

**计算机视觉实验一实验报告**

|  |  |
| --- | --- |
| 姓 名： | 王嘉成 |
| 学 院： | 计算机学院 |
| 专 业： | 计算机科学与技术专业 |
| 班 级： | 计算机2207班 |
| 学 号： | U202215576 |
| 指导教师： | 刘康 |

|  |  |
| --- | --- |
| 分数 |  |
| 教师签名 |  |

2024年 12 月 17 日

目 录

[一、实验要求 1](#_Toc97642866)

[二、数据处理 2](#_Toc97642870)

[1、数据集加载与概述 2](#_Toc97642871)

[2、数据的拟合分布 2](#_Toc97642872)

[3、数据集拆分 3](#_Toc97642873)

[三、实验框架 4](#_Toc97642874)

[1、实验环境 4](#_Toc97642875)

[2、网络结构 4](#_Toc97642876)

[3、损失函数 5](#_Toc97642877)

[四、实验结果 6](#_Toc97642874)

[1、最佳超参数设置 6](#_Toc97642875)

[2、结果与分析 6](#_Toc97642876)

[五、最终结果分析 7](#_Toc97642874)

# 一、实验要求

任务要求:设计一个前馈神经网络，对一组数据实现分类任务，

下载“dataset.csv”数据集，其中包含四类二维高斯数据和它们的标签。设计至少含有一层隐藏层的前馈神经网络来预测二维高斯样本(data1,data2)所属的分类label。这个数据集需要先进行随机排序，然后选取90%用于训练，剩下的10%用于测试。

注意事项:

1.深度学习框架任选。

2.鼓励尝试不同的网络层数、不同的神经元个数、使用不同的激活函数等，观察网络性能。

3.实验报告需包含神经网络架构、每一轮mini-batch训练后的模型在训练集和测试集上的损失、最终的训练集和测试集准确率，以及对应的实验分析。

4.将代码和实验报告打包成ZIP压缩包，以“姓名-学号-实验报告#”命名，比如“张三-2020XXX-实验报告一.zip”，提交到学习通(https://smarthouse.hust.edu.cn/)。

5.截止时间为 12 月 18 号下午 2:00。二、数据处理

## 、数据集加载与概述

首先，我们加载了实验一的数据集，该数据集为一CSV文件，包含了不同标签的二维高斯分布数据。我们使用pandas库读取数据，并进行初步查看,在输出的DataFrame中，包含了特征feature1和feature2，以及与每个样本相关联的标签label。该标签列包含了4个不同的标签值，表示四种类别的二维高斯数据。

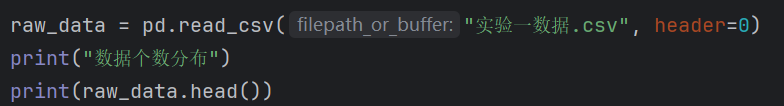


图 1 代码段1

## 、数据的拟合分布

进一步分析每个标签数据的统计特性，我们计算了每个类别的均值和标准差，并进行了3D高斯分布拟合图的绘制。使用groupby()函数分别计算各标签的均值和标准差，然后，利用scipy.stats.multivariate\_normal中的pdf函数来生成每个标签的拟合分布，并绘制3D曲面图，展示每个类别数据的高斯分布拟合情况。



