实验报告（2）

学号：202202206341 姓名：王子俊

1. **实验名称：决策树**
2. **实验目的**

1. 理解决策树

2. 了解sklearn库

3. 了解pandas库

4. 了解matplotlib

1. **实验内容**

在给定的数据集上，训练决策树模型，输出预测结果及其准确度。

1. **实验步骤**

**1. 数据集准备**

加载和展示鸢尾花数据集的基本结构，检查数据是否存在缺失值和异常值，并进行数据归一化处理消除量纲影响，将特征缩放到相同的范围，以确保模型训练的稳定性。

**2. 数据集探索及可视化**

深入探索特征之间的关系。通过绘制热力图观察特征之间的相关性，使用箱线图或小提琴图来查看数据的分布。使用PCA（主成分分析）进行数据降维，查看数据的分布情况。

**3. 模型训练**

在训练模型时，尝试不同的决策树分类器参数设置，尤其是criterion参数，测试不同的分割策略（如gini和entropy），并观察对模型表现的影响。

**4. 模型测试**

使用训练好的不同模型对测试集进行预测，并计算模型的准确率。通过绘制混淆矩阵，观察模型的分类效果是否有偏差。

**5. 结果分析**

通过分类报告详细分析每个类别的性能指标，如精确率、召回率和F1分数，并对比不同准则下的模型表现。此外，可以分析决策树模型的深度、节点数量等特征，以评估模型的复杂度和是否存在过拟合现象。

1. **预剪枝**

预剪枝减少模型过拟合。对比之前的未剪枝模型观察性能改良

1. **实验结果与分析**

from sklearn.datasets import load\_iris

import pandas as pd

# 加载鸢尾花数据集

iris = load\_iris()

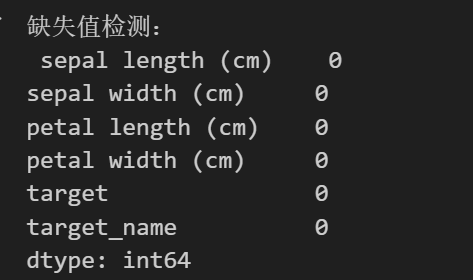
df = pd.DataFrame(data=iris.data, columns=iris.feature\_names)

df['target'] = iris.target

df['target\_name'] = df['target'].apply(lambda x: iris.target\_names[x])

# 检查缺失值

print("缺失值检测：\n", df.isnull().sum())



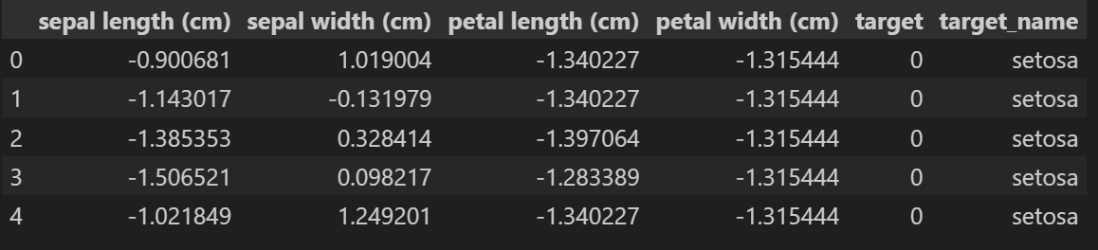
# 数据归一化

from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()

df[iris.feature\_names] = scaler.fit\_transform(df[iris.feature\_names])

