - y\_pred)) + lamda \* np.sum(w \*\* 2)
29. return -np.mean(y\_true \* np.log(y\_pred) + (1 - y\_true) \* np.log(1 - y\_pred))
30. *# 梯度计算函数，flag表示是否加入惩罚项*
31. def gradient(X, y\_true, y\_pred, w, lamda=0.5, flag=False):
32. m = y\_true.shape[0] *# 样本量*
33. y = y\_true.reshape(-1, 1)
34. if flag:
35. dw = (1/m) \* np.dot(X.T, (y\_pred - y)) + lamda \* w  *# 权重的梯度*
36. db = (1/m) \* np.sum(y\_pred - y)                     *# 偏置的梯度*
37. return dw, db
38. dw = (1/m) \* np.dot(X.T, (y\_pred - y))  *# 权重的梯度*
39. db = (1/m) \* np.sum(y\_pred - y)         *# 偏置的梯度*
40. return dw, db
41. *# 逻辑回归模型训练，flag表示是否加入惩罚项*
42. def train\_logistic\_regression(X, y, lr=0.01, epochs=1000, lamda=0.5, flag=False):
43. m, n = X.shape  *# m: 样本量, n: 特征数*
44. *# 初始化参数*
45. w = np.zeros((n, 1))
46. b = 0
47. *# 训练过程*
48. for epoch in range(epochs):
49. *# 计算预测值*
50. z = np.dot(X, w) + b
51. y\_pred = sigmoid(z)
52. *# 计算损失和梯度*
53. loss\_value = ce\_loss(y, y\_pred, w, lamda=lamda, flag=flag)
54. dw, db = gradient(X, y, y\_pred, w, lamda=lamda, flag=flag)
55. *# 更新参数*
56. w -= lr \* dw
57. b -= lr \* db
58. *# 每迭代1/10打印一次loss*
59. if epoch % (epochs // 10) == 0:
60. print(f"Epoch {epoch}, Loss: {loss\_value}")
61. return w, b
62. *# 预测函数*
63. def predict(X, w, b):
64. z = np.dot(X, w) + b
65. y\_pred = sigmoid(z)
66. return (y\_pred >= 0.5).astype(int).reshape(-1)
67. *# 准确率*
68. def accuracy(y\_true, y\_pred):
69. return np.mean(y\_true == y\_pred)
70. *# 主程序*
71. if \_\_name\_\_ == "\_\_main\_\_":
72. is\_Bayes = True
73. is\_punish = True
74. ''' 任务1 手动生成数据集，构建并测试逻辑回归模型 '''
75. X\_train, X\_test, y\_train, y\_test = generate\_data(flag=is\_Bayes)
76. weights, bias = [], []
77. for flag in [False, True]:
78. *# 逻辑回归模型训练*
79. w, b = train\_logistic\_regression(X\_train, y\_train, flag=flag)
80. *# 结果预测*
81. y\_pred = predict(X\_test, w, b)
82. *# 计算准确率*
83. acc = accuracy(y\_test, y\_pred)
84. print(f"Test Accuracy: {acc:.3f}")
85. print("=" \* 50)
86. weights.append(w)
87. bias.append(b)
88. *#绘图*
89. x\_class = np.linspace(np.min(X\_train[:, 0]), np.max(X\_train[:, 0]), 100)
90. y\_class = [(-w[0] \* x\_class - b) / w[1] for w, b in zip(weights, bias)]
91. plt.scatter(X\_train[y\_train == 0, 0], X\_train[y\_train == 0, 1], c= 'red', s=10, label='category\_1')
92. plt.scatter(X\_train[y\_train == 1, 0], X\_train[y\_train == 1, 1], c= 'blue', s=10, label='category\_2')
93. plt.plot(x\_class, y\_class[0], color='green', label='classification\_line\_noL2')
94. plt.plot(x\_class, y\_class[1], color='orange', label='classification\_line\_L2')
95. plt.xlabel('x1')
96. plt.ylabel('x2')
97. plt.legend()
98. plt.show()
99. ''' 任务2 使用UCI的Diabetes数据集，构建并测试逻辑回归模型 '''
100. *# 加载并划分数据集*
101. diabetes = pd.read\_csv('diabetes.csv')
102. X = diabetes.drop('Outcome', axis=1).to\_numpy()
103. y = diabetes['Outcome'].to\_numpy()
104. X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.25, random\_state=random\_seed)
105. *# 逻辑回归模型训练*
106. w, b = train\_logistic\_regression(X\_train, y\_train, lr=0.05, epochs=10000, lamda=0.1, flag=is\_punish)
107. *# 结果预测*
108. y\_pred = predict(X\_test, w, b)
109. *# 计算准确率*
110. acc = accuracy(y\_test, y\_pred)
111. print(f"Test Accuracy: {acc:.3f}")

**七、参考文献**

[1] 刘远超. 深度学习基础: 高等教育出版社, 2023.

[2] 阿库塔姆. Logistic回归（逻辑回归）及python代码实现[EB/OL].CSDN博客, 2023-07-03 [2024-11-22]. https://blog.csdn.net/weixin\_50744311/article/details/131523136.

[3] 思绪无限.UCI数据集整理（附论文常用数据集）[EB/OL].CSDN博客, 2022-09-27[2024-11-22]. https://blog.csdn.net/qq\_32892383/article/details/82225663.