实验 随机森林

实验目的

1. 使用pandas和seaborn对数据进行可视化
2. 实现随机森林算法
3. 掌握sklearn中的随机森林分类器的调用方法
4. 了解sklearn中的PCA使用方法并对结果进行可视化
5. 使用随机森林解决鸢尾花分类问题

实验数据

本实验使用的是 `sklearn` 提供的乳腺癌数据集 `load_breast_cancer`。该数据集包含569个样本，每个样本有30个特征，标签为二分类，分别表示良性和恶性肿瘤。

实验步骤

1.数据读取与可视化

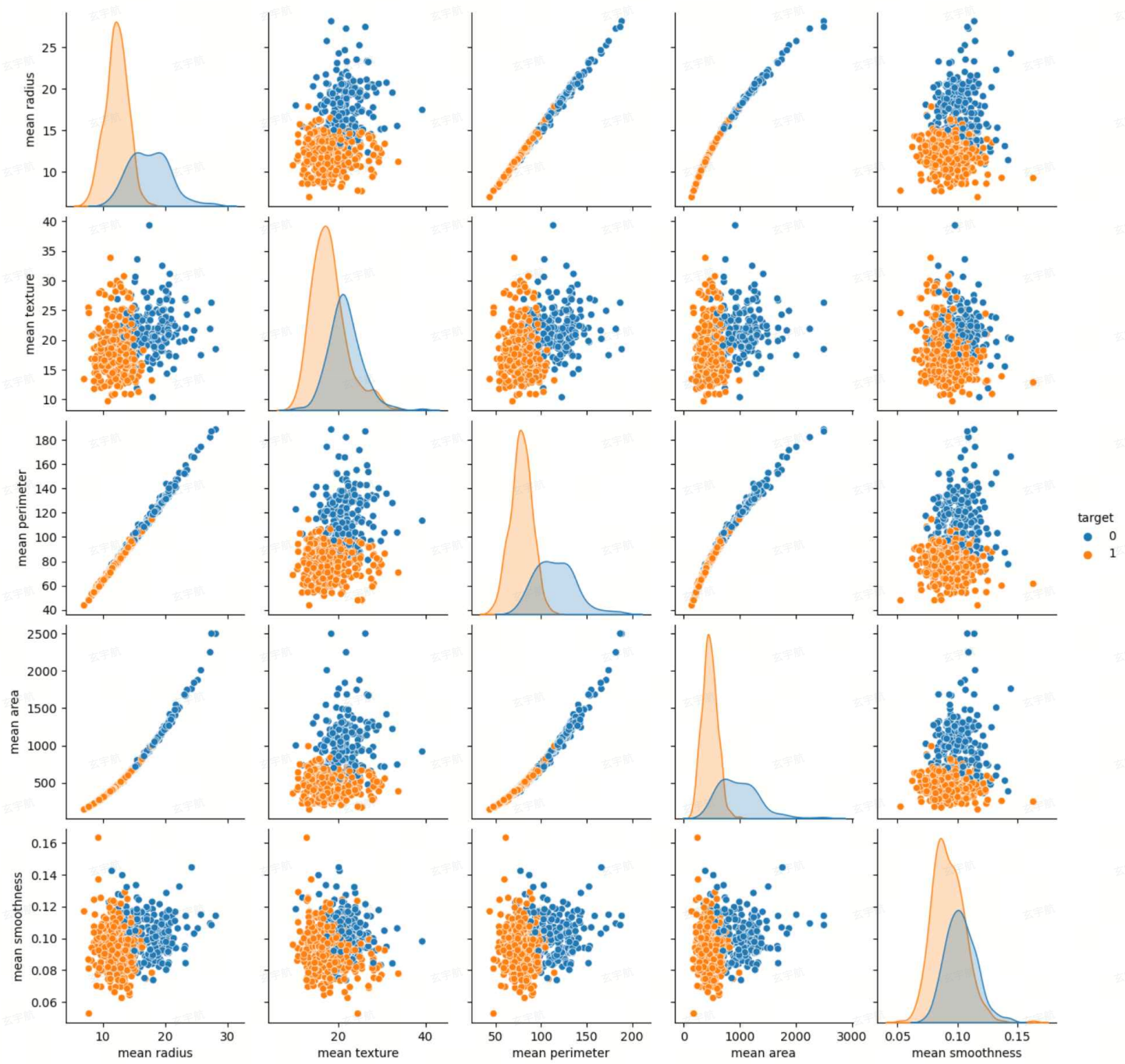
在这一部分，我们将读取乳腺癌数据集，并使用Pandas、Matplotlib和Seaborn库进行初步的数据探索和可视化。

