数据挖掘上机报告（实验一）

1. **实验内容**

本实验利用鸢尾花数据集（Iris Dataset），通过填补缺失数据和划分训练集与测试集，分别应用决策树（Decision Tree）、K 最近邻（KNN）和朴素贝叶斯（Naive Bayes）分类算法，对鸢尾花种类进行分类，并比较不同算法在测试集上的预测准确率。同时对决策树进行可视化，并输出各属性的重要性。

2. **分析及设计**

鸢尾花数据集包含 150 个样本，每个样本有 4 个特征（花萼长度、花萼宽度、花瓣长度、花瓣宽度）以及目标变量（鸢尾花的三个类别：Setosa、Versicolor、Virginica）。本问题是一个多分类问题。

**数据预处理**：

填补缺失值（如有缺失数据）。

数据归一化或标准化（适用于 KNN）。

将数据划分为训练集和测试集。

**分类算法实现**：

决策树：构建并训练分类树，利用测试集评估准确率。

KNN：设置适当的 K 值，训练模型并评估准确率。

朴素贝叶斯：基于特征的条件概率进行分类。

1. **详细实现**

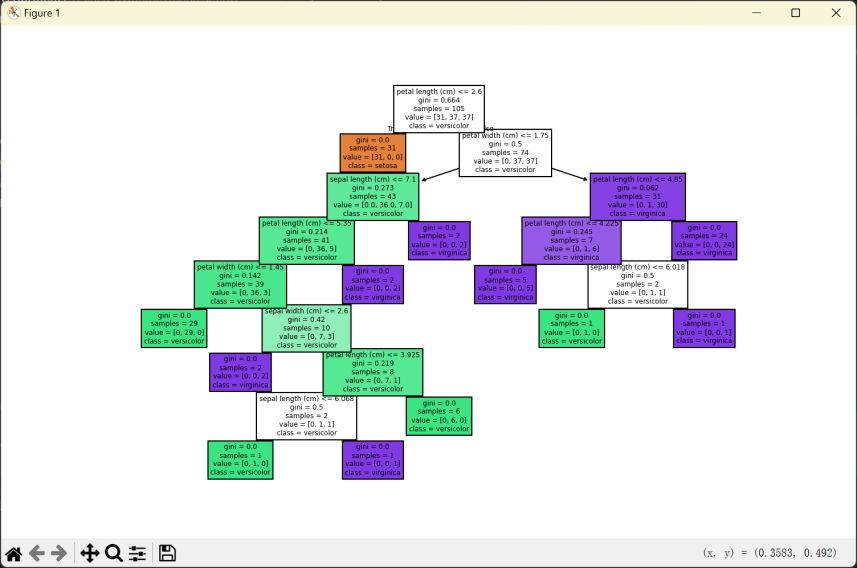
**数据预处理**

import numpy as np  
from sklearn.datasets import load\_iris  
from sklearn.model\_selection import train\_test\_split  
from sklearn.impute import SimpleImputer  
from sklearn.tree import DecisionTreeClassifier, plot\_tree  
from sklearn.neighbors import KNeighborsClassifier  
from sklearn.naive\_bayes import GaussianNB  
from sklearn.metrics import accuracy\_score  
import matplotlib.pyplot as plt  
  
# 1. 加载数据集  
iris = load\_iris()  
X, y = iris.data, iris.target  
# 2. 引入缺失值并填补  
rng = np.random.default\_rng(42)  
missing\_mask = rng.choice([1, 0], size=X.shape, p=[0.1, 0.9]).astype(bool)  
X[missing\_mask] = np.nan  
imputer = SimpleImputer(strategy='mean')  
X\_imputed = imputer.fit\_transform(X)  
  
# 3. 划分训练集和测试集  
X\_train, X\_test, y\_train, y\_test = train\_test\_split(X\_imputed, y, test\_size=0.3, random\_state=42)

**分类：**

#决策树分类dt\_model = DecisionTreeClassifier(random\_state=42)  
dt\_model.fit(X\_train, y\_train)  
y\_pred\_dt = dt\_model.predict(X\_test)  
accuracy\_dt = accuracy\_score(y\_test, y\_pred\_dt)  
print(f"决策树准确率: {accuracy\_dt:.2f}")  
# 决策树可视化  
plt.figure(figsize=(10, 6))  
plot\_tree(dt\_model, feature\_names=iris.feature\_names, class\_names=iris.target\_names, filled=True)  
plt.show()  
# 输出特征重要性  
print("特征重要性:", dt\_model.feature\_importances\_)  
# 5. KNN分类  
knn\_model = KNeighborsClassifier(n\_neighbors=5)  
knn\_model.fit(X\_train, y\_train)  
y\_pred\_knn = knn\_model.predict(X\_test)  
accuracy\_knn = accuracy\_score(y\_test, y\_pred\_knn)  
print(f"KNN准确率: {accuracy\_knn:.2f}")  
# 6. 朴素贝叶斯分类  
nb\_model = GaussianNB()  
nb\_model.fit(X\_train, y\_train)  
y\_pred\_nb = nb\_model.predict(X\_test)  
accuracy\_nb = accuracy\_score(y\_test, y\_pred\_nb)  
print(f"朴素贝叶斯准确率: {accuracy\_nb:.2f}")

1. **实验结果**

****



结果分析：

决策树的解释性较强，分类规则清晰，但可能存在过拟合。

KNN 对距离敏感，适合归一化后的数据。

朴素贝叶斯对数据分布假设较强，但表现稳定。

**5.心得体会**

在完成本次实验的过程中，我对分类算法有了更深刻的理解和实际应用的体验，以下是一些体会和反思：

以前在理论学习中了解了决策树、KNN 和朴素贝叶斯的基本原理，但在实验中通过实际动手实现并测试这些模型，对它们的优缺点有了更加直观的感受。例如，决策树的模型解释性强，但容易过拟合；KNN 对距离度量敏感且计算成本较高；朴素贝叶斯虽然简单，但在数据分布满足条件时效果出色。

实验中需要处理缺失值以及特征的标准化，这让我意识到在实际的机器学习任务中，数据预处理往往是模型性能的关键环节之一。例如，KNN 对于未标准化数据的分类性能可能会受到显著影响，而填补缺失值对于模型的鲁棒性至关重要。

在同一数据集上使用多种模型进行对比分析，能更清晰地了解不同模型的适用场景和性能差异。例如，对于鸢尾花数据集这样的简单分类问题，三种算法的准确率都较高，但随着数据特性和规模的变化，这种比较可能会变得更加复杂和有趣。

决策树的可视化让我深刻感受到模型的可解释性，它不仅可以显示分类的规则，还能够直观展示特征的重要性。这对于实际业务应用中向非技术人员解释模型结果非常有帮助。