

|  |  |  |  |
| --- | --- | --- | --- |
| **数字图像处理PBL报告** | | | |
| 基于MATLAB与PYTHON的卫星遥感图像的分类 | | | |
|  | | | |
|  | 专 业： | 通信工程 |  |
|  | 指导老师： | 董光辉 |  |
|  | 组 长： | 范楚鑫 |  |
|  | 组 员： | 姜宇 线全博 马浩然 |  |

2024年12月

**一、项目分工说明**

**▪马浩然：文献查阅，数据集收集，PPT 制作**

**▪范楚鑫：示例程序功能注释，运行结果分析，设计报告撰写**

**▪姜宇：开源数据集卫星遥感图像分类程序设计**

**▪线全博：设计报告撰写，PPT 讲解，开源数据集分类结果分析**

**二、示例程序功能注释**

### **（一）环境初始化与数据加载模块**

这段代码主要负责初始化Matlab工作环境并加载卫星遥感图像数据集，为进一步的图像处理和分析奠定基础。通过清除工作空间、设置紧凑的输出格式和关闭所有图形窗口，代码确保了一个干净且高效的工作环境。接着，使用`load`语句加载如Salinas场景的高光谱图像数据集到变量`HSI\_Image`中，同时提供了灵活切换不同数据集的选项。此外，代码通过`size`函数获取图像的空间分辨率和波段数量，这些关键信息对于理解图像结构和进行后续的光谱分析至关重要。最后，计算类别数量为分类任务提供基础，确保了后续操作的顺利进行。

（1）环境初始化

|  |
| --- |
| clear  format compact  close all |

（2）数据预加载（包含切换数据集）

|  |
| --- |
| %切换数据集：加载名为"HSI\_Image”的Salinas超立方体数据  load SA.mat %加载Salinas场景的高光谱图像数据  %load Houston.mat %加载Houston场景的高光谱图像数据  %load paviaU.mat %加载PaviaU场景的高光谱图像数据  % [切换数据集操作方法：  % 现在是SA.mat数据集的加载和处理；  % 如果要切换到Houston.mat数据集，则是  % %切换数据集：加载名为"HSI\_Image”的Salinas超立方体数据  % %load SA.mat %加载名为"HSI\_Image"的Salinas超立方体数据  % load Houston.mat %加载名为"HSI\_Image"的Salinas超立方体数据Houston数据集  % %load paviaU.mat %加载名为"HSI\_Image"的Salinas超立方体数据 paviaU数据集  % % HSI\_Image=houstonHS; %切换数据集  % 切换到PaviaU类似。]  % 从加载的数据中提取Salinas图像  HSI\_Image=Salinas\_Image; %切换数据集  %获取图像的空间分辨率/维度（p, n）和波段数量/光谱带数（l）  [p,n,l]=size(HSI\_Image)  % 计算类别数量  c = numel(unique(TR))-1; % 减去1是因为有一个特殊的值用于表示未知类别 |

### **数据集转换与分离模块**

该模块的主要功能是将原始数据集中的一维像素集合转换为二维图像格式，并从中提取出训练集、测试集和操作集的像素信息。通过直接赋值的方式，模块将Training\_Set、Test\_Set和Operational\_Set数组分别对应到Training\_Set\_Image、Test\_Set\_Image和Operational\_Set\_Image变量中，利用已知的空间维度p和n（都等于150）来简化转换过程。接着，初始化三个三维数组Train、Test和Operational，并通过逐波段与掩码相乘的方式，从HSI\_Image中提取出对应集合的像素值。这些操作使得Train、Test和Operational数组仅在对应集合的像素位置上包含非零值，从而有效地分离出不同集合的数据，为后续的特征提取和分类分析做好准备。

**1.数据集重构**

将一维数组(包含训练集、测试集、操作集)重构为二维图像格式

|  |
| --- |
| % 将训练集、测试集和操作集的数据重构为150x150的图像格式  Training\_Set\_Image=TR;  Test\_Set\_Image=TE;  Operational\_Set\_Image=TE; |

训练集、测试集和操作集的像素值赋值给对应的图像变量

**2.提取训练、测试和操作集的像素值**

（1）初始化三维数组：

|  |
| --- |
| Train=zeros(p,n,l); % % 这是一个三维数组，仅在训练像素处包含非零值  Test=zeros(p,n,l); % 这是一个三维数组，仅在测试像素处包含非零值  Operational=zeros(p,n,l); % 这是一个三维数组，仅在操作像素处包含非零值 |

（2）填充数组：通过逐元素相乘的方式，将 HSI\_Image 的每个波段与相应的掩码（如 Training\_Set\_Image>0）相乘，得到训练、测试和操作集的三维数组。

|  |
| --- |
| for i=1:l  % 将HSI\_Image的每个波段与掩码相乘，以提取训练、测试和操作集的像素值  % "Training\_Set\_Image>0", which identifies only the training vectors.  Train(:,:,i)=HSI\_Image(:,:,i).\*(Training\_Set\_Image>0);  Test(:,:,i)=HSI\_Image(:,:,i).\*(Test\_Set\_Image>0);  Operational(:,:,i)=HSI\_Image(:,:,i).\*(Operational\_Set\_Image>0);  End |

### **数据整理与数组合并模块**

数据整理与数组合并模块的主要任务是构建和整合用于机器学习分类的特征向量、响应向量和位置信息。该模块首先初始化六个空数组来分别存储训练集、测试集和操作集的特征向量、响应向量和位置信息。然后，通过嵌套循环遍历图像的每个像素点，检查每个像素是否属于训练集、测试集或操作集，并提取相应的特征向量、响应向量和位置信息。遍历完成后，模块将这些集合的特征向量、响应向量和位置信息合并到`Combined\_array`、`Combined\_array\_response`和`Combined\_array\_pos`中，形成一个统一的数据结构。`Combined\_array\_pos`数组在合并后进行转置操作，以确保位置信息的格式一致性。这一模块的操作将原始的高光谱图像数据转换成适合机器学习算法处理的结构化数据形式，为后续的分类和分析工作提供了便利。

**1.分类数组构建**

**初始化六个数组**,分别用于存储训练集、测试集和操作集的特征向量、响应向量和位置信息。

|  |
| --- |
| % 初始化数组，用于存储训练、测试和操作集的特征向量、响应向量和位置信息  Train\_array=[]; %这是所需的204xN数组  Train\_array\_response=[]; % 此向量保存每个训练像素的标签  Train\_array\_pos=[]; % 此数组保存训练像素在图像中的位置（以行表示）  Test\_array=[]; %这是所需的204xN数组  Test\_array\_response=[]; % 此向量保存每个测试像素的标签  Test\_array\_pos=[]; %此数组保存测试像素在图像中的位置（以行表示）  Operational\_array=[]; %这是所需的204xN数组  Operational\_array\_response=[]; % 此向量保存每个操作像素的标签  Operational\_array\_pos=[]; % 此数组保存操作像素在图像中的位置（以行表示） |

循环嵌套，**遍历图像的每个像素点**，检查是否属于训练集、测试集或操作集（通过掩码Training\_Set\_Image、Test\_Set\_Image和Operational\_Set\_Image中的值是否大于0来判断），并**提取相应的特征向量、响应向量和位置信息**。

对于每个像素点，如果它属于训练集，模块将提取该像素在Train数组中对应的特征向量，并将该像素的类别标签和位置信息添加到Train\_array\_response和Train\_array\_pos中。同样的操作也适用于测试集和操作集

|  |
| --- |
| % 遍历图像的每个像素，提取训练、测试和操作集的特征向量、响应向量和位置信息  for i=1:p  for j=1:n  if(Training\_Set\_Image(i,j)>0) % % 检查 (i, j) 像素是否为训练像素  Train\_array=[Train\_array squeeze(Train(i,j,:))];  Train\_array\_response=[Train\_array\_response Training\_Set\_Image(i,j)];  Train\_array\_pos=[Train\_array\_pos; i j];  end  if(Test\_Set\_Image(i,j)>0) % % 检查 (i, j) 像素是否为测试像素  Test\_array=[Test\_array squeeze(Test(i,j,:))];  Test\_array\_response=[Test\_array\_response Test\_Set\_Image(i,j)];  Test\_array\_pos=[Test\_array\_pos; i j];  end  if(Operational\_Set\_Image(i,j)>0) % % 检查 (i, j) 像素是否为操作像素  Operational\_array=[Operational\_array squeeze(Operational(i,j,:))];  Operational\_array\_response=[Operational\_array\_response Operational\_Set\_Image(i,j)];  Operational\_array\_pos=[Operational\_array\_pos; i j];  end  end  end |

**2.合并数组**

遍历完所有像素点后，模块将训练集、测试集和操作集的特征向量、响应向量和位置信息分别**合并**到Combined\_array、Combined\_array\_response和Combined\_array\_pos中。这样，就得到了一个包含所有像素特征向量、响应向量和位置信息的统一数据结构，为后续的分类和分析工作提供了便利。

Combined\_array\_pos数组则是合并后进行了转置操作，以确保位置信息的**格式一致性**。通过这一模块的操作，原始的高光谱图像数据被转换成了适合机器学习算法处理的结构化数据形式。

|  |
| --- |
| % 合并训练、测试和操作集的数据  Combined\_array = [Train\_array Test\_array Operational\_array];  Combined\_array\_response = [Train\_array\_response Test\_array\_response Operational\_array\_response];  Combined\_array\_pos = [Train\_array\_pos' Test\_array\_pos' Operational\_array\_pos'];  Combined\_array\_pos = Combined\_array\_pos'; |

