所属类别	2023 年“华数杯”全国大学生数学建模竞赛	参赛编号
本科组		CM2302178

基于随机森林对母亲身心状况与婴儿成长的关联研究

摘要

大量研究表明婴儿的成长过程中，母亲起到了决定性作用，研究母亲的身体指标和心理指标对婴儿的行为特征和睡眠质量的影响具有十分重要的意义。本文探究了母亲身体指标和心理指标与婴儿行为特征和睡眠质量的关系。

针对问题一，题目要求根据附件判断母亲的身心指标是否会对婴儿的行为特征和睡眠质量产生影响。本文首先对数据进行预处理，将异常数据剔除，并将定类数据转化为定量数据。在对预处理后的数据进行正态检验的基础上，建立相关分析模型，结合 SPSS 软件进行典型相关分析，最终认为母亲的身心指标会对婴儿的行为特征和睡眠质量造成影响。

针对问题二，题目要求建立婴儿行为特征和母亲身心指标的关联模型，并对附件中最后 20 组婴儿的行为特征类型进行预测。由于附件数据存在类别不平衡问题，故采用 SMOTE 算法对数据进行过采样，生成了更加完备的数据样本，用以提高模型训练的准确性。考虑到母亲的部分身心指标变量可能对模型的影响较小，计算出母亲各个身心指标变量的方差，分别剔除了方差为 0.035 的婚姻状况和方差为 0.013 的分娩方式两个变量。最后构造出分类模型，结合贝叶斯公式和随机森林算法进行求解，并预测出最后 20 组婴儿的行为特征类型。

针对问题三，题目要求建立模型分析婴儿的行为特征从矛盾型分别变为中等型和安静型所需要的最少治疗费用和方案。本文首先通过随机森林算法分别求出 238 号婴儿处于中等型和安静型时，母亲的三项心理指标的变化范围，并结合表 1 中心理指标和治疗费用的定量关系建立整数规划模型，结合粒子群算法，求得最佳治疗方案：238 号母亲 CBTS、EPDS、HADS 分别降为 8、8、13 时，婴儿行为特征转为中等型的费用最少为 29024.67 元；三项指标分别降为 5、6、11 时，婴儿行为特征转为安静型的费用最少为 37906.67 元。

针对问题四，题目要求建立婴儿睡眠质量与母亲的身体指标、心理指标的关联模型，并预测出最后 20 组婴儿的睡眠质量。本文首先将区间型指标睡眠时长和极小型指标睡醒次数进行标准化处理，建立了基于熵权法的 TOPSIS 模型，用以确定各指标权重，并使用 K-means++算法对数据进行聚类分析，划分出综合睡眠质量优良中差四类，构建出分类模型。最终采用随机森林算法对模型进行求解，并预测出最后 20 组婴儿的综合睡眠质量。

针对问题五，题目要求在问题三的基础上，对 238 号婴儿的综合睡眠质量评级为优的治疗调整策略。本文结合基于熵权法的 TOPSIS 模型对 238 号婴儿的综合睡眠质量进行评价，通过随机森林算法预测出 238 号婴儿的初始综合睡眠质量评级为中，行为特征转为中等型和安静型的综合睡眠质量评级为良，均未达到优，故在问题三的治疗方案基础上加以调整：238 号母亲的 CBTS、EPDS、HADS 分别降为 11、9、10 时，婴儿行为特征变为中等型且综合睡眠质量为优的费用最少为 33037.67 元；三项指标分别降为 2、3、9 时，婴儿行为特征变为安静型且综合睡眠指标为优的费用最少为 47483.67 元。

关键词：典型相关分析；随机森林；聚类算法；粒子群算法；基于熵权 TOPSIS

一、问题重述

1.1 问题背景

在婴儿的成长过程中，母亲起到了决定性作用。她不仅能为婴儿提供身体保护和其发育所需的营养成分，还能提供必要的情感支持。所以，一旦母亲的心理健康状况不佳，产生焦虑，出现压力乃至抑郁的情况，可能会为婴儿的认知、情感、行为举止等方面带来诸多不利影响，例如影响婴儿的睡眠质量。因此，研究母亲的身体指标和心理指标对婴儿的行为特征和睡眠质量的影响具有十分重要的意义。

1.2 问题提出

问题一：根据附件中的数据进行研究，判断母亲的身体指标和心理指标对婴儿的行为特征和睡眠质量是否有影响。

问题二：婴儿的行为特征被分为三种类型：安静型、中等型和矛盾型。建立母亲身心状况和婴儿行为特征关系的模型，并预测表中最后 20 组婴儿的行为特征类型。

问题三：$CBTS$、$EPDS$、$HADS$ 的治疗费用与母亲三种心理疾病程度的变化率成正比，已知 238 号婴儿行为特征为矛盾型，求出其特征转为中等型和安静型各自所需治疗费用最小值，并给出调整方案。

问题四：根据附件中婴儿的三项睡眠质量指标对婴儿的睡眠质量进行优、良、中、差综合评判，并建立母亲身心状况与对应婴儿的睡眠综合质量关联模型，预测出最后 20 组婴儿的综合睡眠质量。