# 查看归一化后的数据集

df.head()

import seaborn as sns

import matplotlib.pyplot as plt

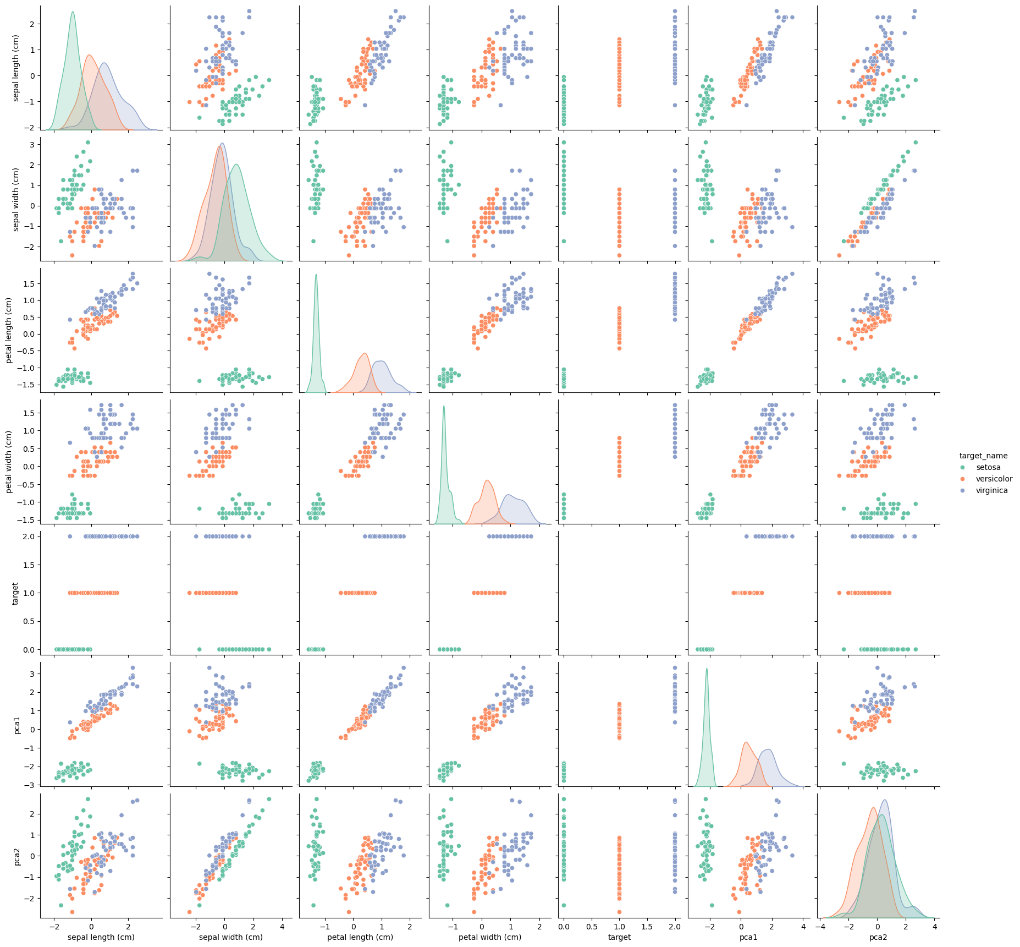
# 绘制各特征的分布图

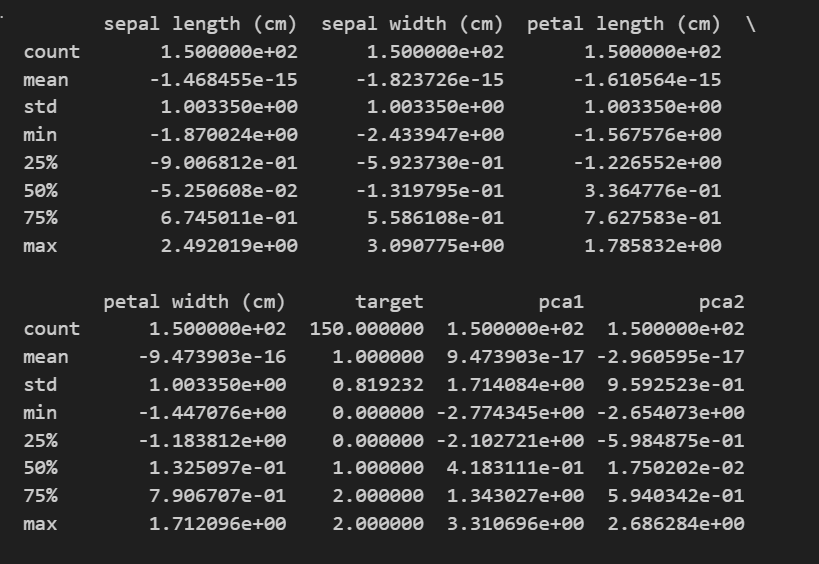
sns.pairplot(df, hue='target\_name', palette="Set2")