- 通过 `load_breast_cancer()` 函数读取乳腺癌数据集，并将其转换为Pandas DataFrame格式，方便进一步处理。
- 使用 `seaborn` 的 `pairplot` 函数对前5个特征进行可视化，查看特征之间的关系以及它们与目标变量的关系。
- 使用 `train_test_split()` 进行数据划分。

```
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd
from sklearn.datasets import load_breast_cancer
from sklearn.model_selection import train_test_split
# 读取数据集
cancer = load_breast_cancer()
X = cancer.data
y = cancer.target
```

```
df = pd.DataFrame(X, columns=cancer.feature_names)
df['target'] = y

# 数据可视化
sns.pairplot(df, hue='target', vars=df.columns[:5])
plt.show()
# 划分训练集和测试集
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)
```



2.实现随机森林

在这一部分，我们将从零实现一个简单的随机森林分类器，主要包括模型的训练和预测函数。需要补全的代码用 `code here` 标注。

```python
def fit(self, X, y):
        n_samples, n_features = X.shape
        for _ in range(self.n_estimators):
            tree = DecisionTreeClassifier(max_depth=self.max_depth)
            # 抽取样本
            indices = np.random.choice(n_samples, n_samples, replace=True)
            '''

            code here

            '''
            # 随机选择特征的最大数量确定方式
            if self.max_features == 'sqrt':
                max_features = int(np.sqrt(n_features))
            elif self.max_features == 'log2':
                max_features = int(np.log2(n_features))
            else:
                max_features = n_features
            '''

            code here

            '''
            self.feature_indices.append(feature_idx)
            '''

            code here

            '''
            self.trees.append(tree)
```

- `fit(self,X,y)` 函数用于随机森林分类器的训练。该部分首先调用了sklearn中的决策树类 `DecisionTreeClassifier` ，该类实例化后的对象tree可以调用 `tree.fit(x,y)` 进行训练。随机森林算法要随机选择m个样本用于训练基分类器，训练过程中要随机选择k个属性并通过计算信息增益等确定最优的属性选择。本次实验中我们只需要随机选择好样本和属性，剩下的交给sklearn调用 `fit()` 即可。
  a. 在第一个 `code here` 中你需要完成m个样本的随机选择，并把选择好的m个样本放在 `bootstrap_X` 中，对应标签放在 `bootstrap_y` 中。
  b. 第二个 `code here` 中你选完成 `max_features` 个特征的随机选取，将索引存放在 `feature_idx` 中。

c. 第三个 `code here` 中把 `bootstrap_X` 中对应的特征抽取作为 `x` 参数传入 `tree.fit(x,y)` 。注意这里属性 `feature_indices` 存储了每一棵树的特征选择索引（列表的第i个元素为第i棵决策树选择的特征索引），用于后续的预测过程。

```
def predict(self, X):
        predictions = np.zeros((X.shape[0], self.n_estimators))
        for i, tree in enumerate(self.trees):
            feature_idx = self.feature_indices[i]
            '''

            code here

            '''
        # 投票取多数票
        majority_vote = np.apply_along_axis(lambda x:
np.bincount(x.astype(int)).argmax(), axis=1, arr=predictions)
        return majority_vote
```

- `predict(self,X)` 函数实现随机森林的预测功能，具体过程就是所有的决策树给出对样本的预测结果，最终取多数票的结果作为森林的预测值。在 `code here` 部分计算第i棵决策树对样本的预测值（注意每棵树只会看自己训练时看到的特征，索引存在 `feature_idx` 中）存在 `predictions` 中。

3.调用sklearn中的随机森林分类器

scikit-learn中的随机森林分类器 `RandomForestClassifier` 的使用方法与我们之前接触的相同，首先对分类器实例化，然后就可以使用 `fit()` 、 `predict()` 进行训练和预测。现在你需要在 `code here` 中用我们的实验数据进行训练和预测。

