### 13. 决策树 + 模糊逻辑案例：个人消费贷款审批

**问题背景**：某银行需制定个人消费贷款审批规则，指标含 6 项（如月收入、征信评分、负债比等），需明确 “若月收入> X 且征信评分 > Y，则批准贷款” 的规则，同时处理连续特征的模糊性（如 “月收入中等”）。

**数据**：

* 5000 条贷款申请记录：月收入（3000-50000 元）、征信评分（300-850 分）、负债比（0-100%）等，及审批结果（1 = 批准，0 = 拒绝）。

**要求**：用模糊逻辑将连续特征转为模糊集合（如 “月收入：低 / 中 / 高”），构建模糊决策树，输出可解释的审批规则，确保规则覆盖率 > 80% 且错误率 < 5%。

### 13. 决策树 + 模糊逻辑代码：个人消费贷款审批

|  |
| --- |
| import numpy as np  import pandas as pd  import matplotlib.pyplot as plt  from sklearn.preprocessing import KBinsDiscretizer  from sklearn.tree import DecisionTreeClassifier, plot\_tree  from sklearn.model\_selection import train\_test\_split  from sklearn.metrics import classification\_report, confusion\_matrix  # 1. 数据准备（5000条贷款申请记录）  np.random.seed(42)  n\_samples = 5000  # 6项指标：月收入(元)、征信评分(300-850)、负债比(%)、工作年限(年)、年龄(岁)、贷款金额(元)  monthly\_income = np.random.randint(3000, 50001, n\_samples)  credit\_score = np.random.randint(300, 851, n\_samples)  debt\_ratio = np.random.uniform(0, 100, n\_samples)  work\_years = np.random.randint(0, 31, n\_samples)  age = np.random.randint(18, 66, n\_samples)  loan\_amount = np.random.randint(10000, 500001, n\_samples)  # 生成审批标签（1=批准，0=拒绝）  # 核心逻辑：收入高、征信好、负债低更易批准  approve\_prob = (monthly\_income / 50000) + (credit\_score / 850) - (debt\_ratio / 100) + (work\_years / 30)  approve\_prob = np.clip(approve\_prob, 0, 2) / 2 # 归一化到0-1  y = np.where(np.random.rand(n\_samples) < approve\_prob, 1, 0)  print(f"贷款批准率：{sum(y)/n\_samples:.2%}")  # 特征矩阵  X = np.column\_stack([monthly\_income, credit\_score, debt\_ratio, work\_years, age, loan\_amount])  features = ['月收入', '征信评分', '负债比', '工作年限', '年龄', '贷款金额']  # 2. 模糊逻辑处理连续特征  def fuzzy\_discretize(data, n\_bins=3, feature\_names=None):  """将连续特征离散化为模糊集合（低/中/高）"""  discretizer = KBinsDiscretizer(n\_bins=n\_bins, encode='ordinal', strategy='quantile')  X\_disc = discretizer.fit\_transform(data).astype(int)    # 生成模糊集合名称  fuzzy\_names = []  for i in range(data.shape[1]):  bins = discretizer.bin\_edges\_[i]  names = [f'{feature\_names[i]}\_低', f'{feature\_names[i]}\_中', f'{feature\_names[i]}\_高']  fuzzy\_names.extend(names)    # 构建模糊特征矩阵（one-hot编码）  X\_fuzzy = np.zeros((data.shape[0], data.shape[1] \* n\_bins), dtype=int)  for i in range(data.shape[0]):  for j in range(data.shape[1]):  X\_fuzzy[i, j\*n\_bins + X\_disc[i, j]] = 1  return X\_fuzzy, fuzzy\_names  X\_fuzzy, fuzzy\_features = fuzzy\_discretize(X, feature\_names=features)  # 3. 划分训练集和测试集  X\_train, X\_test, y\_train, y\_test = train\_test\_split(X\_fuzzy, y, test\_size=0.3, random\_state=42)  # 4. 构建模糊决策树  dt = DecisionTreeClassifier(max\_depth=5, min\_samples\_leaf=20, random\_state=42)  dt.fit(X\_train, y\_train)  # 5. 模型评估  y\_pred = dt.predict(X\_test)  print("\n贷款审批分类报告：")  print(classification\_report(y\_test, y\_pred, target\_names=['拒绝', '批准']))  # 6. 提取决策规则  plt.figure(figsize=(20, 10))  plot\_tree(dt, feature\_names=fuzzy\_features, class\_names=['拒绝', '批准'],  filled=True, rounded=True, fontsize=10)  plt.title('模糊决策树审批规则')  plt.show()  # 7. 规则示例解析  print("\n关键审批规则示例：")  print("1. 若 月收入\_高 且 征信评分\_高 且 负债比\_低 → 批准")  print("2. 若 月收入\_低 且 征信评分\_低 → 拒绝")  print("3. 若 月收入\_中 且 工作年限\_高 且 负债比\_中 → 批准") |