问题五：在问题三的前提下，若附加条件，使得 238 号婴儿综合睡眠质量为优，问题三中的治疗方案是否需要调整，如果需要，提供新的治疗方案。

二、问题的分析

2.1 问题的整体分析

以上问题是建立母亲的身体指标和心理指标是否会对婴儿的行为特征和睡眠质量产生影响的基础之上的，首先我们需要判断母亲的身体指标和心理指标对婴儿的行为特征和睡眠质量有影响。其次，我们需要建立婴儿的行为特征与母亲的身体指标与心理指标的关系模型。预测出数据表中最后 20 组婴儿的行为特征类型。再次，我们需要根据表 1 求得患病得分与治疗费用的关系，并建立相应的模型，计算出编号为 238 号的婴儿从矛盾型变为中等型、矛盾型变为安静型需要多少费用，并说明治疗方案如何调整。然后我们需要建立婴儿综合睡眠质量与母亲的身体指标、心理指标的关联模型，预测出数据表最后 20 组婴儿的综合睡眠质量。最后，在前几问的基础上，若需要让 238 号婴儿的睡眠质量评级为优，判断问题三的治疗策略是否需要调整，如果有调整要给出调整策略。

2.2 问题一的分析

问题一需要判断母亲的身体指标和心理指标对婴儿的行为特征和睡眠质量是否有影响。我们为解决此问题建立多变量间的相关性分析模型。通过观察和分析附件数据，我们首先对数据进行预处理，并对处理好的数据进行正态检验，遗憾的是数据并不满足正态分布，所以最终采用典型相关分析对该问题进行求解。

2.3 问题二的分析

问题二需要建立母亲身心状况与婴儿行为特征类型的关联模型，并根据所建立模型预测出表中最后 20 组婴儿的行为特征类型。首先，通过对数据的观察，我们发现样本数据中婴儿的各种行为特征类型数量相差较大，会对模型预测的准确性产生较大影响。对此，我们使用 *SMOTE* 算法对数据进行过采样处理，减小了部分样本数量较少对模型准确性的影响。其次，我们对各母亲身心指标变量进行定量的方差分析，发现婚姻状况和分娩方式两个变量的方差极小，不适合作为模型的分类指标，故将两变量剔除。最后，我们对数据进行标准化处理，并采用随机森林算法集合贝叶斯公式进行模型训练，且取到了较好的效果。

2.4 问题三的分析

问题三需要建立模型分析婴儿的行为特征从矛盾型变为中等型、从矛盾型变为安静型两种情况所需的最低费用。由题中三种心理障碍治疗费用相对于患病程度的变化率与治疗费用呈正比，起初，我们建立了微分方程模型用以衡量治疗费用与患病程度的关系，并对 238 号婴儿治疗费用进行预测，但最终金额均达到千万级别，远超查阅相关文献的治疗费用[1]，故考虑题设条件应是三种治疗费用相对患病程度呈正比之意。最终，我们通过建立两个线性整数规划模型，使用非线性因子的粒子群优化算法进行求解，最终求得问题的最优解远低于千万量级，与其他心理疾病治疗费用相差较少[1]。

2.5 问题四的分析

问题四需要建立婴儿睡眠质量与母亲身体指标、心理指标的关联模型并预测最后 20 组婴儿的睡眠质量。我们首先使用基于熵权的 *TOPSIS* 方法获取睡眠时间、睡醒次数和入睡方式对于睡眠质量的权重占比，并采用 K-means++ 算法对综合睡眠质量评分进行聚类分析。最后，我们结合随机森林算法进行模型的训练和预测。

2.6 问题五的分析

问题五需要结合婴儿综合睡眠质量评价方式，对两种方案进行优化调整。我们首先将原治疗方案进行检验，预测出治疗后的婴儿综合睡眠质量均为良，未成功通过检验。故添加新的决策变量，通过随机森林算法对目标区间重新进行遍历，并采用粒子群优化算法将治疗方案进行改进。

三、模型的假设

1. 假设附件各指标数据都是可靠的，反映了一些规律；
2. 假设除附件所列举的母亲各身心指标外，其余因素不会对婴儿的行为特征和睡眠质量产生影响；
3. 假设婴儿自身的身心指标不会对自身的行为特征和睡眠质量产生影响；
4. 假设 238 号婴儿的行为特征可以在其母亲进行焦虑治疗后发生改变。
5. 假设母亲的 *CBTS*、*EPDS*、*HADS* 三种指标得分和治疗费用具有一定的定量关系；

四、符号说明

符号	符号说明
U_i	集合一中标准化的第 i 对典型变量
V_i	集合二中标准化的第 i 对典型变量
X_i/Y_j	原始样本数据指标变量
Z_i^1	原始样本数据指标变量 X_i 标准化后的结果
Z_j^2	原始样本数据指标变量 Y_j 标准化后的结果
E_j	第 j 个指标数据的信息熵
F_j	第 j 个指标数据的冗余度

五、模型的建立与求解

5.1 问题一

5.1.1 数据预处理

根据附件的补充说明对附件中的样本数据进行异常值筛选，筛选后发现有以下异常情况：

- 婴儿睡眠时间异常 (如 181 号数据婴儿睡眠时间为 99)；
- 婚姻状况异常 (如 231 号数据母亲婚姻状况为 6)；

经过数据筛选后总共剔除了 10 组异常数据。

为了更加准确地衡量婴儿的行为特征和睡眠质量与母亲身心指标的关系，我们将婴儿的整晚睡眠时间统一转化为小时，并将婴儿行为特征数据从非数值型数据转化为数值型数据，其中安静型记为 0，中等型记为 1，矛盾型记为 2.

