### 案例 13：基于 DT-AdaBoost 模型的新能源汽车电池故障诊断预测

* **问题背景**：新能源汽车电池的可靠性直接影响车辆的安全性和续航能力，电池故障的早期预测对保障行车安全、降低维修成本至关重要。电池故障受充放电次数、工作温度、电池老化程度、电压电流波动等因素影响，且故障样本在总样本中占比较小，容易被误判。
* **问题描述**：某新能源汽车制造商需要基于电池的实时运行数据预测未来 1 个月内是否会发生故障（分为轻微故障、严重故障、无故障三类）。要求模型能够聚焦容易错分的故障样本，提高预测的容错性和准确性，为电池维护提供预警。
* **数据情况**：提供该品牌 1 万辆新能源汽车过去 2 年的电池运行数据，包括每小时的电压、电流、温度、充放电次数、电池容量衰减率等，以及电池故障记录（故障类型、发生时间）。数据中故障样本占比约 5%，存在部分车辆的传感器数据异常情况。

### 案例 13：DT-AdaBoost 模型新能源汽车电池故障诊断预测代码

|  |
| --- |
| import pandas as pd  import numpy as np  import matplotlib.pyplot as plt  from sklearn.preprocessing import StandardScaler, LabelEncoder  from sklearn.model\_selection import train\_test\_split  from sklearn.tree import DecisionTreeClassifier  from sklearn.ensemble import AdaBoostClassifier  from sklearn.metrics import classification\_report, confusion\_matrix, accuracy\_score  from sklearn.model\_selection import cross\_val\_score  import joblib  # 数据加载与预处理  data = pd.read\_csv('ev\_battery\_fault.csv')  data = data.dropna()  # 故障类型编码  le = LabelEncoder()  data['fault\_type'] = le.fit\_transform(data['fault\_type']) # 0:无故障,1:轻微故障,2:严重故障  # 特征与目标变量  X = data.drop(['battery\_id', 'fault\_type'], axis=1)  y = data['fault\_type']  # 数据标准化  scaler = StandardScaler()  X\_scaled = scaler.fit\_transform(X)  # 划分训练集和测试集  X\_train, X\_test, y\_train, y\_test = train\_test\_split(X\_scaled, y, test\_size=0.2, random\_state=42, stratify=y)  # 构建DT-AdaBoost模型  base\_dt = DecisionTreeClassifier(max\_depth=3, random\_state=42) # 弱分类器  adaboost\_model = AdaBoostClassifier(  base\_estimator=base\_dt,  n\_estimators=50,  learning\_rate=0.1,  random\_state=42  )  # 模型训练  adaboost\_model.fit(X\_train, y\_train)  # 模型预测  y\_pred = adaboost\_model.predict(X\_test)  y\_pred\_proba = adaboost\_model.predict\_proba(X\_test)  # 评估模型  print(f'准确率: {accuracy\_score(y\_test, y\_pred)}')  print('混淆矩阵:')  print(confusion\_matrix(y\_test, y\_pred))  print('分类报告:')  print(classification\_report(y\_test, y\_pred, target\_names=['无故障', '轻微故障', '严重故障']))  # 交叉验证  cv\_scores = cross\_val\_score(adaboost\_model, X\_scaled, y, cv=5)  print(f'交叉验证准确率: {np.mean(cv\_scores)} ± {np.std(cv\_scores)}')  # 特征重要性可视化  feature\_importance = adaboost\_model.feature\_importances\_  features = X.columns  plt.figure(figsize=(10, 6))  plt.barh(features, feature\_importance)  plt.xlabel('特征重要性')  plt.title('AdaBoost特征重要性')  plt.savefig('battery\_feature\_importance.png')  plt.show()  # 保存模型  joblib.dump(adaboost\_model, 'dt\_adaboost\_battery\_fault.pkl')  joblib.dump(scaler, 'scaler\_battery.pkl')  joblib.dump(le, 'label\_encoder\_battery.pkl') |