图 2 代码段2

结果如图所示，可发现四类样本均大致满足高斯分布，但是分布较为分散，较难和其它样本完全分开。

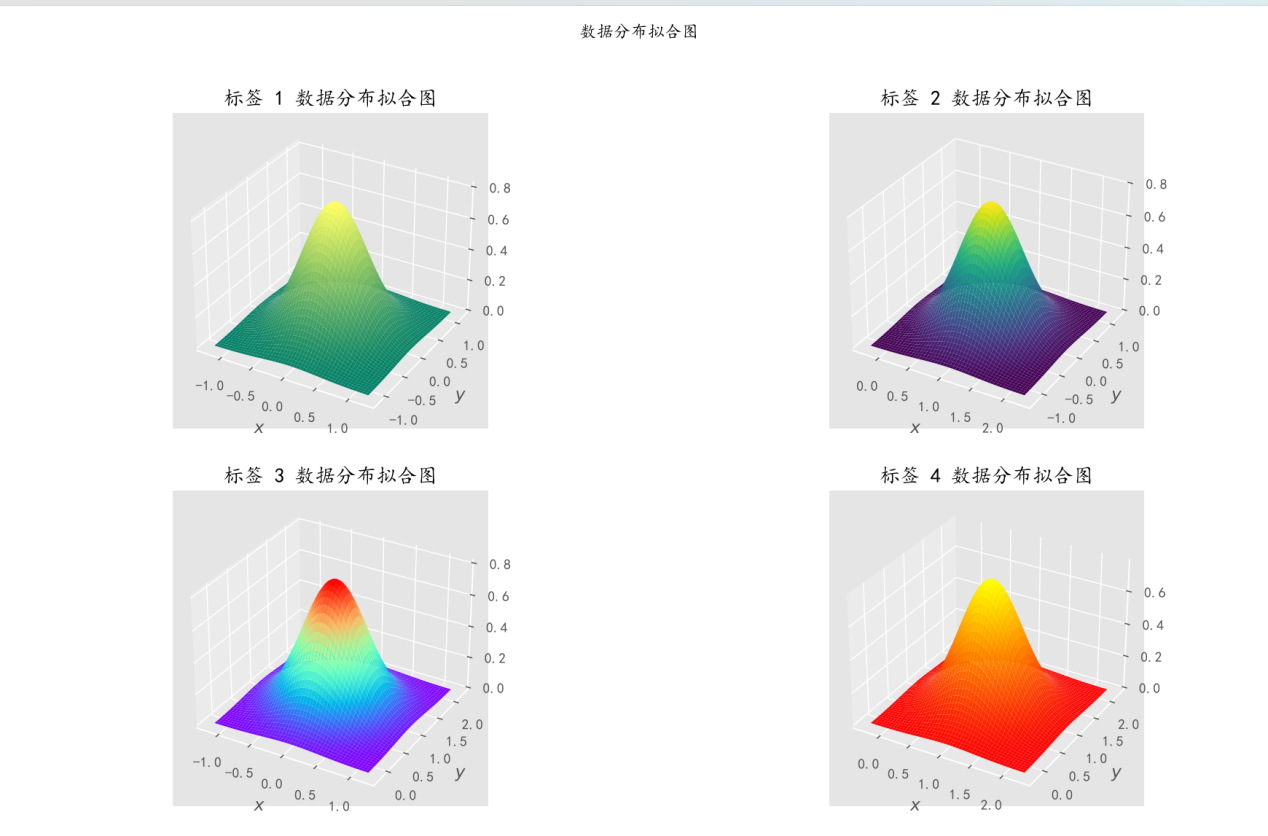


图 3 数据分布拟合图

## 、数据集拆分

为了训练神经网络模型，我们将数据集划分为训练集和验证集。具体做法是使用sklearn.model\_selection.train\_test\_split函数，将数据按照params["valid\_ratio"]的比例进行随机划分，其中90%用于训练10%用于验证，这里，我们使用TensorDataset将训练集和验证集的特征和标签封装成PyTorch数据集对象，以便后续通过DataLoader加载数据。