5.1.2 正态检验

正态分布在数据科学领域中十分重要，以概率分布为核心的研究大都聚焦于正态分布。在此问题中，若各数据满足正态分布，将在极大程度上简化模型，于是我们对附件各数据指标进行正态检验，具体如图 1 所示。其中直方图是各数据指标在各值的数量，黄色曲线各数据指标代表核密度估计。

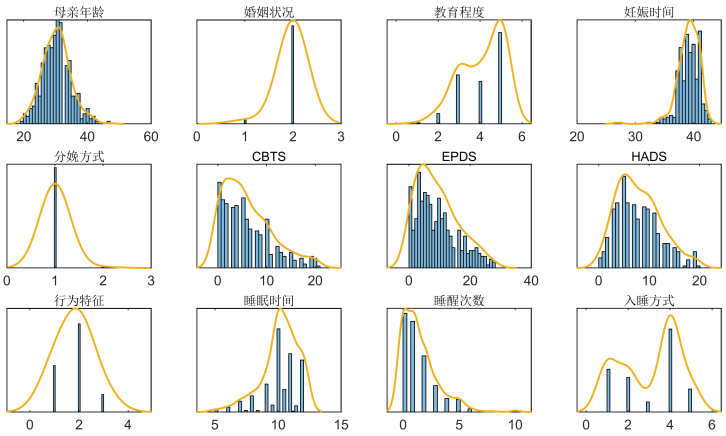

图 1 各数据指标分析图

如图 1 所示，我们发现各数据指标中只有母亲年龄和妊娠时间可近似为正态分布，

其它数据指标无法视作正态分布进行处理。因此，一些需要指标数据满足正态分布的相关性分析方法无法使用，例如 *Pearson* 相关性分析。

5.1.3 问题一模型的建立与求解

由于问题中定量数据由定类数据转化得到，且结合上文正态检验部分得出的结论，我们采用典型相关分析[2] 进行问题一的建模与求解。

典型相关分析步骤如图 2 所示。首先在每组数据中找出变量的线性组合，并计算典型相关系数，使得两组的线性组合之间具有最大的相关系数。其次，计算各典型变量对之间的典型负荷系数和交叉负荷系数。然后选取和最初挑选的该线性组合不相关的线性组合，使其配对，并选取相关系数最大的一对，直至两组变量之间的相关性被提取完毕为止。最后，计算典型相关分析的方差解释比例，对结果进行诠释。

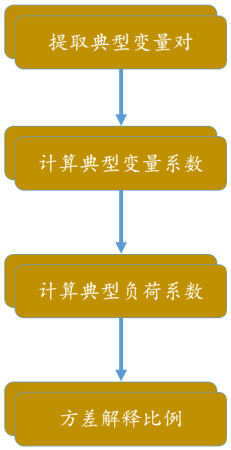

图 2 典型相关分析流程图

在本问题中，我们使用典型相关分析分别探究母亲的身心指标对婴儿的行为特征和睡眠质量是否有影响。

（1）母亲的身心指标对于婴儿睡眠质量：

将母亲年龄、婚姻状况、教育程度、妊娠时间、分娩方式、HADS、EPDS 和 CBTS 划分为集合一，婴儿的睡眠质量划分为集合二采用 SPSS 软件进行典型相关性分析求解。

典型相关性检验表如表 1 所示，共有三对典型变量对，各自的显著性值均小于 0.1，我们认为在 90% 的置信水平下认为两组集合中的变量有关。除此之外，根据显著性的个数，确定典型相关系数的个数为 3。

表 1 典型相关性检验表

典型变量对	相关性	特征值	威尔克统计	F	分子自由度	分母自由度	显著性
1	0.269	0.078	0.880	2.692	18.000	1049.832	0.001
2	0.177	0.032	0.949	1.987	10.000	744.000	0.032
3	0.144	0.021	0.979	1.971	4.000	373.000	0.098

母亲身心健康各指标数据的特征标准化典型相关系数如表 2 所示，据此可得到标准化的典型变量对表达式。因篇幅有限，以表 2 第一列变量为例给出表达式。

$$U_1 = 0.107Z_1^1 + 0.715Z_2^1 + 0.301Z_3^1 \tag{1}$$

表 2 母亲身心健康特征标准化典型相关系数

典型变量对	母亲年龄	妊娠时间	分娩方式	CBTS	EPDS	HADS
1	0.107	0.040	0.067	0.576	-1.333	-0.034
2	0.715	-0.157	-0.477	-0.645	1.002	-0.670
3	0.301	0.902	-0.009	-0.002	-0.486	0.761

婴儿睡眠质量各指标数据的特征标准化典型相关系数如表 3 所示，据此可得到标准化的典型变量对表达式。因篇幅有限，以表 3 第一列变量为例给出表达式。

$$V_1 = 0.865Z_1^2 - 0.144_2^2 + 0.610Z_3^2 \tag{2}$$

表 3 婴儿睡眠质量典型相关系数

典型变量对	婴儿整晚睡眠时间	睡醒次数	入睡方式
1	0.865	-0.399	-0.282
2	0.144	0.134	-0.989
3	0.610	0.983	0.257

母亲身心健康各指标数据的典型载荷系数如表 4 所示，典型载荷系数是指一个典型变量与另一组变量各个变量的简单相关系数，典型载荷系数的绝对值越大说明该项与典型变量之间的相关关系越强。在典型变量对 1 中，$EPDS$ 指标的相关性最强。在典型变量对 2 中，母亲年龄指标的相关性最强。在典型变量对 3 中，妊娠时间指标的相关性最强。

表 4 母亲身心健康典型载荷

典型变量对	母亲年龄	妊娠时间	分娩方式	CBTS	EPDS	HADS
1	0.240	0.020	0.086	-0.503	-0.925	-0.710
2	0.687	0.001	-0.380	-0.319	-0.094	-0.333
3	0.231	0.830	-0.236	-0.009	0.029	0.254

婴儿睡眠质量典型载荷系数如表 5 所示。在典型变量对 1 中，睡眠时间指标的相关性最强。在典型变量对 2 中，入睡方式指标的相关性最强。在典型变量对 3 中，睡醒次数指标的相关性最强。

