代码

```
import pandas as pd
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import f1_score, accuracy_score, recall_score, roc_auc_score
from sklearn.impute import SimpleImputer
from sklearn.preprocessing import LabelEncoder

# 读取数据
data = pd.read_csv('fraudulent.csv')

# 处理缺失值，这里使用众数填充
imputer = SimpleImputer(strategy='most_frequent')
data_imputed = pd.DataFrame(imputer.fit_transform(data), columns=data.columns)

# 将非数值特征转换为数值特征
for column in data_imputed.select_dtypes(include=['object']).columns:
    le = LabelEncoder()
    data_imputed[column] = le.fit_transform(data_imputed[column])

# 划分特征和标签
X = data_imputed.drop('y', axis=1)
y = data_imputed['y']

# 划分训练集和测试集，随机种子设为1
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=1)

# 创建决策树模型
model = DecisionTreeClassifier(random_state=1)

# 使用网格搜索进行超参数调优
param_grid = {
    'max_depth': [None, 10, 20, 30],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4]
}

grid_search = GridSearchCV(model, param_grid, cv=5, scoring='f1', n_jobs=-1)
grid_search.fit(X_train, y_train)

# 获取最佳模型
```

```
best_model = grid_search.best_estimator_

# 预测测试集
y_pred = best_model.predict(X_test)

# 计算评估指标
f1 = f1_score(y_test, y_pred)
accuracy = accuracy_score(y_test, y_pred)
recall = recall_score(y_test, y_pred)
roc_auc = roc_auc_score(y_test, y_pred)

print("F1值:", f1)
print("准确率:", accuracy)
print("召回率:", recall)
print("ROC-AUC:", roc_auc)

# 输出最佳参数
print("最佳参数:", grid_search.best_params_)
```