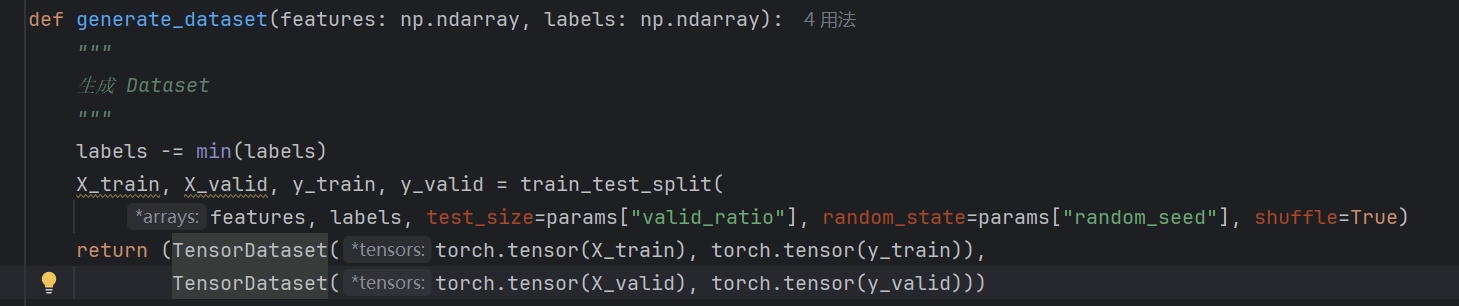


图 4 代码段3

# 三、实验框架

## 、实验环境

操作系统：windows 11

CUDA：11.8

显卡：NVIDIA GeForce RTX 3070 Ti Laptop GPU

深度学习框架：PyTorch

开发工具：PyCharm Community Edition 2024.3.1

## 、网络结构

本实验中，我们设计了一个前馈神经网络，用于解决多分类问题。该神经网络由输入层、多个隐藏层和一个输出层组成。网络使用了ReLU或LeakyReLU激活函数，并引入了批量归一化和Dropout技术，以防止过拟合并加速收敛。

模型的输入层接受2个特征，输入维度为2。输入层通过线性层将输入映射到第一个隐藏层的维度。这里，input\_size=2，而第一个隐藏层的节点数由 hidden\_layers\_list[0]确定。