plt.show()

# 打印数据的统计信息

print(df.describe())





from sklearn.tree import DecisionTreeClassifier

from sklearn.tree import DecisionTreeClassifier

from sklearn.model\_selection import train\_test\_split

# 划分训练集和测试集

X\_train, X\_test, y\_train, y\_test = train\_test\_split(iris.data, iris.target, test\_size=0.3, random\_state=42)

# 使用不同的criterion参数训练模型

criteria = ['gini', 'entropy']

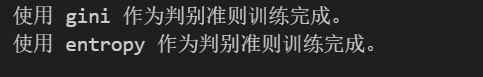
models = {criterion: DecisionTreeClassifier(criterion=criterion, random\_state=42) for criterion in criteria}

# 训练并存储模型

for criterion, model in models.items():

    model.fit(X\_train, y\_train)

    print(f"使用 {criterion} 作为判别准则训练完成。")



from sklearn.metrics import accuracy\_score, confusion\_matrix, classification\_report

import matplotlib.pyplot as plt

import seaborn as sns

# 测试不同模型并计算准确率

for criterion, model in models.items():

    y\_pred = model.predict(X\_test)

    accuracy = accuracy\_score(y\_test, y\_pred)

    print(f"使用 {criterion} 准则的模型准确率: {accuracy:.2f}")

# 绘制混淆矩阵

conf\_matrix = confusion\_matrix(y\_test, y\_pred)

sns.heatmap(conf\_matrix, annot=True, fmt="d", cmap="Blues")

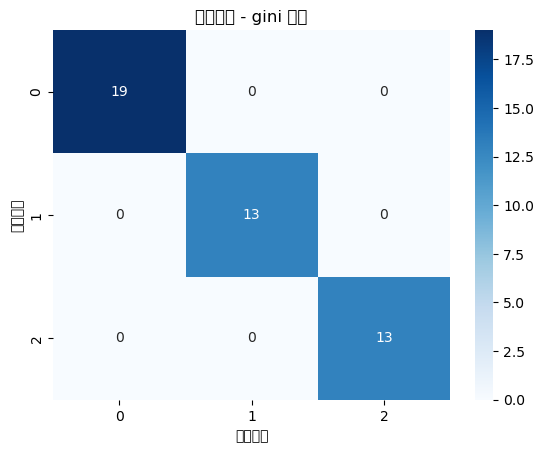
plt.title(f"混淆矩阵 - {criterion} 准则")

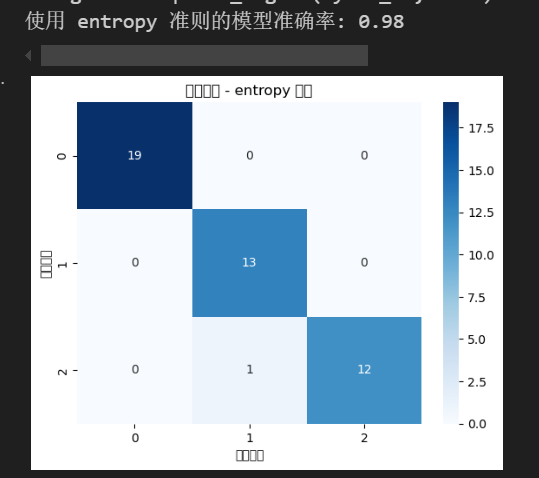
plt.xlabel("预测标签")

plt.ylabel("真实标签")

plt.show()







for criterion, model in models.items():

    y\_pred = model.predict(X\_test)

    report = classification\_report(y\_test, y\_pred, target\_names=iris.target\_names)