### **计算高斯最大似然估计**

该模块的功能是计算高斯最大似然估计，通过遍历所有类别和光谱带，使用高斯混合模型为每个类别在每个光谱带上估计出均值和协方差参数。具体来说，该模块利用`Gaussian\_ML\_estimate`函数计算每个类别在每个光谱带上的平均光谱响应（均值m\_hat）和光谱响应变化（协方差S\_hat），这些统计特征对于基于概率的分类方法，如朴素贝叶斯分类器，至关重要。它们使得分类算法能够准确地评估像素与各类别之间的关系，提高分类的准确性和可靠性。通过这种高斯最大似然估计，模块为精确分类提供了坚实的基础，更好地捕捉和利用高光谱图像数据中的光谱特性。

**每个类别i和每个波段j：**

|  |
| --- |
| % 使用高斯混合模型估计每个类别的每个光谱带的参数  for i=1:c % 计算每个类别的每个光谱波段（204个中的）  for j=1:l  [m\_hat(i,j), S\_hat(i,j)]=Gaussian\_ML\_estimate(Train\_array(j,find(Train\_array\_response==i)));  end  end |

### **分类算法实现与结果评估模块**

**1.朴素贝叶斯分类器**

利用朴素贝叶斯分类器对测试集和合并集进行分类，并评估分类性能。该模块首先使用从高斯分布参数中得到的均值和协方差，对测试集进行分类，并计算混淆矩阵来评估分类准确率。随后，对合并集进行分类，并通过遍历像素位置信息，将分类结果填充到一个二维图像中，最后使用imagesc函数显示这个图像，直观地展示分类效果。这一过程不仅实现了数据的分类处理，还提供了一种直观的方法来评估和展示分类器的性能，对于理解模型效果和进行后续优化具有重要意义。

（1）分类

|  |
| --- |
| % 使用朴素贝叶斯分类器对测试集进行分类  classified = naive\_bayes\_classifier(S\_hat,m\_hat,c,l,Test\_array); |

计算混淆矩阵和分类准确率

通过计算混淆矩阵来评估分类准确率，混淆矩阵是一种特别有用的工具，它显示了分类结果与真实标签之间的匹配情况，从而可以计算出分类的准确率，即正确分类的像素数占总像素数的比例。

|  |
| --- |
| % 计算混淆矩阵和分类准确率  confusion\_matrix = confusion\_matrix(classified,Test\_array\_response,c)  trace(confusion\_matrix)/sum(sum(confusion\_matrix)) % 准确率 |

对合并数组进行分类并显示分类结果

模块对合并集Combined\_array进行分类，并将分类结果可视化。通过遍历Combined\_array\_pos中的位置信息，模块在Result\_Set\_Image中填充分类结果，创建一个分类图像，然后使用imagesc函数显示这个图像，直观地展示朴素贝叶斯分类器的分类效果。

|  |
| --- |
| % 使用朴素贝叶斯分类器对合并集进行分类  classified = naive\_bayes\_classifier(S\_hat,m\_hat,c,l,Combined\_array);  % 根据分类结果创建分类图像  Result\_Set\_Image = zeros(p,n); %Creating the picture  for i=1:length(Combined\_array\_pos)  Result\_Set\_Image(Combined\_array\_pos(i,1),Combined\_array\_pos(i,2)) = classified(1,i);  end  % 显示朴素贝叶斯分类器的分类结果  figure('Name','Naive Bayes Classifier'); imagesc(Result\_Set\_Image); |

**2.欧几里得分类器**

基于欧几里得距离。它计算测试样本与每个类别中心（这里用均值m\_hat表示）的欧几里得距离,并根据距离的远近将样本分配给最近的类别。

同样计算混淆矩阵和准确率来评估性能。通过对比模型的分类预测结果与真实标签 来生成混淆矩阵，根据混淆矩阵计算准确率。

(1)清除之前的分类结果和混淆矩阵

|  |
| --- |
| % 清除临时变量  clear classified;  clear confusion\_matrix; |

(2)对测试集进行分类

|  |
| --- |
| % 使用欧几里得距离分类器对测试集进行分类  classified = euclidean\_classifier(m\_hat,c,Test\_array); |

(3)计算混淆矩阵和分类准确率

|  |
| --- |
| % 计算混淆矩阵和分类准确率  confusion\_matrix = confusion\_matrix(classified,Test\_array\_response,c)  trace(confusion\_matrix)/sum(sum(confusion\_matrix)) % 准确率 |

（4）对合并数组进行分类并显示分类效果

|  |
| --- |
| % 清除临时变量  clear classified;  clear distance;  % 使用欧几里得距离分类器对合并集进行分类  classified = euclidean\_classifier(m\_hat,c,Combined\_array);  % 根据分类结果创建分类图像  Result\_Set\_Image = zeros(p,n);  for i=1:length(Combined\_array\_pos) %Creating the picture  Result\_Set\_Image(Combined\_array\_pos(i,1),Combined\_array\_pos(i,2)) = classified(1,i);  end  % 显示欧几里得距离分类器的分类结果  figure('Name','Euclidean Classifier'); imagesc(Result\_Set\_Image); |

**3.KNN分类器（K最近邻）**

基于近邻规则，测量待分类样本与训练集中样本的距离，寻找测试样本的 K 个最近邻样本。K 值是一个关键结果参数，较小的K值意味着分类结果受异常值的影响较大，而较大的K值可能会平滑分类边界，使分类结果更稳定。它决定了考虑多少个邻居来决定测试样本的类别。

根据这 K 个最近邻样本中大多数所属的类别来对测试样本进行分类。

(1)清除之前的分类结果和混淆矩阵

|  |
| --- |
| % 清除临时变量  clear confusion\_matrix;  clear classified; |

(2)计算最佳K值并进行分类

|  |
| --- |
| % 计算最佳的K值  best\_k = calculate\_best\_k(Train\_array\_response,Train\_array\_pos,c)  % 使用KNN分类器对测试集进行分类  classified = knn\_classifier(c,best\_k,Train\_array\_pos',Train\_array\_response,Test\_array\_pos');% 对测试集进行KNN分类  pr\_err = sum(classified~=Test\_array\_response)/length(Test\_array\_response); |

(3)计算混淆矩阵和分类准确率

|  |
| --- |
| % 计算混淆矩阵和分类准确率  confusion\_matrix = confusion\_matrix(classified,Test\_array\_response,c)  trace(confusion\_matrix)/sum(sum(confusion\_matrix)) % 准确率 |

（4）对合并数组进行分类并显示分类结果

|  |
| --- |
| % 清除临时变量  clear classified;  % 使用KNN分类器对合并集进行分类  classified = knn\_classifier(c,best\_k,Train\_array\_pos',Train\_array\_response,Combined\_array\_pos'); % 对合并集进行KNN分类  % 根据分类结果创建分类图像  Result\_Set\_Image = zeros(p,n);  for i=1:length(Combined\_array\_pos) %Creating the picture  Result\_Set\_Image(Combined\_array\_pos(i,1),Combined\_array\_pos(i,2)) = classified(1,i);  end  % 显示KNN分类器的分类结果  figure('Name','KNN Classifier'); imagesc(Result\_Set\_Image); |