该神经网络包含多个隐藏层，其节点数通过hidden\_layers\_list提供。在本模型中，隐藏层的节点数分别为50和30。每个隐藏层都包括以下几个部分：

·线性层：将前一层的输出映射到当前层的输出。

·批量归一化：对每一层的输出进行标准化，以减少训练过程中的协方差偏移。

·激活函数：通过引入非线性函数，使网络能够学习更复杂的特征表示。

·Dropout：在训练过程中随机丢弃一定比例的神经元，以防止过拟合。

输出层由一个线性层构成，将网络的输出映射至4个类别，输出层的结果是一个包含4个元素的向量，表示每个类别的预测得分。

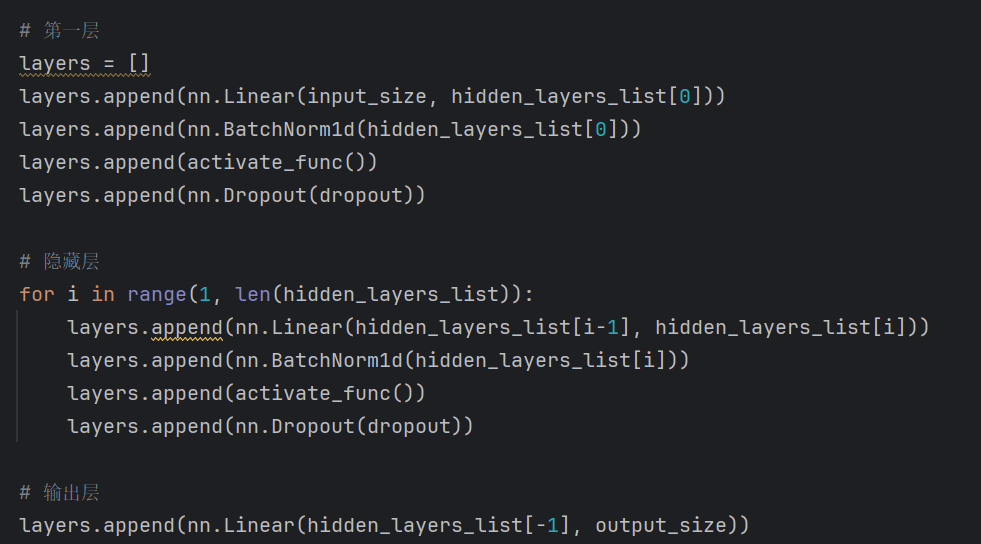


图 5 代码段4

## 、损失函数

在本任务中，我们进行的是多类别分类问题，因此损失函数采用交叉熵损失函数。交叉熵损失函数用于评估模型输出的类别概率分布与真实标签分布之间的差异。

假设有个样本，每个样本有个类别。对于第个样本，模型输出的概率分布是,其中是模型预测样本属于类别的概率。实标签是一个一热编码向量，表示样本的真实类别。如果样本属于类别，则，其他类别的标签（对于）。

交叉熵损失函数的公式为：

为了防止模型过拟合，我们在损失函数中加入了正则化项。在本实验中，我们尝试了L1正则化和L2正则化两种方法，L1正则化通过在损失函数中添加模型参数的1-范数来对参数进行约束，L1正则化有助于产生稀疏的模型参数矩阵，从而简化模型并实现特征选择，所以我们最终选择了L1正则化。L1正则化项的计算公式为：

结合交叉熵损失和 L1 正则化项，最终的总损失函数计算公式为：



# 实验结果

## 1、最佳超参数设置

在本次实验中，为了确保实验结果的可复现性，我们将随机数种子固定为1024，并利用GPU加速进行模型训练。接下来，我们通过网格搜索对超参数进行了优化，探索了13824种参数组合，最终得到了最优的超参数配置。

在网格搜索的过程中，最佳的超参数配置为：

* 激活函数：ReLU
* Dropout率：0.4
* L2 正则化权重：0
* L1 正则化权重：1e-3
* 批量大小：360
* 学习率：1e-3
* 隐藏层结构：[100]

该配置下，模型在验证集上的准确率和 F1 分数均达到最佳值。

在网络架构方面，我们采用了一个含有单层隐藏层的全连接网络，隐藏层的神经元数量为50。激活函数选择了LeakyReLU，负半轴的斜率设置为0.01。

训练过程中，我们进行了200个epoch的训练，并且由于显存足够，我们使用了全数据集进行训练，batch\_size设置为3600。学习率设定为1.0，dropout概率为0.4，L1正则化系数为0.001。

## 结果与分析

在本实验中，我们使用了预设的超参数对模型进行训练，并在每一轮训练后评估模型在验证集上的表现，验证过程采用了准确率和宏平均F1分数作为主要评价指标。为了保证模型在验证集上的最佳性能，我们在训练过程中实时监控验证集的表现，并选择验证集上准确率最高的模型参数进行保存。此外，为了更全面地评估模型的训练情况，我们还记录了训练集损失和验证集损失的变化，并将其可视化，同时，验证集上的准确率和宏平均F1分数的变化趋势分别展示在图8和图9中。这些图表不仅帮助我理解模型在训练过程中的表现，也为后续的超参数调优和模型优化提供了有价值的参考。