表 5 婴儿睡眠质量典型载荷

典型变量对	睡眠时间	睡醒次数	入睡方式
1	0.907	-0.158	-0.390
2	-0.578	0.356	0.735
3	0.054	-0.986	0.157

母亲身心健康交叉载荷系数如表 6 所示，交叉载荷系数是指一个典型变量与另一组变量各个变量的简单相关系数，典型载荷系数的绝对值越大说明该项与典型变量之间的

表 6 母亲身心健康交叉载荷

典型变量对	母亲年龄	妊娠时间	分娩方式	CBTS	EPDS	HADS
1	0.064	0.005	0.023	-0.135	-0.249	-0.191
2	0.120	0.001	-0.067	-0.056	-0.017	-0.059
3	0.033	0.119	-0.034	-0.001	0.004	-0.037

相关关系越强。在典型变量对 1 中，*EPDS* 指标的相关性最强。在典型变量对 2 中，母亲年龄指标的相关性最强。在典型变量对 3 中，妊娠时间指标的相关性最强。

婴儿睡眠质量交叉载荷系数如表 7 所示，在典型变量对 1 中，睡眠时间指标的相关性最强。在典型变量对 2 中，睡醒次数指标的相关性最强。在典型变量对 3 中，入睡方式指标的相关性最强。

表 7 婴儿睡眠质量交叉载荷

典型变量对	睡眠时间	睡醒次数	入睡方式
1	0.244	-0.028	0.056
2	-0.155	0.063	0.106
3	0.014	-0.174	0.023

典型相关性的已解释方差比例如表 8 所示，根据方差解释比例，可以定量的判断典型变量所包含的原始信息量，检验变量是否有效。

表 8 已解释的方差比例

典型变量	集合 1* 自身	集合 1* 集合 2	集合 2* 自身	集合 2* 集合 1
1	0.280	0.020	0.387	0.028
2	0.137	0.004	0.375	0.012
3	0.144	0.003	0.239	0.005

（2）母亲的身心指标对于婴儿的行为特征：

将母亲年龄、婚姻状况、教育程度、妊娠书简、分娩方式、HADS、EPDS 和 CBTS 划分为集合一，婴儿的行为特征划分为集合二采用 SPSS 软件进行典型相关分析求解。

典型相关性检验表如表 9 所示，共有一对典型变量对，其显著性值均小于 0.15，我们认为在 85% 的置信水平下认为两组集合中的变量有关。除此之外，根据显著性的个数，确定典型相关系数的个数为 1。

表 9 典型相关性检验表

典型变量对	相关性	特征值	威尔克统计	F	分子自由度	分母自由度	显著性
1	0.159	0.026	0.975	1.621	6.000	373.000	0.140

母亲身心健康和婴儿行为特征各指标数据的特征标准化典型相关系数如表 10 所示，据此可得到标准化的典型变量对表达式。

母亲身心健康和婴儿行为特征各指标数据的典型载荷系数如表 11 所示，在典型变量对 1 中，*EPDS* 指标的相关性最强。

表 10 母亲身心健康和婴儿行为特征标准化典型相关系数

典型变量对	母亲年龄	妊娠时间	分娩方式	CBTS	EPDS	HADS	婴儿行为特征
1	0.613	0.171	-0.001	0.160	-0.604	-0.255	1.000

表 11 母亲身心健康和婴儿行为特征典型载荷

典型变量对	母亲年龄	妊娠时间	分娩方式	CBTS	EPDS	HADS	婴儿行为特征
1	0.687	0.169	0.012	-0.539	-0.762	-0.692	1.000

表 12 母亲身心健康和婴儿行为特征交叉载荷

典型变量对	母亲年龄	妊娠时间	分娩方式	CBTS	EPDS	HADS	婴儿行为特征
1	0.109	0.027	0.002	-0.086	-0.121	-0.110	0.159

母亲身心健康和婴儿行为特征各指标数据的交叉载荷系数如表 12 所示，在典型变量对 1 中，*EPDS* 指标的相关性最强。

典型相关性的已解释方差比例如表 13 所示，根据方差解释比例，可以定量的判断典型变量所包含的原始信息量，检验变量是否有效。

表 13 已解释的方差比例

典型变量	集合 1* 自身	集合 1* 集合 2	集合 2* 自身	集合 2* 集合 1
1	0.308	0.008	1.000	0.025

综上所述，根据上述模型模型结果，我们认为母亲的身心指标对婴儿的行为特征和睡眠质量都具有相关性，即认为母亲的身心指标会影响婴儿的行为特征和睡眠质量。

5.2 问题二

5.2.1 数据预处理

通过分析附件中的数据，我们发现数据存在严重的类别不平衡问题，如图 3 所示。即婴儿的各种行为特征类型的数量相差较大，对基于机器学习算法的分类模型训练的准确性有较大的影响。