### **创建合并集结果图像**

**1.根据训练、测试和操作集的响应向量创建合并集图像**

初始化一个全零的矩阵Result\_Set\_Image，其尺寸与原始图像相同

遍历每个集合的像素位置信息和响应向量，对应的分类结果填充到Result\_Set\_Image中。

这个过程实际上是将三个不同集合的分类结果合并到一个单一的图像中，以便进行整体的可视化。

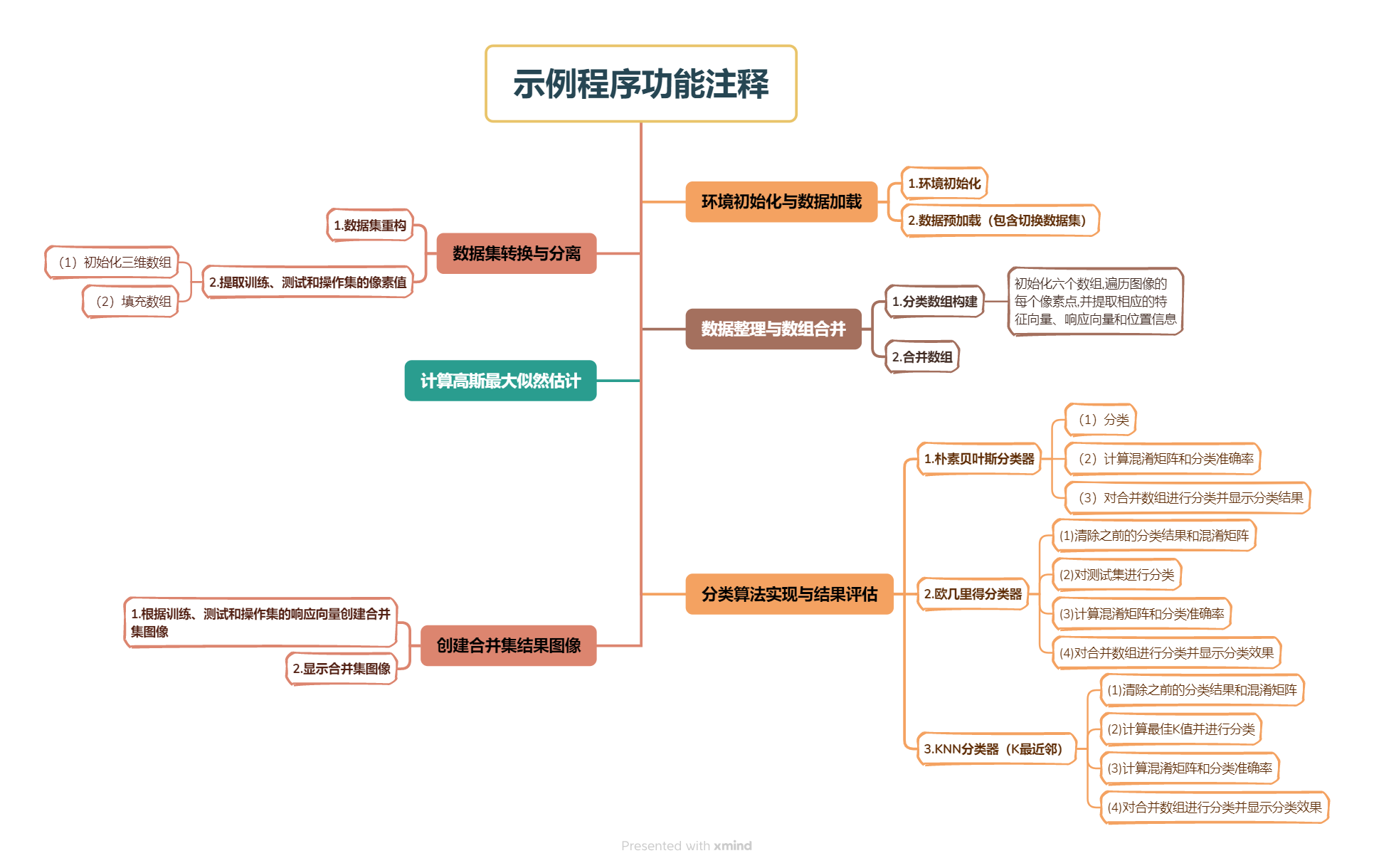
|  |
| --- |
| % 根据训练、测试和操作集的响应向量创建合并集图像  Result\_Set\_Image = zeros(p,n);%创建包含所有三个集合的结果图像  for i=1:length(Test\_array\_pos)  Result\_Set\_Image(Test\_array\_pos(i,1),Test\_array\_pos(i,2)) = Test\_array\_response(1,i);  end  for i=1:length(Train\_array\_pos)  Result\_Set\_Image(Train\_array\_pos(i,1),Train\_array\_pos(i,2)) = Train\_array\_response(1,i);  end  for i=1:length(Operational\_array\_pos)  Result\_Set\_Image(Operational\_array\_pos(i,1),Operational\_array\_pos(i,2)) = Operational\_array\_response(1,i);  end |

**2.显示合并集图像**

使用imagesc函数显示这个合并后的图像，使得用户可以直观地查看整个数据集的分类结果，包括训练集、测试集和操作集的分布和分类情况。

|  |
| --- |
| % 显示合并集图像  figure('Name','Combined Set'); imagesc(Result\_Set\_Image); |

**（七）示例程序流程图表示**

****

1. **开源数据集卫星遥感图像分类程序设计**

该程序基于github开源项目遥感图像的分类（2024睿抗机器人开发者大赛全国总决赛国一项目），环境配置tensorflow-gpu==1.12.0 、python==3.1.3、anaconda3，使用RSSCN7 DataSet 遥感图像数据集，并对程序重新进行注释与修改参量，经实际验证模型在测试集上的总体准确率为92.25%,在遥感图像分类任务上表现出色