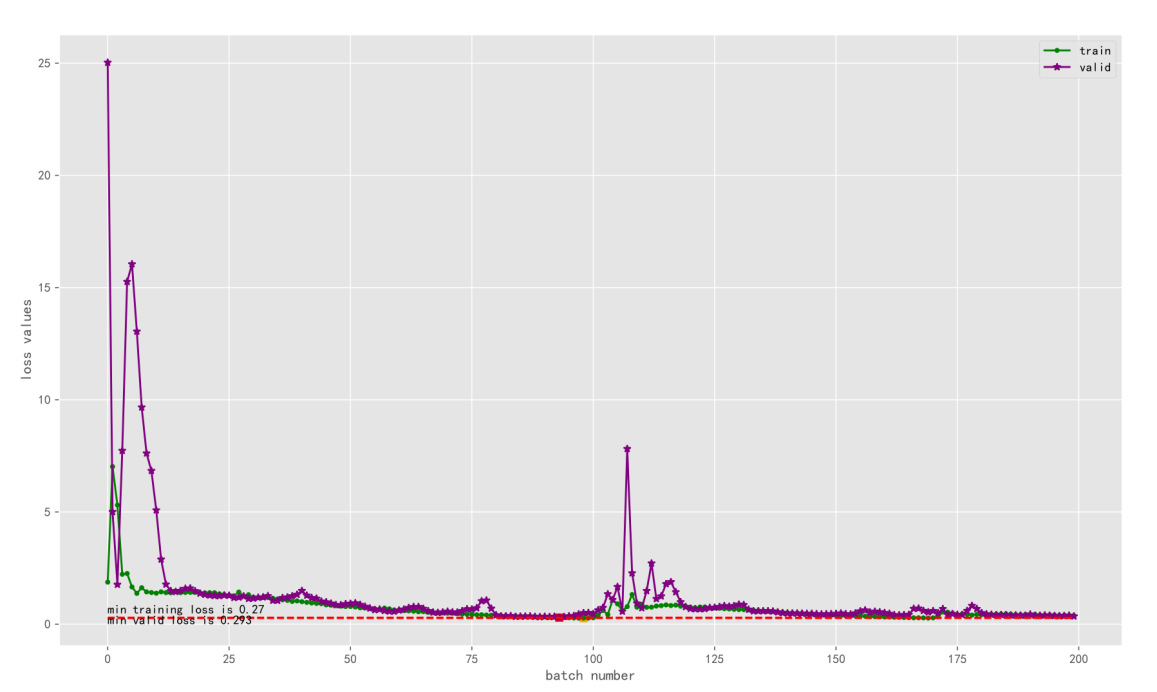


图 6 损失函数变化图



图 7 acc变化图

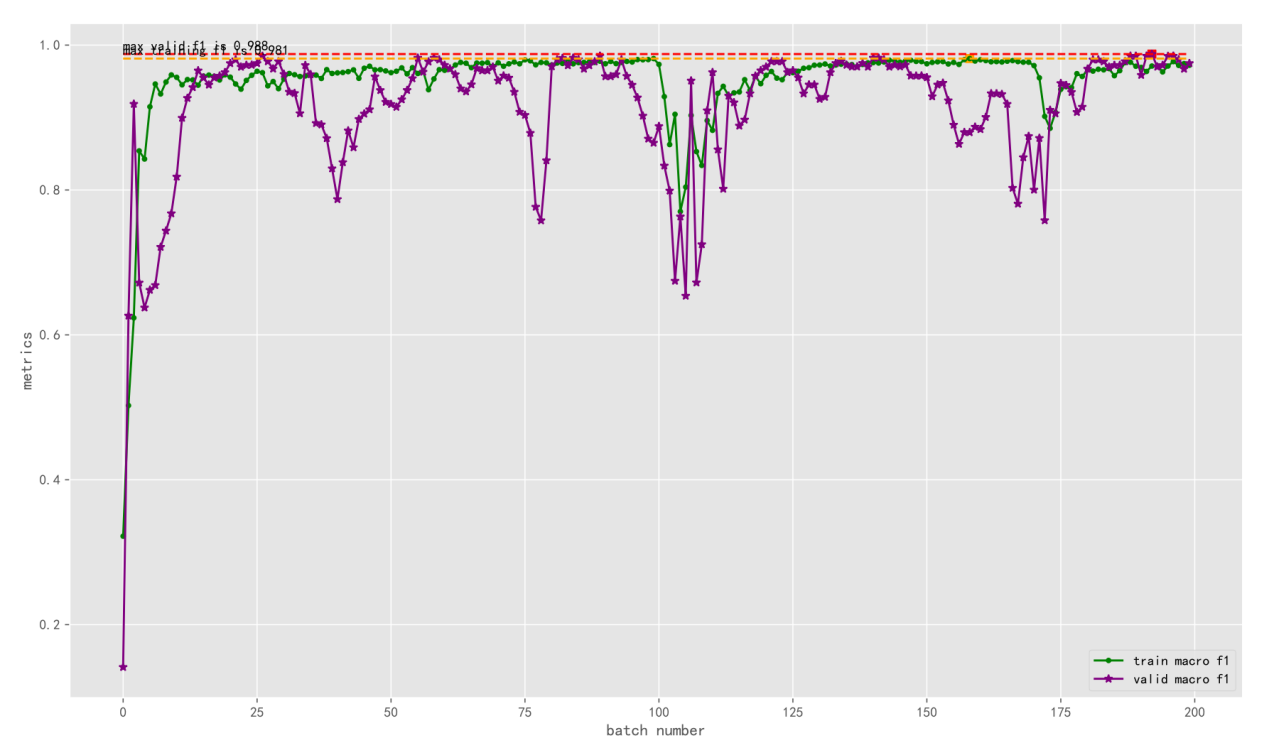


图 8 f1变化图

为了更清晰地展示模型的分类效果，我们绘制了最优模型的分类边界图。通过观察分类边界，可以直观地看到模型在不同类别之间的区分情况，分类边界显示出模型能够较好地将数据划分为不同的类别，且对于一些复杂样本，模型的分类能力依然保持较高的准确性，分类边界的平滑度也反映了模型对不同数据特征的适应能力。