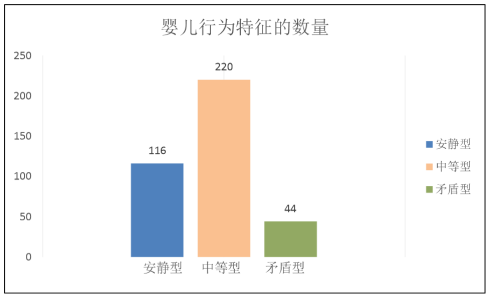

图 3 婴儿各行为特征类型数量

首先，为了解决此问题，我们采用 *SMOTE* 算法对数据进行过采样[3]，生成更加完备的数据样本。*SMOTE* 算法的产生新数据样本的过程如下：

1. 对于少数类中每一个样本 x，以欧氏距离为标准计算它到少数类样本集中所有样本的距离，得到其 k 近邻；

2. 根据样本不平衡比例设置采样比例以确定采样倍率 N，对于每一个少数类样本 x，从其 k 近邻中随机选择若干个样本，假设选择的近邻为 x_n；

3. 对于每一个随机选出的近邻 x_n，分别与原样本按照如下的公式构建新的样本；

$$x_{new} = x + rand(0,1) * (\overline{x} - x) \tag{3}$$

经 *SMOTE* 算法对数据进行处理后，使得安静型、中等型和矛盾型的婴儿行为特征数量大致相等。

其次，由于母亲的身心指标变量过多，会出现部分无关数据不同程度地降低模型性能的问题。于是，我们计算了母亲各身心指标变量的方差，具体如表 14 所示。

表 14 各指标变量方差

母亲年龄	婚姻状况	教育程度	妊娠时间	分娩方式	CBTS	EPDS	HADS
19.470	0.035	0.975	3.597	0.013	25.293	46.170	18.568

由表中的数据可发现婚姻状况和分娩方式的方差过小，不具有明显的区分性，故将其剔除，只使用剩下的 6 类指标变量作为分类依据。

最后，我们使用 $z-score$ 标准化方法对除教育程度以外的指标数据进行标准化，标准化公式如下：

$$x' = \frac{x - \mu}{s} \tag{4}$$

其中 x 是原始数据，μ 是原始数据均值，s 是原始数据标准差。

5.2.2 问题二模型建立与求解

我们采用随机森林算法结合贝叶斯公式对构造的分类模型进行求解[4]。随机森林算法由很多决策树构成，且不同的决策树之间没有关联。当我们进行分类任务时，新的输入样本进入，就让森林中的每一棵决策树分别进行判断和分类，每个决策树各自得到一个分类结果，随机森林会将决策树中分类最多的结果作为最终的输出样本。

（1）构造随机森林的步骤：

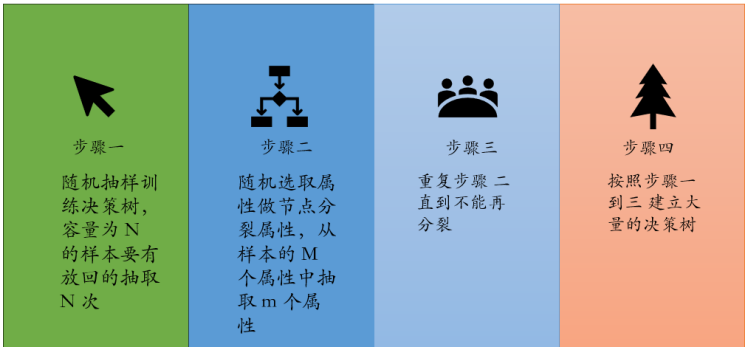

图 4 构造随机森林的步骤

（2）随机森林算法部分参数：

我们在预处理好的样本数据中随机抽取 80% 的样本数据作为训练样本数据，剩下的 20% 样本数据作为测试样本。

表 15 相关参数

参数名	参数值
决策树数量	120
决策树最大深度	15
叶子节点最大数量	15
内部节点分类的最小样本树	10

训练过程中的部分参数值如下表所示：

（3）结果分析：

图 5 为母亲各自身心指标在分类模型中的重要程度占比，从中可以发现，HADS 指标占据的权重最大，母亲年龄占据的权重最小。

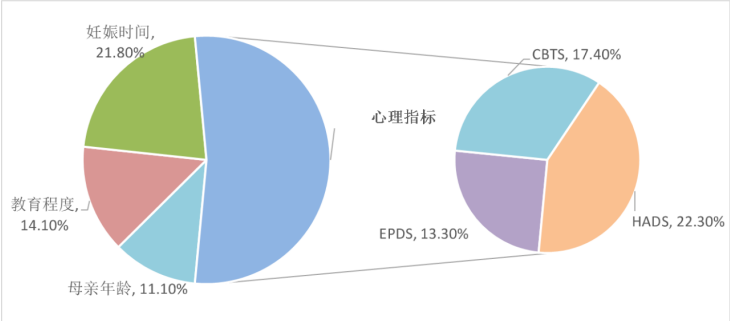

图 5 各指标特征重要性

图 6 为测试集分类结果的混淆矩阵，从中可以大致看出大部分样本分类正确，表明我们构建的随机森林分类模型效果较好。

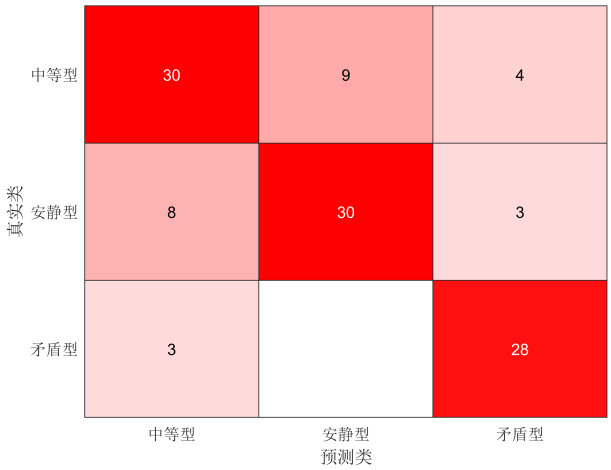

图 6 混淆矩阵

表 16 列出了部分使用测试集验证模型分类性能的指标数据：

- 召回率：所有实际为正类的样本中，被预测为正类的比例；
- 精确率：正类预测的准确性；
- F1 score：精确率和召回率的调和平均值；

表 16 测试集模型分类结果

婴儿行为特征	召回率	精确率	F1 score
安静型	0.7317	0.7692	0.7500
中等型	0.9032	0.8000	0.8485
矛盾型	0.6977	0.7317	0.7143