### **(一）分类程序流程**

1.获取原始数据集

2.将数据集按8:2的比例分为训练集和验证集

3.使用DenseNet201作为基础模型进行训练

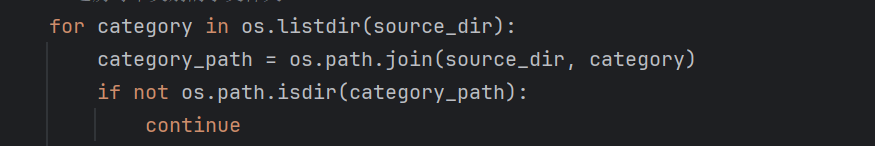
4.得到训练模型

5.利用训练得到的模型进行分类

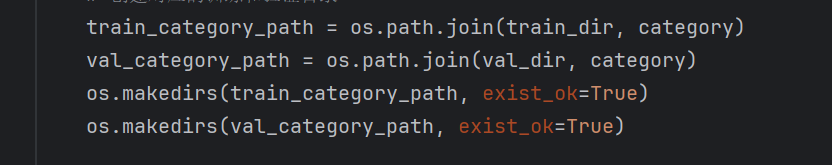
### **(二）程序解读**

### **1.split.py**

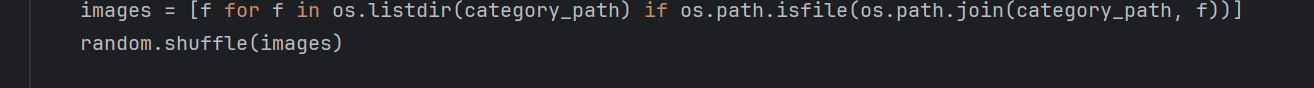
（1）首先获取所有图片



（2）然后创建存放训练集和验证集的目录



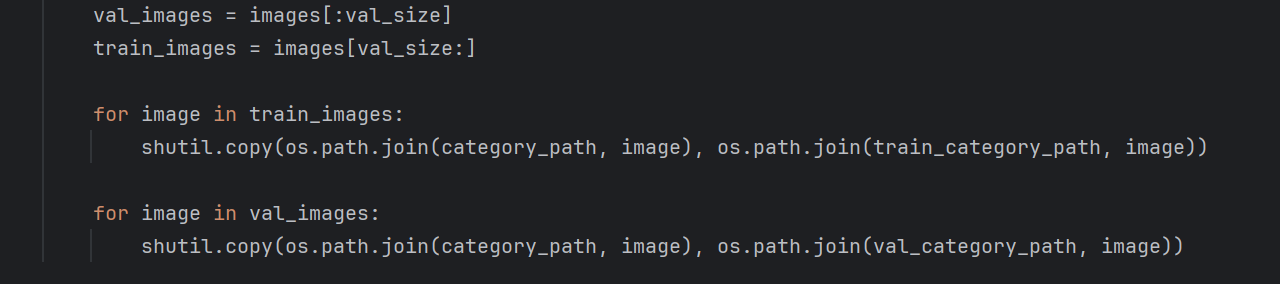
（3）打乱文件



（4）计算验证集大小

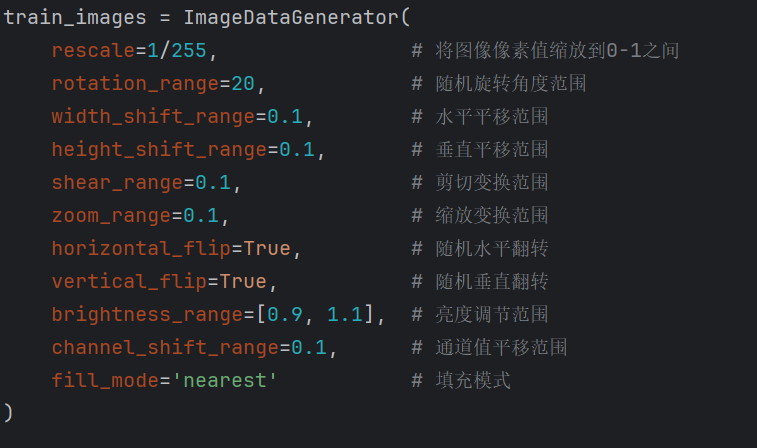


（5）选取部分图片作为验证集

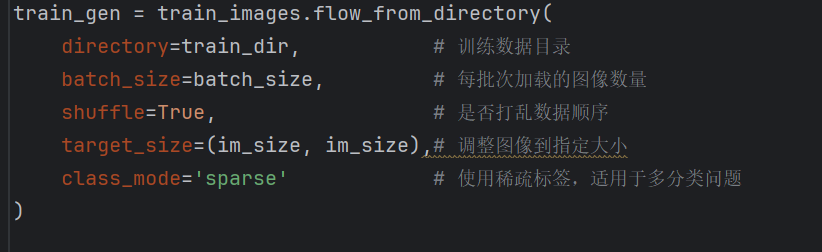


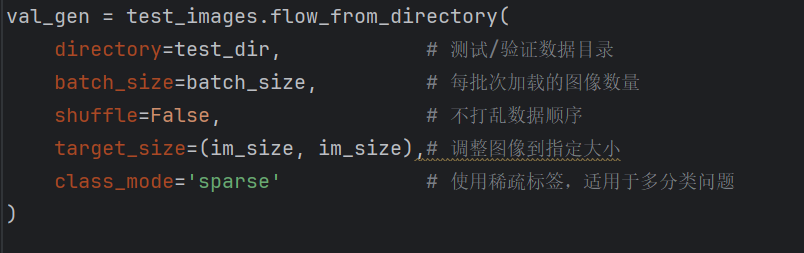
### **2.train.py**

（1）首先对训练数据进行预处理的数据生成器



（2）通过数据生成器加载数据

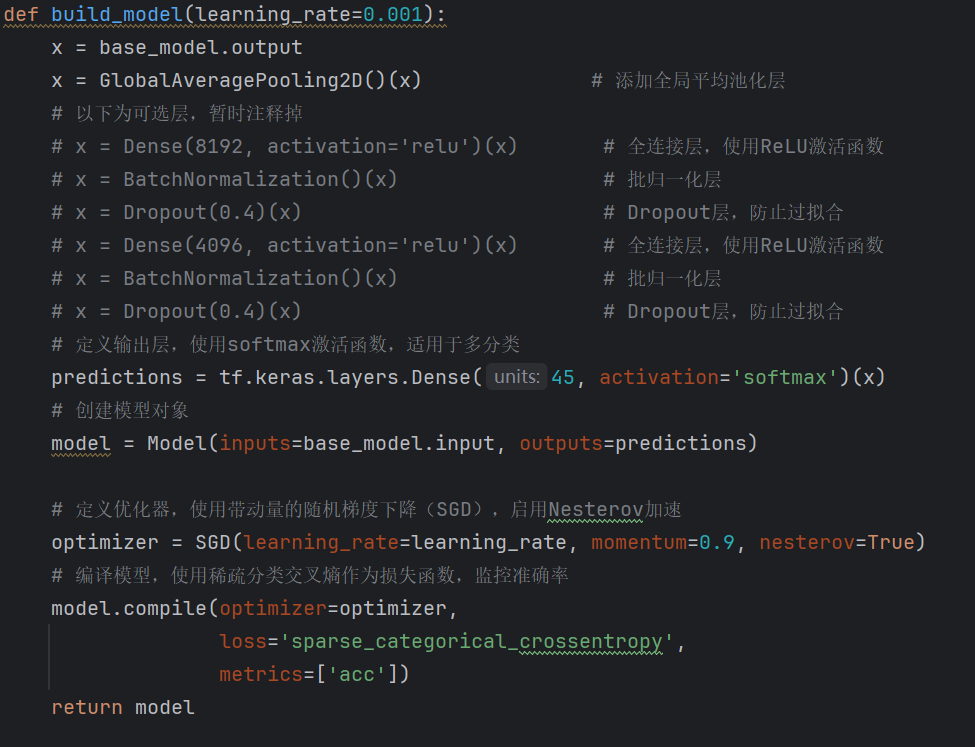




（3）使用预训练的DenseNet201作为基础模型



(4)构建自定义的模型



(4)定义早停回调，当20轮都没有提升时，结束训练

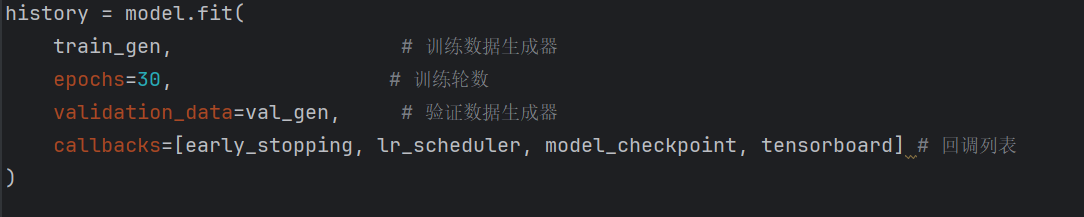


(5)定义模型检查点回调：当val\_acc提高时，保存模型权重



(6)定义模型并开始训练





(7)保存最终的模型

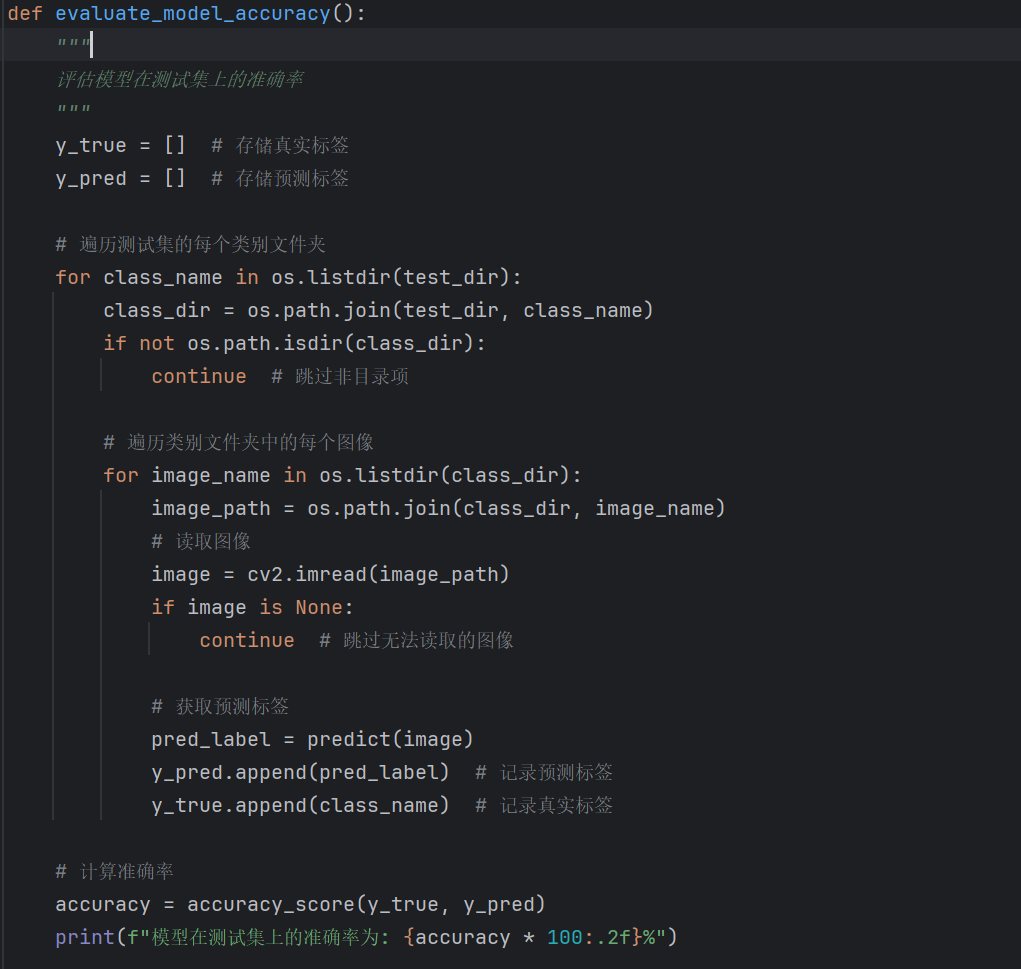


### **3.predict.py**

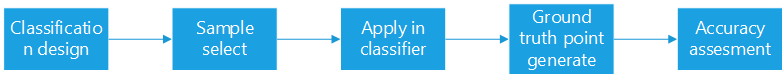
(1)定义一个预测函数，对输入图片的类型进行判断



(2)调用预测函数遍历测试集的所有图片，得到测试集的准确度



### **(三）分类程序流程图**



与遥感其他的监督分类方法一样，需要设计类别，选择样本，训练模型，最后输出训练结果。

**四、算法分析**

**（一）示例程序算法结果分析**

**1. 生成结果图像：**

在地物边界清晰的区域，各算法的分类结果较为准确，显示出良好的分类能力。

在城市边缘或混合像元区域，分类结果差异较大。例如，欧几里得分类器可能因难以处 理不典型特征而误分类，而KNN分类器通过考虑邻域信息，可能更好地处理这些区域。

朴素贝叶斯分类器在这些区域的表现取决于其特征独立性假设与实际数据的契合程度， 如果假设不成立，也可能导致分类不准确。

**2. 准确率与混淆矩阵**

**朴素贝叶斯分类器：**

准确率：计算速度快，依赖于数据集通常低于基于距离的分类器。在某些情况下， 由于其简单性或因特征独立性假设不成立，可能无法捕捉数据的复杂结构。