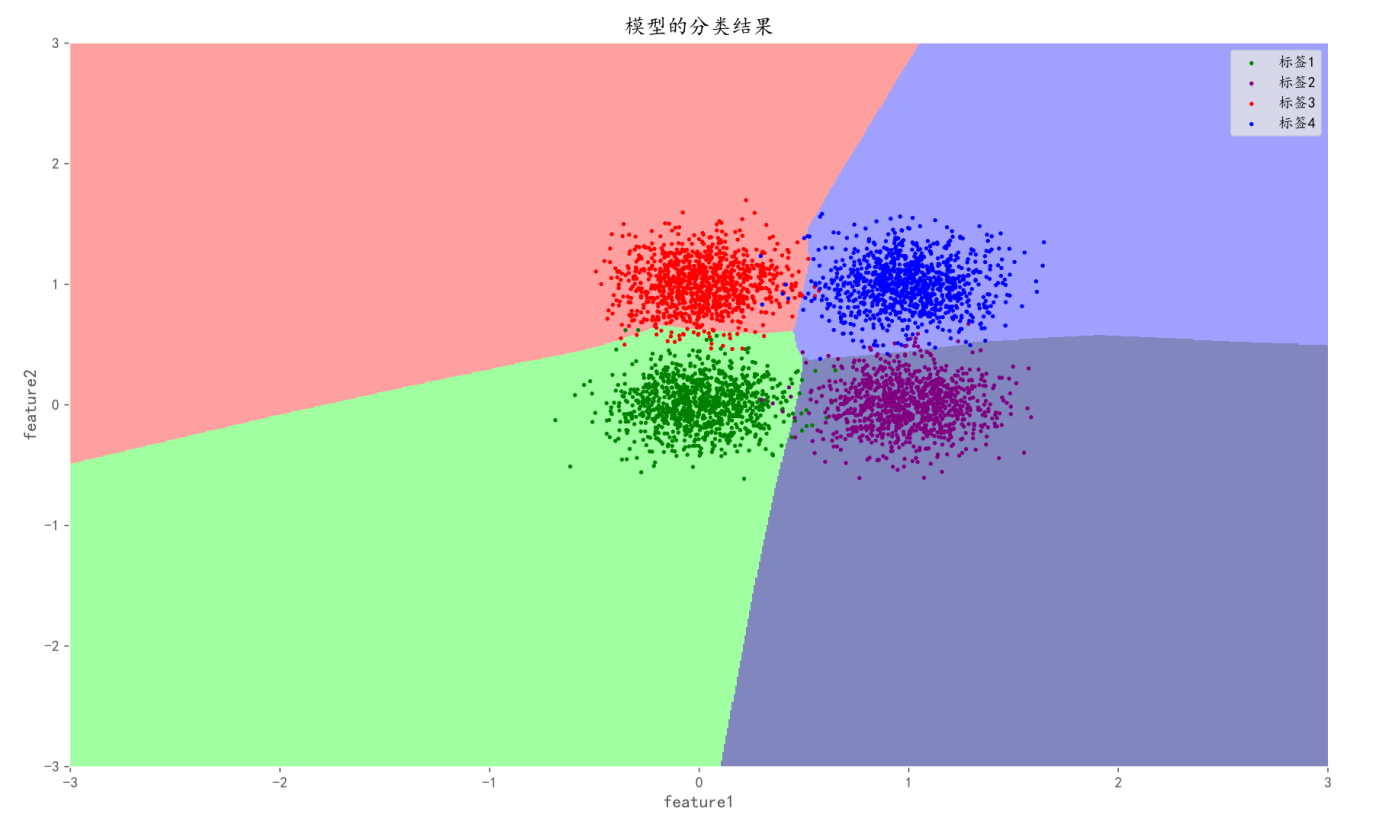


图 9 模型分类结果

从各项评估指标来看，选取的最优模型在验证集上表现出色，具体来说，验证集上的准确率和宏观 F1分数均达到了较高的水平，我们选取了验证集上准确率最高的模型参数并保存，以确保模型在实际应用中的表现最佳。

通过在验证集上的多次评估，模型显示出较强的泛化能力，即使在不同超参数配置下，模型也能较好地适应各种数据分布，保持稳定的性能表现。这表明我们的训练策略以及正则化技术有效地防止了过拟合，使得模型具有较好的泛化能力。

# 最终结果分析

## 各类别相关指标展示

类别0的精确度为0.96，召回率为1.00，F1分数为0.98，表示该类别具有高召回率，但精确度略有下降。

类别1的精确度为1.00，召回率为0.99，F1分数为1.00，显示该类别分类性能非常优秀。

类别2的精确度为1.00，召回率为0.96，F1分数为0.98，精确度高，召回率稍低，但整体表现良好。