使用举例：

```
clf = RandomForestClassifier(n_estimators=10,
                             max_features='sqrt',
                             max_depth=None,
                             random_state=42)
clf.fit(X_train, y_train)
y_pred = clf.predict(X_train)
```

```
'''

```

```
code here

'''
accuracy_sklearn_train = accuracy_score(y_train, y_pred_sklearn_train)
accuracy_sklearn_test = accuracy_score(y_test, y_pred_sklearn_test)
print(f'Accuracy of sklearn Random Forest on training set: {accuracy_sklearn_train:.4f}')
print(f'Accuracy of sklearn Random Forest on test set: {accuracy_sklearn_test:.4f}')
```

4.PCA与实验结果可视化

PCA（主成分分析，Principal Component Analysis）是一种常用的数据降维技术，通过线性变换将数据从高维空间投影到低维空间，使得投影后的数据在新坐标轴上的方差最大。这些新坐标轴称为主成分。通过这种方式，PCA 可以有效减少数据的维度，同时保留数据的主要特征。

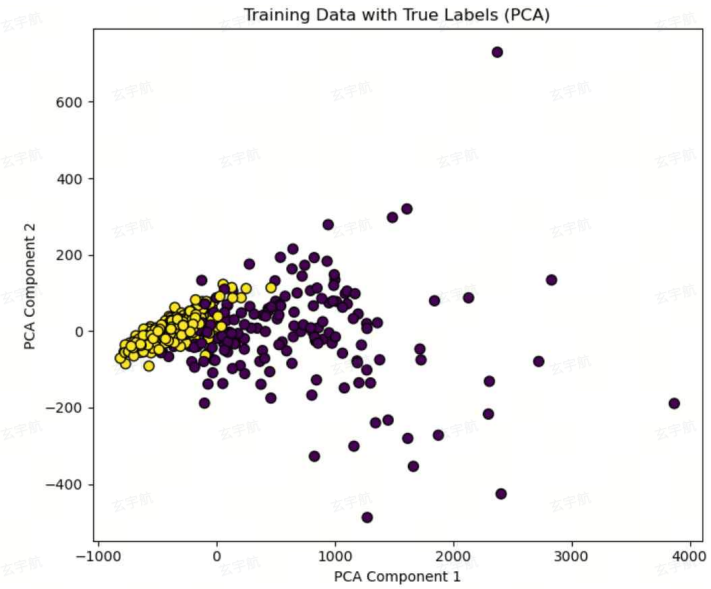
我们主要使用 `sklearn.decomposition` 模块中的 `PCA` 类来实现 PCA。下面是一个基本的流程：