混淆矩阵：在某些类别上显示出较高的召回率，但精度较低，表明可能存在将其他 类别误判为该类别的情况。

**欧几里得分类器：**

准确率：通常高于朴素贝叶斯，因为它考虑了特征空间中的距离。在特征空间分布 较为规则时表现良好，但在类别重叠或样本分布不均的情况下准确率可能下降。

混淆矩阵：在某些类别上可能达到较高的精度，但如果数据集类别重叠较多或样本 分布不均，其整体召回率可能受到影响

**KNN分类器：**

准确率：在PaviaU数据集上表现最佳，这可能是因为KNN能够通过考虑邻近样本 的信息来更好地处理复杂的分类边界。但准确率与K值的选择密切相关

混淆矩阵：处理复杂区域考虑邻近样本类别信息，精度上达到较好水平

**3. 适用性与限制**

**朴素贝叶斯：**适用于特征之间相对独立的情况。在高光谱数据中，这一假设可能不成立， 导致性能受限，准确率较低。

**欧几里得分类器：**适用于特征空间中类别分布较为规则的情况。但整体性能可能受到类 别重叠或样本分布不均的影响。在处理高光谱数据时，可能需要考虑特征的物理意义和 相关性。

**KNN分类器：**适用于样本数量充足且特征空间中类别分布复杂的情况。然而，它对噪 声敏感，且计算成本较高。

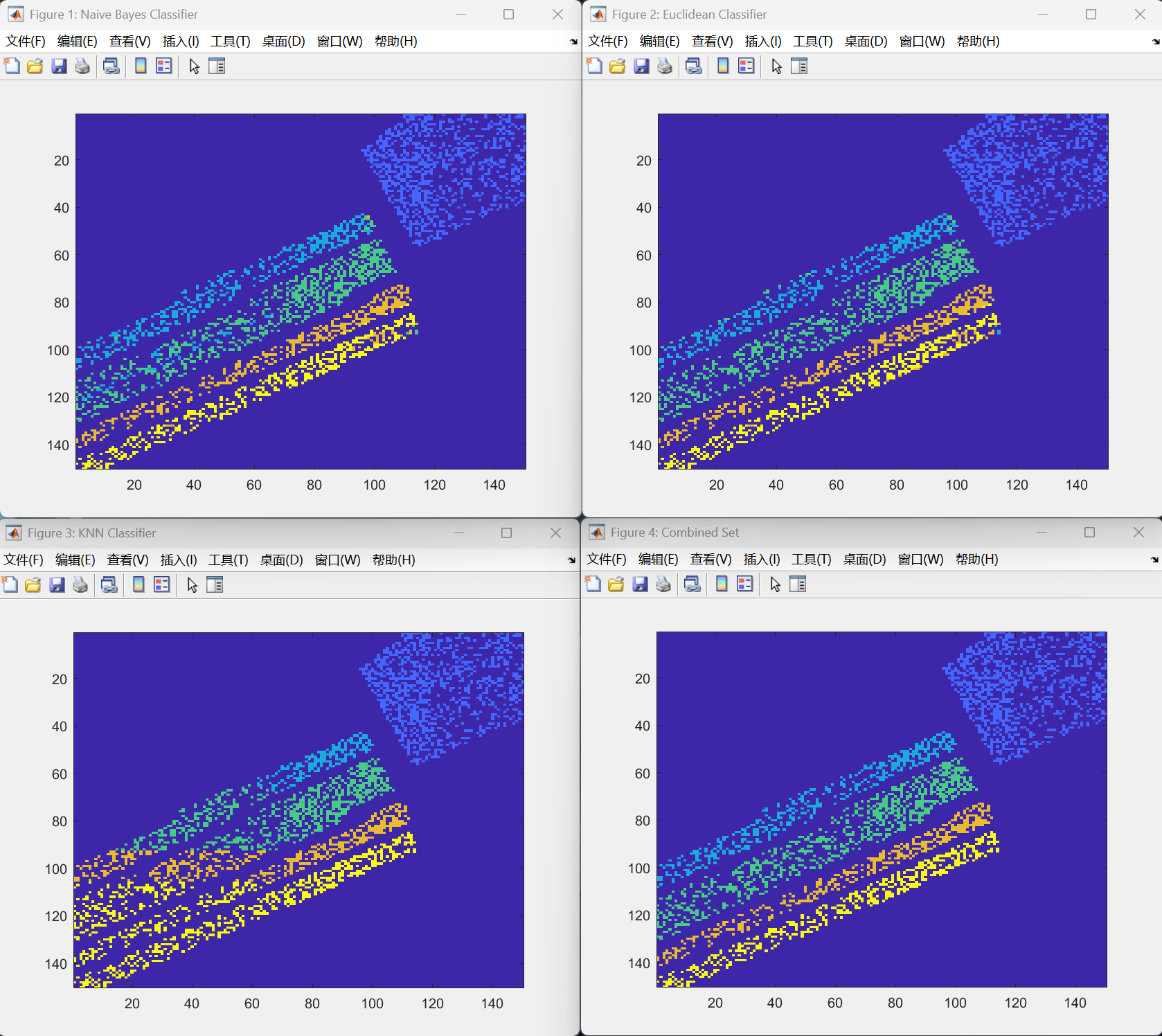
**结论：**

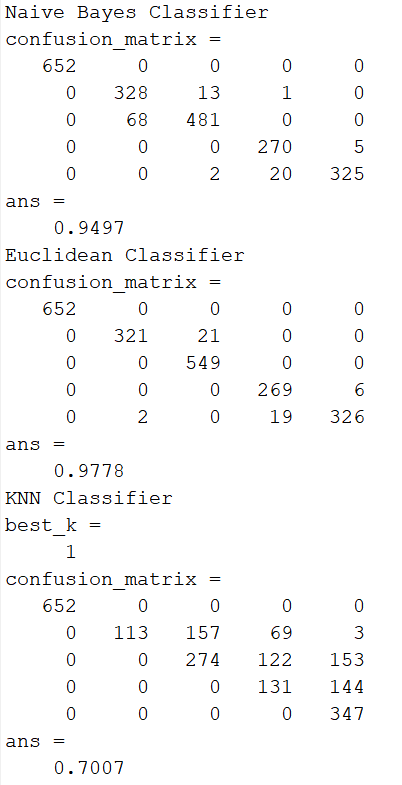
在PaviaU数据集上，KNN分类器提供了最好的性能，这可能是因为它能够更好地处理 高光谱数据的复杂性。

欧几里得分类器在某些情况下表现良好，但可能需要进一步的特征工程来提高性能。

朴素贝叶斯分类器在高光谱数据上可能不是最佳选择，除非数据特征之间相对独立。

**①对于Salinas场景下的高光谱图像数据：**





根据所给的运行结果，我们可以对三种分类器的性能进行如下分析和比较：

朴素贝叶斯分类器：

准确率：94.97%

分析：朴素贝叶斯分类器在当前数据集上的表现是相对较好的，准确率超过了94%。这表明分类器能够有效地区分不同类别的地物。尽管朴素贝叶斯分类器假设特征之间相互独立，这在实际应用中可能不总是成立，但在这个数据集上，这种假设似乎并没有显著影响分类效果。分类器可能在某些类别上存在误分类，但总体上保持了较高的分类精度。

欧几里得分类器：

准确率：97.78%

分析：欧几里得分类器的准确率最高，接近98%，这表明它在处理Salinas场景的高光谱图像数据时非常有效。这种分类器通过计算样本与各类别中心的欧几里得距离来进行分类，可能在特征空间中较好地表示了类别间的分布。欧几里得分类器在处理具有明显类别边界的数据集时表现出色，这可能是它在当前数据集上表现优异的原因。

KNN分类器：

准确率：70.07%

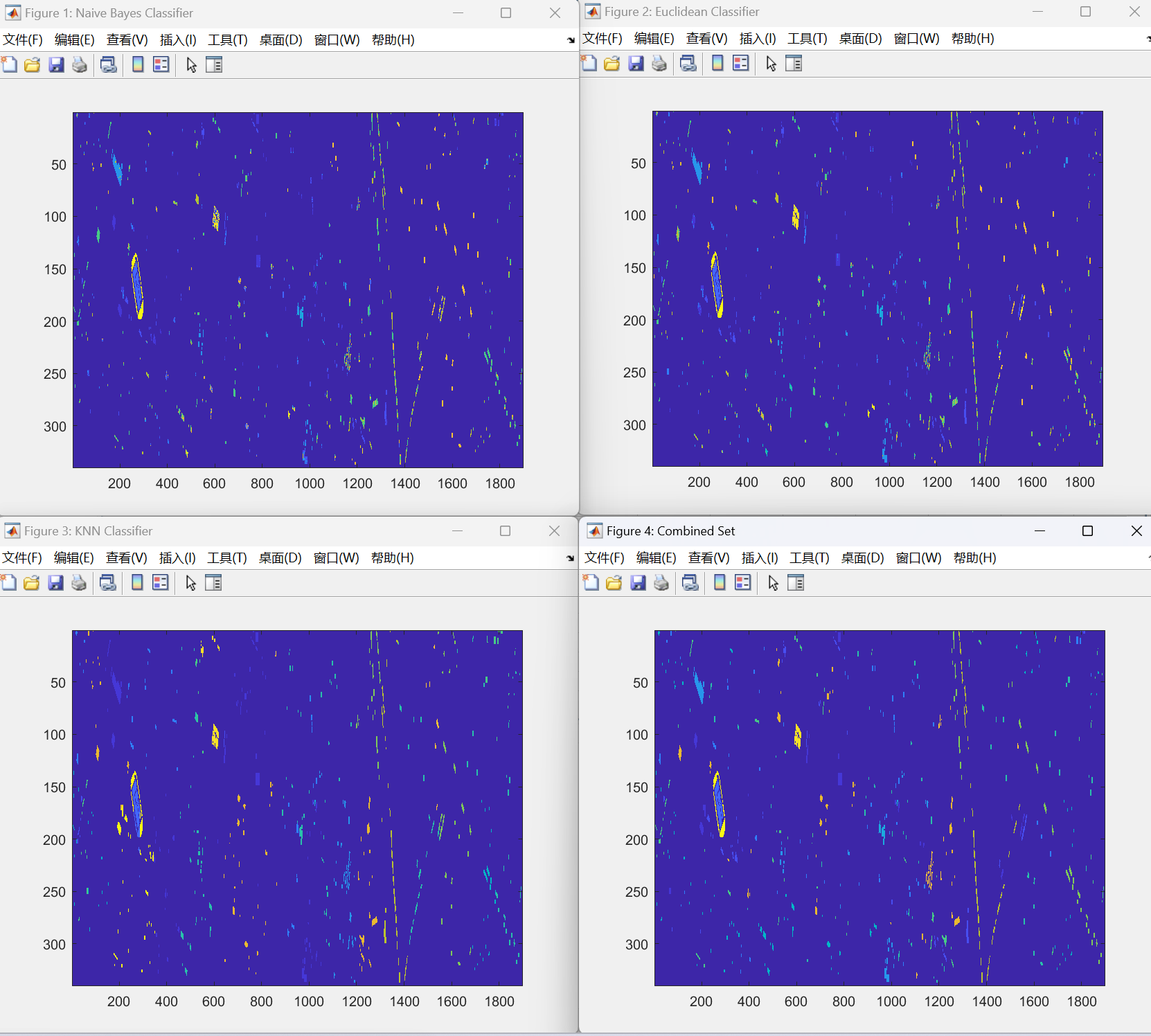
分析：KNN分类器的准确率明显低于其他两个分类器，仅为70.07%。这可能是由于KNN分类器对高维数据的敏感性较高，尤其是在选择最佳的K值时。在本例中，最佳的K值为1，这可能导致分类器对噪声和异常值过于敏感。此外，KNN分类器在计算距离时没有考虑到类别的先验概率，这可能在类别不平衡的数据集中导致性能下降。

