### 2. TOPSIS + 灰色关联分析案例：乡村旅游项目开发决策

**问题背景**：某县文旅局计划从 3 个乡村旅游开发方案中选 1 个重点扶持，方案涉及生态保护、经济收益、就业带动等矛盾指标（如方案 A 收益高但生态破坏风险大）。由于县域旅游数据积累少（仅 2 年试点数据），需兼顾全局最优与指标关联性。

**数据**：

* 3 个方案的 4 项指标：生态影响评分（满分 10 分：7/8/6）、年预期收益（万元：500/300/400）、带动就业人数（人：120/80/100）、投资回收期（年：3/2/2.5）。

**要求**：用 TOPSIS 确定全局排序，用灰色关联分析补充指标协调度，组合评价后推荐最优方案，说明如何平衡 “收益” 与 “生态”。

### TOPSIS + 灰色关联分析代码：乡村旅游项目开发决策

|  |
| --- |
| import numpy as np  import pandas as pd  # 数据准备  data = np.array([  [7, 500, 120, 3], # 方案1：生态影响、收益、就业、回收期  [8, 300, 80, 2], # 方案2  [6, 400, 100, 2.5] # 方案3  ])  projects = [f'方案{i+1}' for i in range(3)]  # 指标类型：[效益型, 效益型, 效益型, 成本型]  is\_cost = [False, False, False, True]  # 数据标准化  def normalize(data, is\_cost):  norm\_data = np.zeros\_like(data, dtype=float)  for j in range(data.shape[1]):  if is\_cost[j]:  norm\_data[:, j] = data[:, j].max() - data[:, j]  else:  norm\_data[:, j] = data[:, j] - data[:, j].min()  norm\_data[:, j] /= (norm\_data[:, j].max() + 1e-10)  return norm\_data  norm\_data = normalize(data, is\_cost)  # TOPSIS计算  def topsis\_score(norm\_data):  weights = np.ones(norm\_data.shape[1]) / norm\_data.shape[1]  weighted = norm\_data \* weights  ideal\_best = weighted.max(axis=0)  ideal\_worst = weighted.min(axis=0)  d\_best = np.sqrt(np.sum((weighted - ideal\_best)\*\*2, axis=1))  d\_worst = np.sqrt(np.sum((weighted - ideal\_worst)\*\* 2, axis=1))  return d\_worst / (d\_best + d\_worst)  topsis\_scores = topsis\_score(norm\_data)  # 灰色关联分析  def grey\_relation(norm\_data):  reference = norm\_data.max(axis=0)  rho = 0.5  min\_min = np.min(np.min(np.abs(norm\_data - reference)))  max\_max = np.max(np.max(np.abs(norm\_data - reference)))  xi = (min\_min + rho \* max\_max) / (np.abs(norm\_data - reference) + rho \* max\_max)  return np.mean(xi, axis=1)  grey\_scores = grey\_relation(norm\_data)  # 组合评价  combined\_scores = 0.5 \* topsis\_scores + 0.5 \* grey\_scores  result = pd.DataFrame({  '方案': projects,  'TOPSIS得分': topsis\_scores.round(4),  '灰色关联度': grey\_scores.round(4),  '组合得分': combined\_scores.round(4),  '排名': (combined\_scores.argsort()[::-1].argsort() + 1)  }).sort\_values('组合得分', ascending=False)  print(result) |