### 8. 贝叶斯网络 + 证据理论案例：无人机故障诊断

**问题背景**：某无人机厂商需构建故障诊断系统，融合 5 个传感器数据（电池电压、电机转速、GPS 信号强度等），判断无人机是否存在 “动力系统故障”，传感器数据存在噪声和冲突（如某传感器显示异常但其他正常）。

**数据**：

* 1000 组飞行记录，每组含 5 项传感器数据（数值型）及故障标签（1 = 故障，0 = 正常），部分记录存在传感器数据缺失。

**要求**：用贝叶斯网络建模传感器与故障的因果关系，用证据理论融合冲突数据，输出故障概率及诊断置信度，分析对缺失数据的处理能力。

### 8. 贝叶斯网络 + 证据理论代码：无人机故障诊断

|  |
| --- |
| import numpy as np  import pandas as pd  from pgmpy.models import BayesianNetwork  from pgmpy.estimators import MaximumLikelihoodEstimator  from pgmpy.inference import VariableElimination  from scipy.stats import norm  # 1. 数据准备（模拟1000组飞行记录）  np.random.seed(42)  n\_samples = 1000  # 传感器数据（5项：电池电压、电机转速、GPS信号、温度、振动）  # 正常状态下的分布  battery\_voltage = norm.rvs(loc=12.0, scale=0.5, size=n\_samples)  motor\_speed = norm.rvs(loc=3000, scale=200, size=n\_samples)  gps\_signal = norm.rvs(loc=80, scale=10, size=n\_samples) # 信号强度0-100  temperature = norm.rvs(loc=45, scale=5, size=n\_samples)  vibration = norm.rvs(loc=0.1, scale=0.05, size=n\_samples)  # 故障标签（10%故障概率）  fault = np.random.binomial(1, 0.1, n\_samples)  # 故障状态下传感器数据异常  battery\_voltage[fault == 1] = norm.rvs(loc=9.0, scale=1.0, size=sum(fault))  motor\_speed[fault == 1] = norm.rvs(loc=2000, scale=300, size=sum(fault))  gps\_signal[fault == 1] = norm.rvs(loc=40, scale=15, size=sum(fault))  temperature[fault == 1] = norm.rvs(loc=60, scale=8, size=sum(fault))  vibration[fault == 1] = norm.rvs(loc=0.5, scale=0.2, size=sum(fault))  # 离散化传感器数据（3个状态：低、中、高）  def discretize(data, bins=3):  return np.digitize(data, np.percentile(data, [100\*i/bins for i in range(1, bins)]))  sensors\_discrete = np.column\_stack([  discretize(battery\_voltage),  discretize(motor\_speed),  discretize(gps\_signal),  discretize(temperature),  discretize(vibration)  ])  # 构造数据集  data = pd.DataFrame(sensors\_discrete, columns=['电池电压', '电机转速', 'GPS信号', '温度', '振动'])  data['动力系统故障'] = fault  # 2. 贝叶斯网络建模  model = BayesianNetwork([  ('电池电压', '动力系统故障'),  ('电机转速', '动力系统故障'),  ('GPS信号', '动力系统故障'),  ('温度', '动力系统故障'),  ('振动', '动力系统故障')  ])  # 参数学习  model.fit(data, estimator=MaximumLikelihoodEstimator)  # 3. 证据理论融合  def dempster\_shafer(evidence\_list):  # 基本概率分配（BPA）  bpa = {0: 0.1, 1: 0.1} # 初始不确定性  for evidence in evidence\_list:  new\_bpa = {0: 0, 1: 0}  # 传感器证据支持度  p\_fault = evidence['P(故障|证据)']  new\_bpa[1] += p\_fault \* 0.8  new\_bpa[0] += (1 - p\_fault) \* 0.8  # 剩余概率分配给不确定性  new\_bpa[0] += 0.2 \* bpa[0]  new\_bpa[1] += 0.2 \* bpa[1]  bpa = new\_bpa  # 归一化  total = sum(bpa.values())  return {k: v/total for k, v in bpa.items()}  # 4. 故障诊断示例（随机选择10条记录）  infer = VariableElimination(model)  sample\_indices = np.random.choice(n\_samples, 10, replace=False)  results = []  for idx in sample\_indices:  sample = data.iloc[idx]  evidence = {col: sample[col] for col in ['电池电压', '电机转速', 'GPS信号', '温度', '振动']}  # 贝叶斯网络推理  fault\_proba = infer.query(variables=['动力系统故障'], evidence=evidence)['动力系统故障'].values[1]  # 证据融合  evidence\_list = [{'P(故障|证据)': fault\_proba}]  fused\_bpa = dempster\_shafer(evidence\_list)  # 诊断结果  diagnosis = 1 if fused\_bpa[1] > 0.5 else 0  results.append({  '样本索引': idx,  '实际故障': sample['动力系统故障'],  '贝叶斯故障概率': round(fault\_proba, 4),  '融合后故障概率': round(fused\_bpa[1], 4),  '诊断结果': diagnosis  })  # 展示结果  result\_df = pd.DataFrame(results)  print("无人机故障诊断结果：")  print(result\_df)  print(f"\n诊断准确率：{sum(result\_df['实际故障'] == result\_df['诊断结果'])/len(result\_df):.2%}") |