Iris分类模型流程说明文档

概述

本文档说明如何使用Python实现一个基于组合分类器的Iris分类模型流程。全流程包括数据加载与处理，模型训练与测试，以及使用训练好的模型进行预测。

一：数据加载

模型训练数据加载

1. 数据文件地址：实验中使用bezdekIris.data，路径为E:/Lris/iris/bezdekIris.data。

2. 数据列名和标签映射：

• 列名定义为：['sepal\_length', 'sepal\_width', 'petal\_length', 'petal\_width', 'target']

• 目标标签映射：{'Iris-setosa': 0, 'Iris-versicolor': 1, 'Iris-virginica': 2}

3. 数据加载代码：  
file\_path = 'E:/Lris/iris/bezdekIris.data'

column\_names = ['sepal\_length', 'sepal\_width', 'petal\_length', 'petal\_width', 'target']

data = pd.read\_csv(file\_path, header=None, names=column\_names)

data['target'] = data['target'].map({'Iris-setosa': 0, 'Iris-versicolor': 1, 'Iris-virginica': 2})

测试数据加载

1. 测试文件使用iris.data，路径为E:/Lris/iris/iris.data。

2. 与训练数据相同，列名和标签映射连续使用。

二：数据处理

1. 数据分割为特征和目标标签：

X = data.drop('target', axis=1)

y = data['target']

2.帧化处理：

scaler = StandardScaler()

X\_scaled = scaler.fit\_transform(X)

三：模型训练

1.数据分割：

• 按照 8:2的比例分割训练集和测试集。  
X\_train, X\_test, y\_train, y\_test = train\_test\_split(X\_scaled, y, test\_size=0.2, random\_state=42)

2.初始模型比较：

• 通过交叉验证比较几个基本分类器：

models = []

models.append(('LR', LogisticRegression()))

models.append(('KNN', KNeighborsClassifier()))

models.append(('CART', DecisionTreeClassifier()))

models.append(('SVM', SVC()))

models.append(('RF', RandomForestClassifier()))

for name, model in models:

kfold = cross\_val\_score(model, X\_train, y\_train, cv=5, scoring='accuracy')

print(f"{name}: {kfold.mean():.3f} ({kfold.std():.3f})")

3.模型调变和组合：

• 通过类组查询（GridSearchCV）实现RandomForest的最佳参数。

• 构建组合分类器作为最终模型：  
voting\_clf = VotingClassifier(estimators=[('lr', LogisticRegression()),

('rf', RandomForestClassifier(n\_estimators=100)), ('svc', SVC(probability=True))], voting='soft')

voting\_clf.fit(X\_train, y\_train)

4.测试集预测与评价

(1)预测测试集：

• 使用训练好的模型对测试数据进行预测：

y\_pred = voting\_clf.predict(X\_test)

(2)生成分类报告和混沌矩阵：

print(classification\_report(y\_test, y\_pred))

print(confusion\_matrix(y\_test, y\_pred))

5.模型保存与重载

(1)使用joblib 保存训练好的模型。

joblib.dump(voting\_clf, 'iris\_classifier\_model.joblib')

(2)重载模型并预测测试数据：

• 使用下列代码完成：

model\_path = 'iris\_classifier\_model.joblib'

model = joblib.load(model\_path)

# 重新帧化测试数据

scaler = StandardScaler()

X\_test\_scaled = scaler.fit\_transform(X\_test)

# 预测并输出每个样本的预测结果

y\_pred = model.predict(X\_test\_scaled)

for i in range(len(X\_test)):

print(f"样本 {i + 1}: 特征值: {X\_test.iloc[i].values}, Ground Truth: {y\_test.iloc[i]}, Prediction: {y\_pred[i]}")