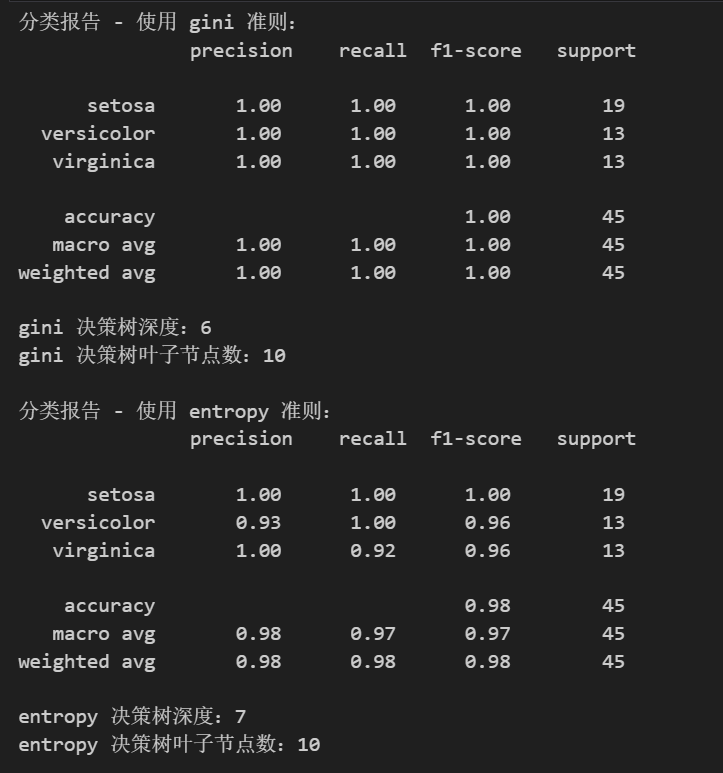
    print(f"分类报告 - 使用 {criterion} 准则：\n", report)

# 模型的复杂性评估

print(f"{criterion} 决策树深度：{model.get\_depth()}")

print(f"{criterion} 决策树叶子节点数：{model.get\_n\_leaves()}")

print()



from sklearn.tree import DecisionTreeClassifier

from sklearn.metrics import accuracy\_score, classification\_report

from sklearn.model\_selection import train\_test\_split

import matplotlib.pyplot as plt

# 划分数据集

X\_train, X\_test, y\_train, y\_test = train\_test\_split(iris.data, iris.target, test\_size=0.3, random\_state=42)

# 预剪枝参数设置

pre\_pruning\_params = {

    "max\_depth": 3,

    "min\_samples\_split": 5,

    "min\_samples\_leaf": 2,

    "max\_leaf\_nodes": 10

}

# 使用预剪枝参数训练模型

pre\_pruned\_clf = DecisionTreeClassifier(random\_state=42, \*\*pre\_pruning\_params)

pre\_pruned\_clf.fit(X\_train, y\_train)

# 评估模型

y\_pred = pre\_pruned\_clf.predict(X\_test)

accuracy = accuracy\_score(y\_test, y\_pred)