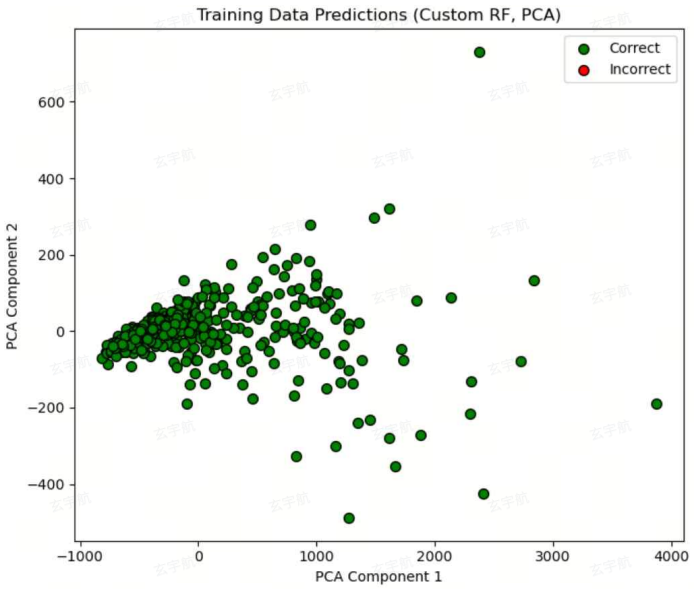
1. 导入库
2. 实例化 PCA 对象
3. 拟合和转换数据

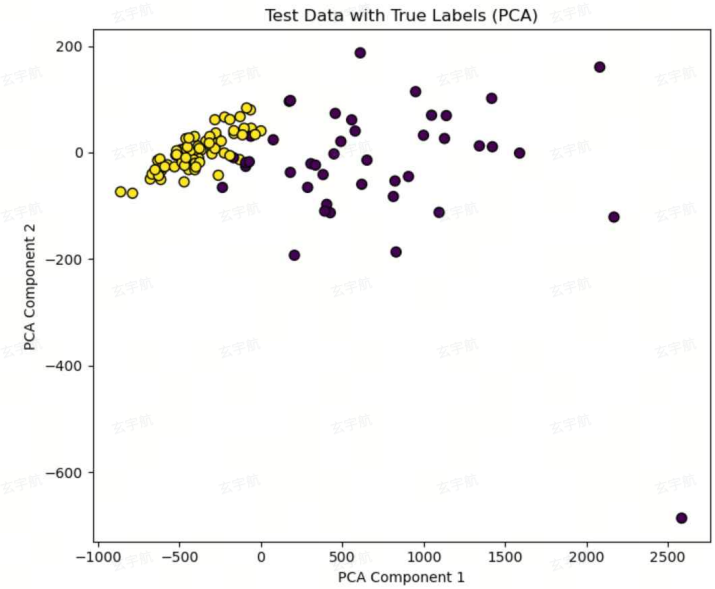
使用举例：

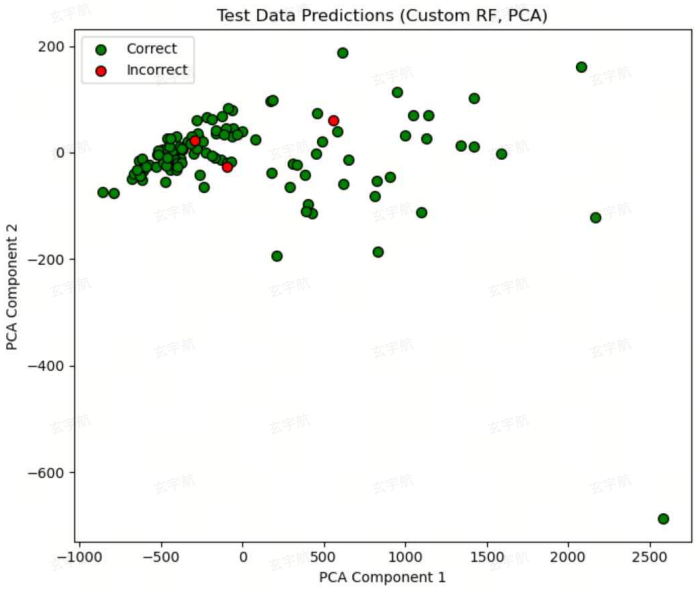
```
from sklearn.decomposition import PCA
pca = PCA(n_components=2) # 我们实例化一个 PCA 对象，并指定我们希望将数据降维到2个主成分（即二维）。
X_pca = pca.fit_transform(X) # fit_transform 方法对数据 X 进行拟合，并返回降维后的数据 X_pca。
```

实验代码中将训练好的模型的预测结果进行了可视化，下图分别展示了训练集和测试集上的原始数据和预测数据的可视化结果。右侧图上绿色为预测准确的样本，红色代表预测错误。可以看到训练集上的预测准确率为100%，而测试机上略低一些。









5.使用鸢尾花数据集进行试验

鸢尾花数据集（Iris Dataset）是机器学习和统计学中最著名的经典数据集之一，常用于分类算法的教学和验证。它最早由英国统计学家和生物学家 Ronald A. Fisher 在1936年提出。鸢尾花数据集包含150个样本，每个样本代表一朵鸢尾花。数据集中共有四个特征和一个标签。特征是花萼和花瓣的长度和宽度，标签是鸢尾花的种类。鸢尾花数据集共有三种不同的鸢尾花类别：Setosa、Versicolor 和 Virginica，每个类别各50个样本。

请使用我们上面所学的方式（自己实现的随机森林或sklearn中的随机森林分类器均可），参考本次实验的步骤完成鸢尾花数据集上的实验。

```
from sklearn.datasets import load_iris
data = load_iris()
X = data.data
y = data.target
```