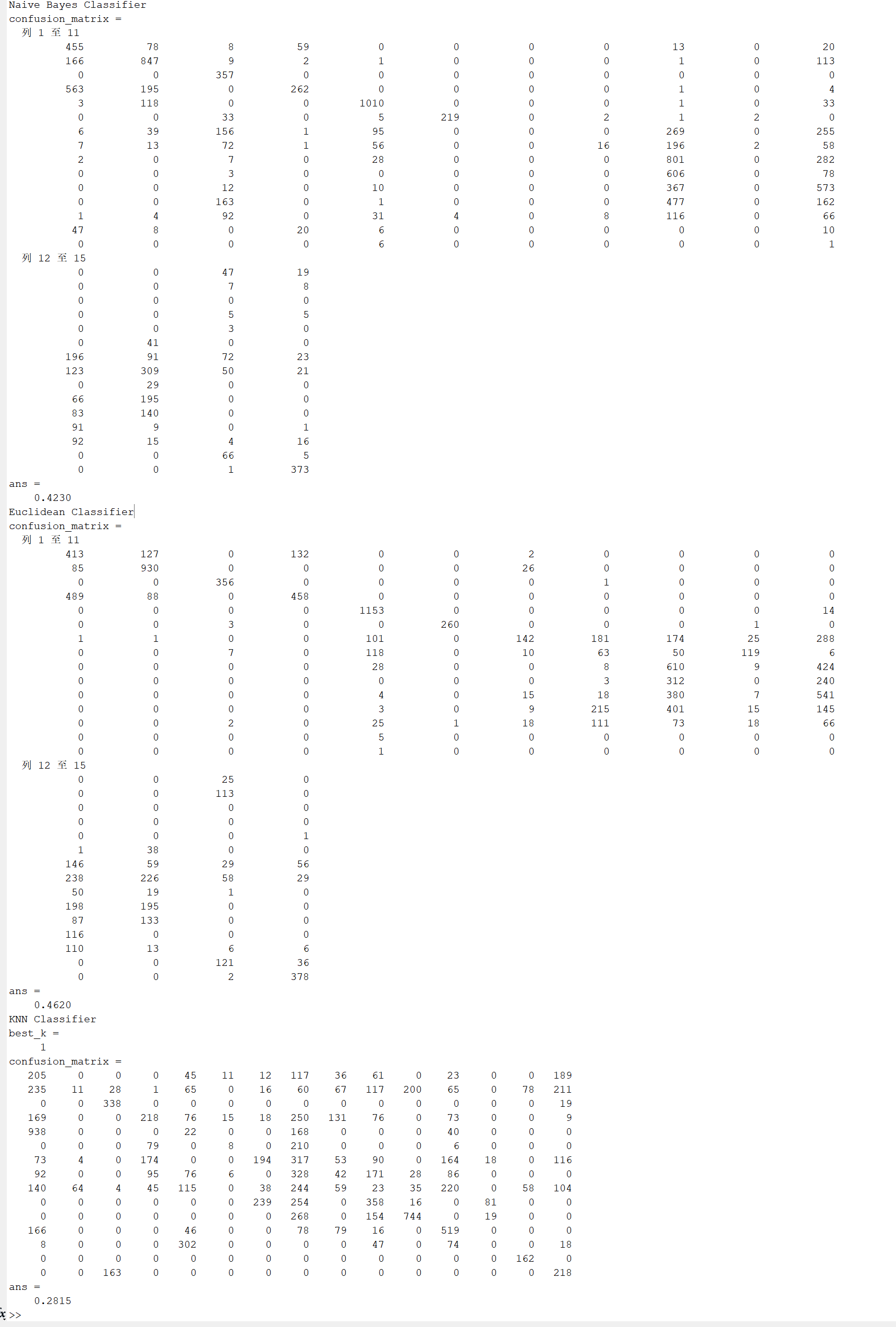
综合分析：

1.性能比较：在这组实验中，欧几里得分类器的性能最佳，其次是朴素贝叶斯分类器，而KNN分类器的性能最差。这可能与数据集的特性有关，例如类别之间的可分性、特征空间的分布等。

2.适用性：欧几里得分类器可能更适合于当前的数据分布和特征空间，因为它能够捕捉到类别间的局部结构。朴素贝叶斯分类器虽然假设特征独立，但在当前数据集上仍然表现良好。KNN分类器可能需要进一步调整K值或采用其他策略来提高其在高维数据集上的性能。

**②对于Houston场景下的高光谱图像数据集：**





对于Houston场景下的高光谱图像数据集，根据运行结果，我们可以进行以下分析：

朴素贝叶斯分类器：

准确率：0.4230（42.30%）

分析：朴素贝叶斯分类器在Houston数据集上的表现不佳，准确率仅为42.30%。这可能是因为高光谱数据的特征之间存在复杂的相关性，这与朴素贝叶斯分类器所依赖的特征独立性假设相矛盾。此外，高光谱数据的高维度特性可能导致了分类器性能的下降。

欧几里得分类器：

准确率：0.4620（46.20%）

分析：欧几里得分类器的准确率略高于朴素贝叶斯分类器，达到了46.20%。这表明在某些情况下，基于距离的分类方法可能比朴素贝叶斯的独立性假设更能捕捉到数据的空间关系。然而，提升幅度有限，可能是因为高光谱数据的高维度和复杂性，使得简单的距离度量难以有效区分类别。

KNN分类器：

准确率：0.2815（28.15%）

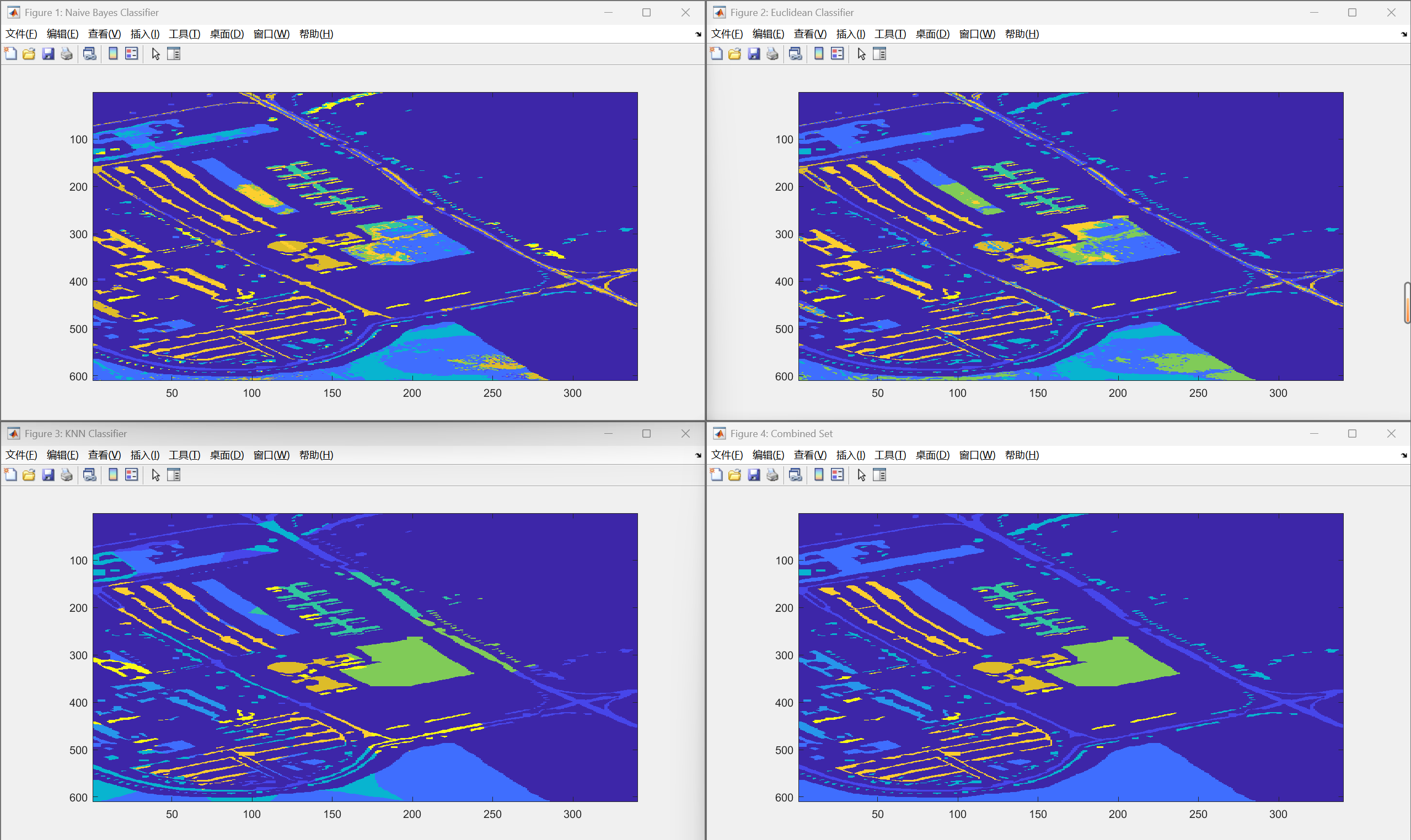
分析：KNN分类器的准确率最低，仅为28.15%。这可能是由于数据集中存在较多的噪声或异常值，影响了邻居的选择。此外，K值的选择可能也不合适，或者高维度数据导致的“维度灾难”使得KNN分类器难以有效工作。