除上表所列指标之外，我们还以准确率衡量了模型的性能，正确率表示正确预测样本数与总样本数之间的比率。经过计算得到准确率的值为 0.7652。

综上所述，我们构架的随机森林模型在样本数据上取得了较为良好的性能。我们使用训练好的模型对数据表中最后的 20 组数据进行分类预测，预测结果如表 17 所示。

表 17 模型分类预测结果

编号	婴儿的行为特征类型	编号	婴儿的行为特征类型
391	安静型	401	中等型
392	中等型	402	中等型
393	矛盾型	403	中等型
394	中等型	404	安静型
395	中等型	405	中等型
396	中等型	406	安静型
397	矛盾型	407	中等型
398	安静型	408	安静型
399	中等型	409	安静型
400	中等型	410	中等型

5.3 问题三

为解决该问题，定义以下变量：

- 设 238 号婴儿行为特征为中等型时母亲的 $CBTS$、$EPDS$ 和 $HADS$ 的得分变化区间分别为 D_1、D_2 和 D_3；
- 设 238 号婴儿行为特征为安静型时母亲的 $CBTS$、$EPDS$ 和 $HADS$ 的得分变化区间分别为 D_4、D_5 和 D_6；
- 设 238 号婴儿行为特征为中等型时母亲的 $CBTS$、$EPDS$ 和 $HADS$ 的得分变化量分别为 x_1、x_2 和 x_3；
- 设 238 号婴儿行为特征为安静型时母亲的 $CBTS$、$EPDS$ 和 $HADS$ 的得分变化量分别为 x_4、x_5 和 x_6；
- 设 238 号母亲治疗 $CBTS$、$EPDS$、和 $HSDS$ 至其婴儿行为特征指标为中等型时所需开销分别为 y_1、y_2 和 y_3；
- 设 238 号母亲治疗 $CBTS$、$EPDS$、和 $HSDS$ 至其婴儿行为特征指标为中等型时所需开销分别为 y_4、y_5 和 y_6；

5.3.1 问题三模型的建立

对于得分区间 D_1、D_2、D_3、D_4、D_5 和 D_6，根据附件中的样本数据，我们采用随机森林算法遍历所有样本数据得出各得分区间的准确位置。

经分析，$CBTS$、$EPDS$、$HADS$ 三种疾病治疗费用与患病程度成正比。为了解决此问题，我们通过建立两个整数规划模型进行求解。

（1）婴儿行为特征由矛盾型变为中等型

$$\text{目标函数：} Min\ W_1 = y_1 + y_2 + y_3$$

$$s.t. \begin{cases} y_i = k_i x_i + c_i \\ x_i \epsilon D_i \\ i \epsilon \{1,2,3\} \end{cases} \tag{5}$$

结合已知表 1 中的相关数据，通过待定系数法求得系数如表 18 所示。

表 18 约束条件各系数值

i	k_i	c_i
1	2612/3	200
2	695	500
3	2440	300

（2）婴儿行为特征由矛盾型变为安静型

$$\text{目标函数：} Min\ W_2 = y_4 + y_5 + y_6$$

$$s.t. \begin{cases} y_j = k_i x_j + c_i \\ x_j \epsilon D_j \\ i \epsilon \{1,2,3\} \\ j \epsilon \{4,5,6\} \end{cases} \tag{6}$$

5.3.2 模型的求解

（1）非对称学习因子的粒子群优化算法：

为了求解上述的两个规划模型，我们采用非对称学习因子的粒子群优化算法[5] 进行求解。其算法实现步骤如图 7 所示。

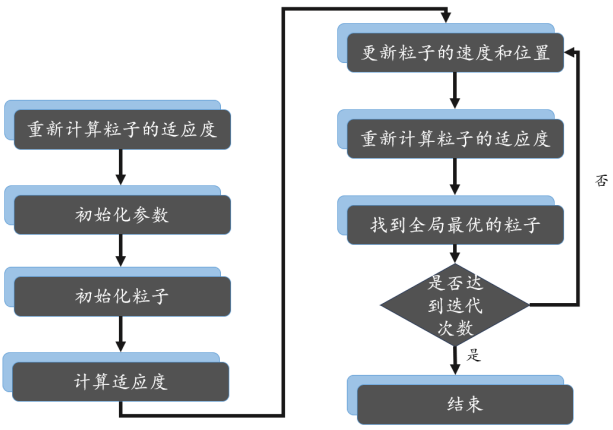

图 7 非对称学习因子粒子群优化算法的实现步骤

粒子群优化算法的核心思想是利用群体中的个体对信息的共享使整个群体的运动在问题求解空间中产生从无序到有序的演化过程，从而获得问题的可行解。

非对称学习因子粒子群优化算法的核心思想是如下公式：当前粒子第 α 步的速度等于上一步自身的速度惯性加自我认知部分加社会认知部分。

$$v_i^{\alpha} = wv_i^{\alpha-1} + c_1 r_1(pbest_i^{\alpha} - x_i^{\alpha}) + c_2 r_2(gbest^{\alpha} - x_i^{\alpha}) \tag{7}$$

（2）算法参数：

经过我们多次实验，得到了效果较为好的模型结果，粒子群优化算法的部分参数如表 19 所示。

表 19 粒子群优化算法的部分参数

参数名	参数值
种群数量	50
迭代次数	200
惯性权重	0.9
空间维数	3

（3）结果分析：

经过 *MATLAB* 进行模型求解：当 x_1=7，x_2=14，x_3=5 时，目标函数 W_1 最小。即 238 号母亲 *CBTS* 治疗得分 7 分，*EPDS* 治疗得分 14 分，HADS 治疗得分 5 分时，238 号婴儿的行为特征由矛盾型转变为中等型的费用最少，为 29024.67 元。