类别3的精确度为0.99，召回率为1.00，F1分数为0.99，表明该类别也具有较好的分类效果。

整个模型的准确率为0.99，表示99%的样本被正确分类，整个模型的准确率为0.99，表示99%的样本被正确分类，加权平均则根据每个类别的支持度对结果进行加权，得到加权后的精确度、召回率和F1分数，结果为0.99，表现出模型在不同类别中具有相对均衡的分类能力。

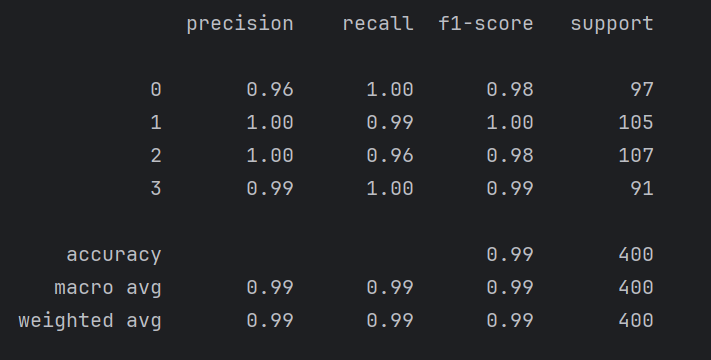


图 10 各类指标

## 不同的 L1 正则化系数对准确率的影响

从图中可以观察到，随着L1正则化系数的增加，模型的准确率呈现先缓慢上升，随后急剧下降的趋势。在L1正则化系数较小时，模型的准确率保持在较高水平，约为0.993。这表明在该阶段，模型能够有效地保留大部分特征，并取得较好的预测性能。随着 L1 正则化系数的逐渐增大，准确率略微上升至0.995，这可能是由于正则化的适度作用使模型的复杂度得以控制，并减少了过拟合现象。