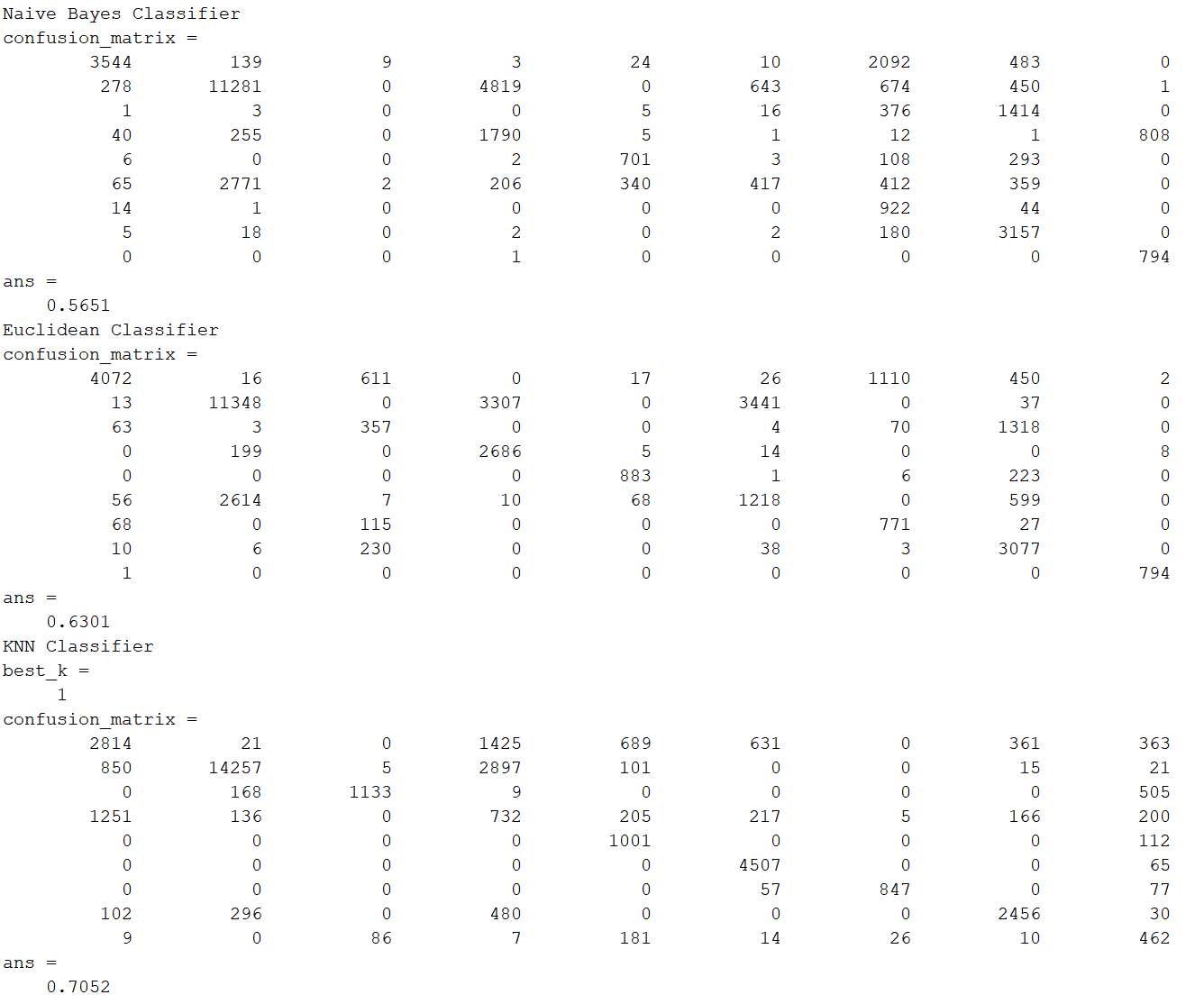
综合分析：

1.性能比较：在这三个分类器中，欧几里得分类器的性能略优于朴素贝叶斯分类器，而KNN分类器的性能最差。这可能与数据集的特性有关，如类别之间的可分性、特征空间的分布等。

2.适用性：对于Houston场景的高光谱图像数据，这些分类器可能都需要进一步的优化或选择更适合高光谱数据特性的算法。

**③对于PaviaU场景的高光谱图像数据集：**





对于PaviaU场景下的高光谱图像数据集，根据运行结果，我们可以进行以下分析：

朴素贝叶斯分类器：

准确率：56.51% 分析：朴素贝叶斯分类器在PaviaU数据集上的表现一般。尽管它是一种简单且计算效率高的分类器，但在高光谱图像数据集上，其特征独立性的假设可能不成立，导致分类效果有限。从混淆矩阵可以看出，它在某些类别上存在较多的误分类。

欧几里得分类器：

准确率：63.01% 分析：欧几里得分类器的准确率略高于朴素贝叶斯分类器，达到了63.01%。这表明基于距离的分类方法在某些情况下可能比朴素贝叶斯的独立性假设更能捕捉到数据的空间关系。然而，从混淆矩阵中可以看出，它在某些类别上仍然存在一定的误分类。

KNN分类器：

准确率：70.52% 分析：KNN分类器的准确率最高，为70.52%，是这三个分类器中表现最好的。KNN分类器通过考虑邻近样本的信息来预测未知样本的类别，这使得它在处理高光谱数据时能够更好地识别复杂的模式和边界。从混淆矩阵中可以看出，KNN分类器在大多数类别上都取得了较好的分类效果，尤其是在黄色和绿色区域的划分上。

综合分析：

1.性能比较：在这三个分类器中，KNN分类器的性能最佳，其次欧几里得分类器，朴素贝叶斯分类器的性能最差。这可能与数据集特性有关，如类别之间的可分性、特征空间的分布等。

2.适用性：对于PaviaU场景的高光谱图像数据，KNN分类器可能更适合处理这类高维度和复杂性的数据。欧几里得分类器在某些情况下也能提供合理的分类结果，而朴素贝叶斯分类器可能需要更多的特征工程或与其他方法结合使用。

总结：

分类器性能差异：在所有数据集中，朴素贝叶斯分类器的性能相对较差，准确率在42.30%到56.51%之间。这可能是由于其特征独立性的假设在高光谱数据中不成立，导致分类效果受限。欧几里得分类器在Salinas和PaviaU数据集上的表现略优于朴素贝叶斯，准确率分别达到97.78%和63.01%，但在Houston数据集上表现不佳，准确率为46.20%。这表明欧几里得分类器在某些数据集上能更好地捕捉类别间的空间关系，但并非所有情况下都有效。KNN分类器在Salinas数据集上的表现不佳，准确率为70.07%，但在PaviaU数据集上表现最好，准确率达到70.52%。KNN分类器能够更好地处理PaviaU数据集的复杂性。

数据集特性影响：不同数据集的特性对分类器的性能有显著影响。例如，Salinas数据集可能具有较为清晰的类别边界和较少的噪声,更适合欧几里得分类器，而Houston数据集可能特征分布复杂，类别间重叠较多，没有合适的分类器适用于此类型高光谱数据。

分类器选择的重要性：选择合适的分类器对于提高高光谱图像的分类性能至关重要。在实际应用中，可能需要根据数据集的具体特性和分类任务的需求来选择或定制分类器。

改进空间：

1.对于所有数据集，尤其是Houston数据集，现有的分类器性能都有很大的提升空间。可能需要考虑更复杂的算法，如集成学习、随机森林或深度学习模型，这些算法可能更适合处理高光谱数据的高维度和复杂性。

2.参数优化和特征工程：对于KNN分类器，选择合适的K值至关重要较小的K值意味着分类结果受异常值的影响较大，而较大的K值可能会平滑分类边界，使分类结果更稳定。特征选择和特征提取技术可以进一步优化，以提高所有分类器的性能。总的来说，高光谱图像数据集的分类是一个复杂的问题，需要综合考虑数据集的特性、分类器的选择和参数优化等多个方面。