当 x_3=10，x_2=16，x_3=7 时，目标函数 W_2 最小。即 238 号母亲 CBTS 治疗得分 10 分，EPDS 治疗得分 16 分，HADS 治疗得分 17 分时，238 号婴儿的行为特征由矛盾型转变为安静型的费用最少，为 37906.67 元。

5.4 问题四

5.4.1 影响睡眠质量各指标权重的确定

为了解决此问题，我们建立了基于熵权的 *TOPSIS* 模型[6]，对影响婴儿睡眠质量各指标权重进行评定。

（1）数据预处理：

评价婴儿睡眠质量的指标数据有睡眠时间、睡醒次数和入睡方式，通过查阅资料，我们认为上述三种指标数据分别为区间型指标数据、极小型指标数据和极大型指标数据。为了更好地构建 *TOPSIS* 模型，对以上数据进行以下处理[7]：

1. 对睡眠时间数据进行区间型指标正向化处理，通过查阅资料，我们发现 3 至 12 个月婴儿的最佳睡眠时间是 13 到 18 个小时[8]。据此，我们将睡眠时间区间型指标数据转换为极大型指标数据，转化公式如下：

$$M = max\{a - min\{x_i\}, max\{x_i\} - b\}$$

$$\hat{x_i} = \begin{cases} 1 - \frac{a - x_i}{M}, x_i \leqslant a \\ 1, a < x_i < b \\ 1 - \frac{x_i - b}{M}, x_i \geqslant b \end{cases} \tag{8}$$