然而，当L1正则化系数进一步增大时，模型的准确率急剧下降，最终降至0.71。此时，正则化作用过强，导致模型的权重被过度稀疏化，甚至丢弃了部分有用的特征，从而使得模型的表现大幅度下降，无法有效捕捉数据中的模式。

综上所述，从图中的变化趋势来看，L1正则化系数为0.001时，模型的准确率达到最佳效果，因此我们选择这一参数作为最终模型的正则化系数。

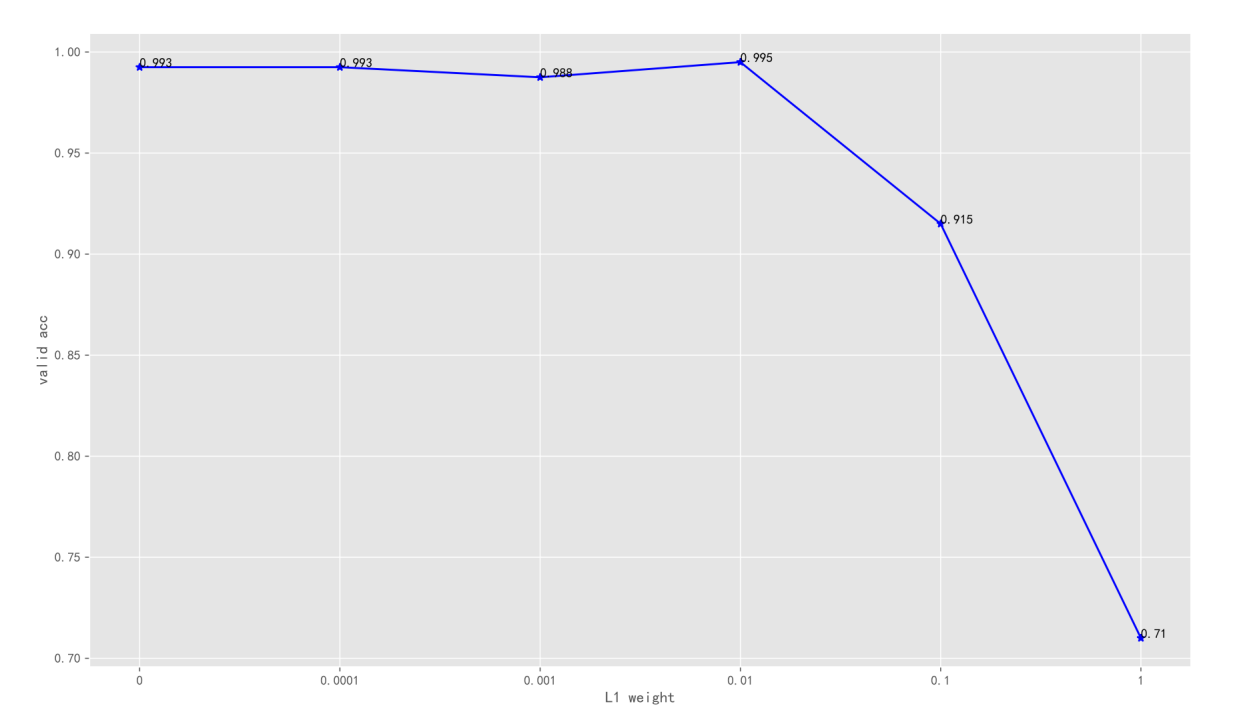


图 11 不同L1系数对结果的影响