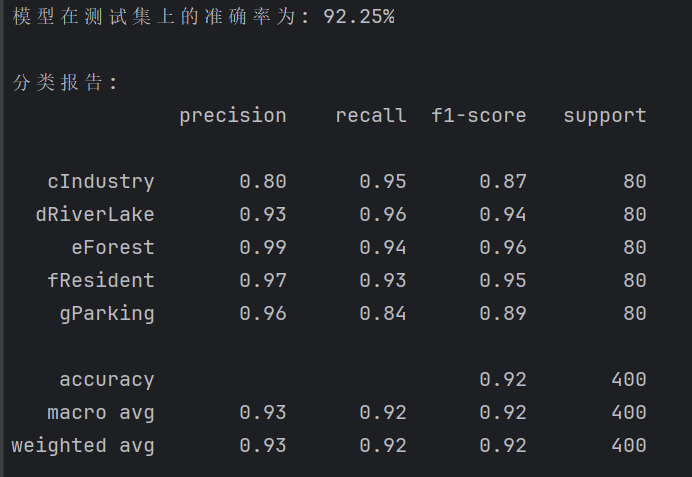
1. **开源数据集程序算法结果分析**

**数据集来源与种类：**

RSSCN7 Dataset 包含 2800 幅遥感图像，这些图像来自于 7 个典型的场景类别 —— **草地、森林、农田、停车场、住宅区、工业区和河湖**， 其中每个类别包含 400 张图像，分别基于 4 个不同的尺度进行采样。

该数据集中每张图像的像素大小为 400\*400，场景图像的多样性导致其具有较大的挑战性，这些图像来源于不同季节和天气变化，并以不同的比例进行采样。





**1.分类报告分析：**

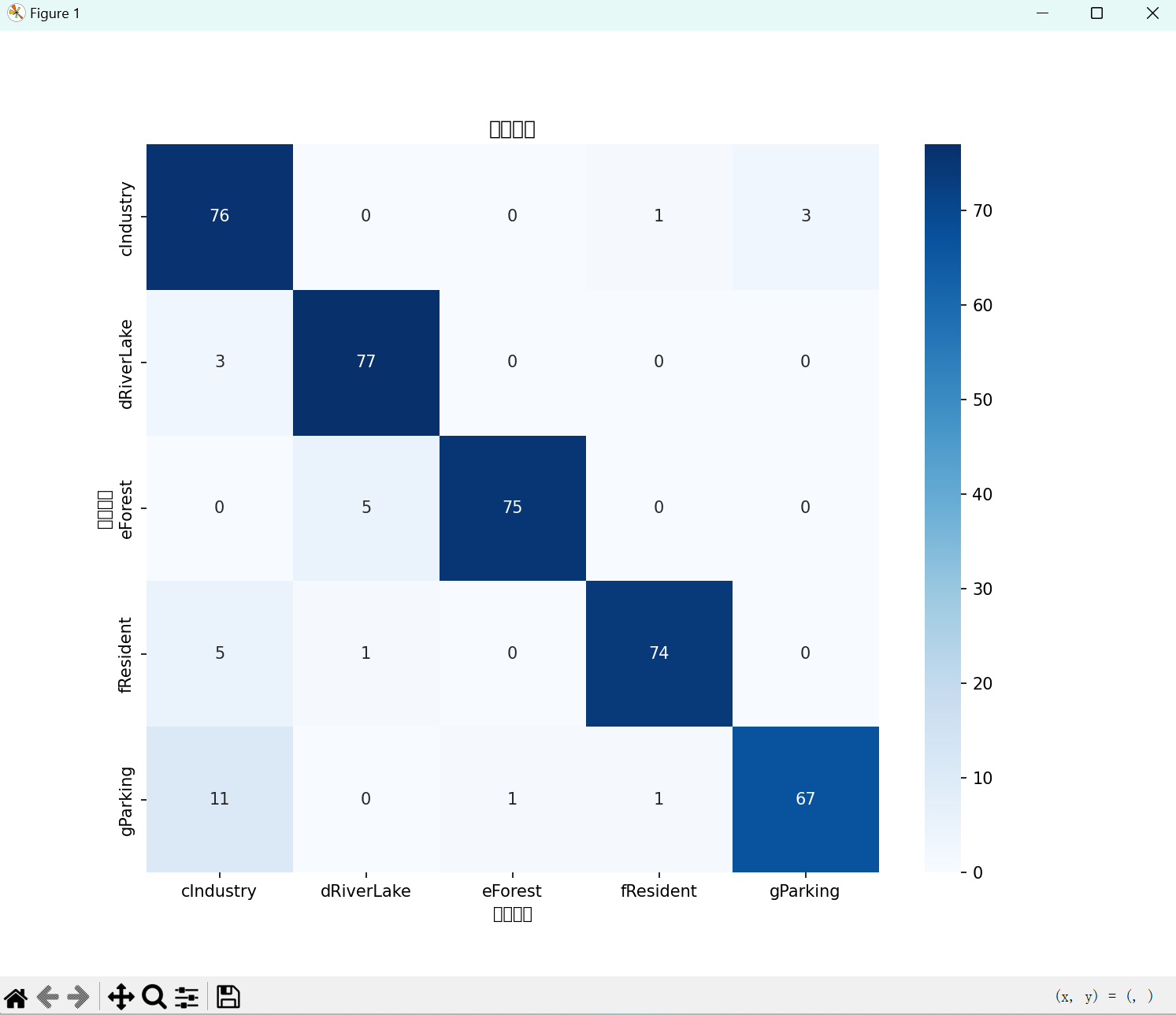
准确率 (Accuracy): 模型在测试集上的总体准确率为92.25%，这表明模型在大多数情况下能够正确分类图像。

精确率 (Precision): 精确率反映了在所有被预测为某一类别的样本中，实际属于该类别的比例。例如，eForest类别的精确率为0.99，意味着在所有被预测为森林的图像中，99%实际上是森林。

召回率 (Recall): 召回率表示在所有实际属于某一类别的样本中，被正确预测为该类别的比例。cIndustry类别的召回率为0.95，说明95%的实际工业区被正确识别。

F1分数 (F1-score): F1分数是精确率和召回率的调和平均数，用于衡量模型的准确性和召回率的平衡。dRiverLake类别的F1分数为0.94，显示了较好的分类性能。

支持 (Support): 每个类别的支持数表示测试集中该类别的样本数量。所有类别的支持数均为80，表明测试集在类别上是平衡的。



**2.混淆矩阵分析：**

混淆矩阵提供了模型预测与实际标签之间的详细对比。从矩阵中可以看出：

对角线元素: 混淆矩阵的对角线元素表示正确分类的样本数量。例如，eForest有75个样本被正确分类。

非对角线元素: 非对角线元素表示分类错误的样本数量。例如，有3个cIndustry被错误分类为gParking。

分类错误模式: 混淆矩阵显示了分类错误的模式。例如，gParking类别有11个样本被错误分类为cIndustry，这可能表明这两个类别在特征上存在一定的相似性，导致模型难以区分。

**3.模型性能评估：**

整体性能: 模型在大多数类别上表现良好，特别是在eForest和fResident类别上，精确率和召回率都很高。

类别间混淆: cIndustry和gParking之间的混淆较为明显，这可能是由于这两个类别在遥感图像中的视觉特征相似。

改进方向: 为了提高cIndustry和gParking的分类准确率，可以考虑增加更多的训练数据，或者使用更复杂的特征提取方法来区分这两个类别。

**总结：**

使用预训练的DenseNet201作为基础模型显著提高了遥感图像分类的性能。这种迁移学习的方法允许模型利用在大型数据集（如ImageNet）上学到的特征，这些特征对遥感图像分类任务也是有用的。

模型在训练集上的表现非常好，accuracy可以达到99+，val\_acc可以达到1.0，这表明模型在训练和验证集上都有很好的泛化能力。

**五、设计总结**

**六、参考文献**

[1]汪太月,明廷桥.基于矩阵变换的数字图像处理技术[J].湖北理工学院学报,2019,35(02):24-30.

[2]周法律.基于数字图像的中药植物叶片分类方法研究[D].安徽中医药大学,2016.

[3]马飞虎,曾聪,金依辰,等.一种基于属性加权朴素贝叶斯算法的OTSU图像分割方法[J].应用科学学报,2022,40(02):224-232.

[4]黄燕,杨裴裴,董富江.基于贝叶斯算法优化的多阈值图像分割仿真[J].计算机仿真,2024,41(06):223-226+312.

[5]曹兆伟.基于人工神经网络的海岛遥感影像地物分类研究[D].上海海洋大学,2016.

[6]谢凯.基于地物波谱形态参数化的高光谱图像分类方法研究[J].湖北大学学报(自然科学版),2015,37(02):116-121+142.

[7]林海晏.基于SVM遥感图像分类的参数优化研究[D].西南林业大学,2014.

[8]龚健雅.人工智能时代测绘遥感技术的发展机遇与挑战 [J]. 武汉大学学报(信息科学版),2018,43(12):1788-1796.

[9]徐逸之,姚晓婧,李祥,等.基于全卷积网络的高分辨遥感 影像目标检测[J].测绘通报,2018(1):77-82.

[10]杨必胜,梁福逊,黄荣刚.三维激光扫描点云数据处理研究进展、挑战与趋势[J].测绘学报,2017,46(10):1509 1516.

[11]齐小英,李晗语,杨海平.基于AlexNet模型的南海地图多 标签自动分类研究[J/OL].数据分析与知识发现:1-14 [2024-02-25].

[12]马旗,朱斌,张宏伟.基于VGG网络的双波段图像融合方 法[J].激光与红外,2019,49(11):1374-1380.

[13]李潍瀚,刘日阳,邵彦川,等.基于GF-1遥感图像和 ResNet50网络的PM2.5 浓度预测研究[J].环境科学导刊, 2023,42(5):82-86.

[14]赵永强,饶元,董世鹏,等.深度学习目标检测方法综述 [J].中国图象图形学报,2020,25(4):629-654.

[15]雷浩川.多分类器集成的遥感影像分类研究[D].北京: 中国地质大学(北京),2018