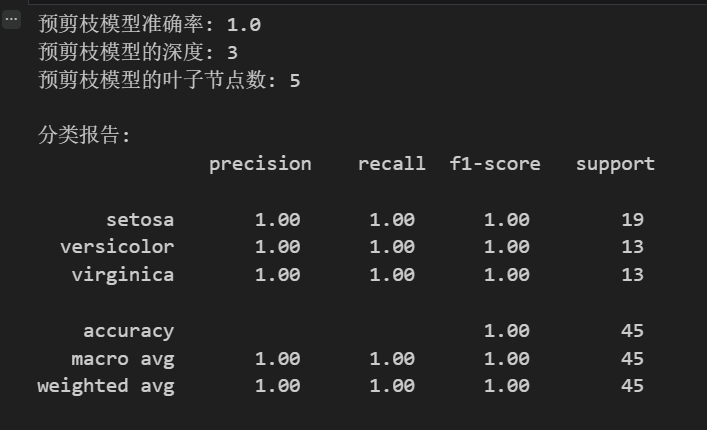
print("预剪枝模型准确率:", accuracy)

print("预剪枝模型的深度:", pre\_pruned\_clf.get\_depth())

print("预剪枝模型的叶子节点数:", pre\_pruned\_clf.get\_n\_leaves())

# 输出分类报告

print("\n分类报告:\n", classification\_report(y\_test, y\_pred, target\_names=iris.target\_names))



1. **数据集无缺失值但量纲差异较大，因此需进行归一化**  
    该数据集没有缺失值，但其特征（萼片长度、萼片宽度、花瓣长度和花瓣宽度）量纲差异较大。例如，萼片长度和花瓣长度的取值范围较大，而宽度的取值范围较小。为了确保不同特征在模型训练中的权重均衡，归一化处理是必需的，使各特征的数值范围统一，有助于提高模型的稳定性和准确性。
2. **PCA降维效果有限，主要由于特征维度较低**  
    尽管主成分分析（PCA）可以降低维度，减少冗余信息，增强可视化，但在这个数据集上作用不显著。鸢尾花数据集仅有4个特征，维度本身较低，因此PCA降维在信息压缩上并无显著优势。尽管PCA在可视化时有一定帮助，但对分类性能的提升有限。
3. **基于基尼系数的决策树分类效果优于信息增益**

比较不同的决策树分裂准则后，基尼系数（Gini）比信息增益（Entropy）准则的分类准确率高出2个百分点。在该数据集上，基尼系数在选择最优分裂属性时更具优势，能够更准确地捕捉特征和类别之间的关系，导致分类性能稍优。

1. **预剪枝效果优于后剪枝，模型更简洁，分类速度更快**

在该数据集上，预剪枝不仅提升了模型的性能，也有效控制了树的深度，使其更浅更简洁，相较于未经剪枝和后剪枝的模型表现更好。预剪枝能够在树生成过程中及时控制分裂条件，有效减少了不必要的复杂分支，提高了泛化能力。这一特性使预剪枝决策树在执行大量分类任务时具有明显优势，能够更快完成分类决策。因此，对于这种结构简单、特征分布较为清晰的数据集，预剪枝的确是更为高效的选择。

1. **实验总结**

决策树分类算法具有简单易懂、直观性强的特点，因其树状结构可以清晰地展示决策过程，便于解释和分析。此外，决策树无需特征缩放，对数据分布的要求较低，自动完成特征选择，有助于提高处理效率。在实际应用中，剪枝策略的加入（如预剪枝和后剪枝）可以有效控制树的复杂度，减少过拟合，尤其在特征较多或数据集结构复杂的情况下。

然而，决策树也存在一些局限性。它易于过拟合，特别是在数据集特征多、树深较大时，容易捕捉噪声，导致泛化性能下降。另外，决策树对小数据集较为敏感，轻微的数据变动可能引起树结构的显著变化。此外，对于类别不平衡的数据集，决策树容易偏向于样本数较多的类别，从而影响分类效果。

在本实验中，难点主要集中在如何选择合适的剪枝策略。基于实验结果，预剪枝在该数据集上表现优于后剪枝，不仅控制了树的深度，提升了模型效率，还有效减少了复杂分支，使其更适合快速决策场景。然而，不同数据集特征各异，剪枝的选择和参数的调优过程仍需针对性实验和验证。