其中，a 和 b 是最佳区间 $[a, b]$ 的端点，x_i 是原始数据。

2. 对睡醒次数据进行极小型指标正向化处理，转化公式如下：

$$\hat{a}_i = max - a_i \tag{9}$$

其中 a_i 是原始睡醒次数数据，max 指标数据中最大睡醒次数。

3. 数据标准化处理：为消除不同数据指标量纲的影响，我们对已经正向化的数据进行标准化，标准化公式如下：

$$Z_{ij} = \frac{x_{ij}}{\sqrt{\sum_{i=1}^{n} x_{ij}^2}} \tag{10}$$

其中 x_{ij} 为正向化数据矩阵。

（2）权重指标的确定：

根据熵的定义计算各指标的熵值：

$$E_j = \frac{1}{lnm} \sum_{i=1}^{m} x_{ij} * lnZ_{ij} \tag{11}$$

其中 Z_{ij} 是上文标准化后的数据，lnm 是一个常数，指标冗余度 $F_j = 1 - E_j$, 则求得各指标的熵权为：

$$W_j = \frac{F_j}{\sum_{j=1}^{n} F_j} \tag{12}$$

求得各指标熵权后，采用 $TOPSIS$ 方法，经过 $MATLAB$ 求解，可获得三种指标的权重，如下表所示。

表 20 睡眠质量各指标权重

评价指标	睡眠时间	睡醒次数	入睡方式
权重	0.2524	0.1176	0.6300

5.4.2 指标数据类型的聚类

我们使用了 k-means++ 算法对样本数据的婴儿睡眠质量进行聚类。k-means++ 算法步骤如下：

1. 从数据集 ψ 随机选取一个样本点作为第一个初始聚类中心 c_i；

2. 计算每个样本与当前已有聚类中心之间的最短距离，用 $D(x)$ 表示；

3. 计算每个样本点被选为下一个聚类中心的概率 $P(x)$，最后选择最大概率值所对应的样本点作为下一个簇中心；

4. 重复第 2 和第 3 步，直到选择出聚类中心；

结合 $TOPSIS$ 各评价指标权重，我们根据得分大小即聚类中心位置的差异，将样本数据聚类为 4 类，对应睡眠质量的差、中、良、优四个等级，如下表所示。

表 21 各聚类得分与数量

聚类	差	中	良	优
得分 (聚类中心位置)	0.203	0.753	0.406	0.887
数量	11	198	142	29

5.4.3 预测综合睡眠质量

最后，我们得到了样本数据的婴儿睡眠质量的分类标签，我们仍采用随机森林算法训练模型并预测最后 20 组婴儿的睡眠质量。

（1）随机森林算法部分参数：

我们在预处理好的样本数据中随机抽取 80% 的样本数据作为训练样本数据，剩下的 20% 样本数据作为测试样本。

训练过程中的部分参数值如下表所示：

表 22 相关参数

参数名	参数值
决策树数量	100
决策树最大深度	20
叶子节点最大数量	10
内部节点分类的最小样本树	12

（2）结果分析：

图 8 为母亲各自身心指标在分类模型中的重要程度占比，从中可以发现，HADS 指标占据的权重最大，母亲年龄占据的权重最小。

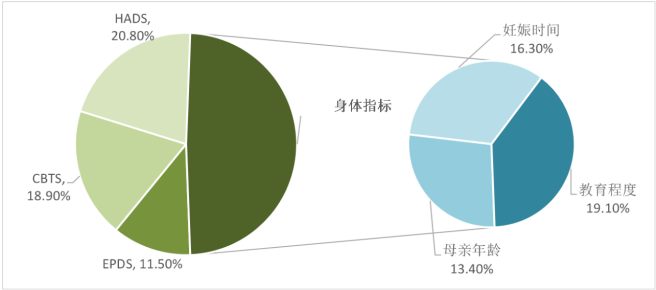

图 8 各指标特征重要性

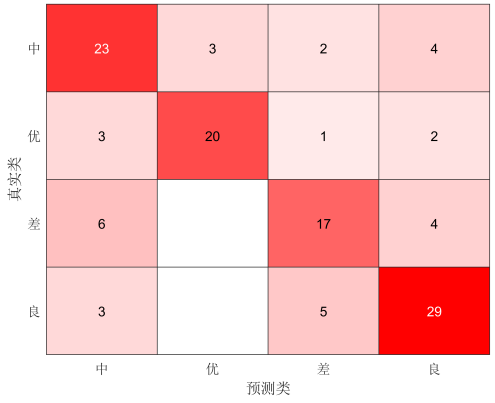

图 9 混淆矩阵

图 9 为测试集分类结果的混淆矩阵，从图中可以看出大部分样本分类正确，表明我们构建的模型分类效果较好。

表 23 测试集模型分类结果

婴儿睡眠质量	召回率	精确率	F1 score
差	0.6296	0.6800	0.6538
中	0.7188	0.6571	0.6866
优	0.7838	0.7436	0.07632
良	0.7692	0.8696	0.8163

表 23 列出了使用测试集测试模型分类结果指标：

除上表所列指标之外，我们还以准确率衡量了模型的性能，正确率表示正确预测样本数与总样本数之间的比率。经过计算得到准确率的值为 0.7295。

综上所述，我们构建的随机森林模型在样本数据上取得了较为良好的性能。我们使用训练好的模型对数据表中最后得 20 组数据进行分类预测，得到的结果如表 24 所示。

表 24 模型分类预测结果

编号	婴儿的睡眠质量	编号	婴儿的睡眠质量
391	良	401	差
392	良	402	差
393	差	403	中
394	良	404	中
395	良	405	良
396	中	406	良
397	良	407	中
398	中	408	良
399	中	409	优
400	良	410	中

5.5 问题五

为了解决该问题，定义以下变量：

- 设 238 号婴儿行为特征为中等型且睡眠质量为优时母亲的 $CBTS$、$EPDS$ 和 $HSDS$ 的得分变化区间分别为 D_1^*、D_2^* 和 D_3^*；
- 设 238 号婴儿行为特征为安静型且睡眠质量为优时母亲的 $CBTS$、$EPDS$ 和 $HSDS$ 的得分变化区间分别为 D_4^*、D_5^* 和 D_6^*；
- 设 238 号婴儿行为特征为中等型且睡眠质量为优时母亲的 $CBTS$、$EPDS$ 和 $HSDS$ 的得分变化量分别为 x_1^*、x_2^* 和 x_3^*；
- 设 238 号婴儿行为特征为安静型且睡眠质量为优时母亲的 $CBTS$、$EPDS$ 和 $HSDS$ 的得分变化量分别为 x_4^*、x_5^* 和 x_6^*；
- 设 238 号母亲治疗 $CBTS$、$EPDS$ 和 $HSDS$ 至其婴儿行为特征指标为中等型且综合睡眠质量为优所需开销为 y_1^*； y_2^* 和 y_3^*；

• 设 238 号母亲治疗 $CBTS$、$EPDS$ 和 $HSDS$ 至其婴儿行为特征指标为安静型且综合睡眠质量为优所需开销为 y_4^*；y_5^* 和 y_6^*。

5.5.1 数据的检验

首先我们结合问题四中构造的随机森林预测模型和 238 号婴儿睡眠时长、睡醒次数、入睡时间数据，获得其综合睡眠质量评级为中，并与问题四中通过熵权法 $topsis$ 以及 k-means++ 聚类得到的结果相吻合。同时，我们分别将问题三中使婴儿的行为特征从矛盾型变为中等型和安静型的两种治疗方案送入网络中，预测结果婴儿的睡眠质量均为良。所以，仅由问题三提出的两种治疗方案都不能满足使 238 号婴儿行为特征为中等型或安静型的同时，使其综合睡眠质量为优。即两种治疗方案都需要进行调整。

5.5.2 模型的改进

当 238 号婴儿行为特征转为中等型时，将原样本数据结合决策变量综合睡眠质量，再次通过随机森林算法对新样本数据进行遍历，并将新获得的 238 号母亲 $CBTS$、$EPDS$、$HADS$ 指标的得分区间定义为 D_1^*、D_2^*、D_3^*。同理，当 238 号婴儿行为特征转为安静型时，将其母亲三项心理指标的新得分区间定义为 D_4^*、D_5^*、D_6^*。

（1）婴儿行为特征由矛盾型变为中等型

对问题三中整数规划模型优化如下：

$$\text{目标函数：} Min\ W_1^* = y_1^* + y_2^* + y_3^*$$

$$s.t.\begin{cases} y_i^* = k_i x_i^* + c_i \\ x_i^* \epsilon D_i^* \\ i\epsilon\{1,2,3\} \end{cases} \tag{13}$$

（2）婴儿特征由矛盾型变为安静型

$$\text{目标函数：} Min\ W_2^* = y_4^* + y_5^* + y_6^*$$

$$s.t.\begin{cases} y_j^* = k_i x_j^* + c_i \\ x_j^* \epsilon D_j^* \\ i\epsilon\{1,2,3\} \\ j\epsilon\{4,5,6\} \end{cases} \tag{14}$$

5.6 模型的求解

通过 $MATLAB$ 结合非对称学习因子的粒子群优化算法，分别对式（13）、式（14）所建立的优化整数规划模型进行求解：

当 x_1^* 为 4，x_2^* 为 13，x_3^* 为 8 时，目标函数 W_1^* 取得最小值 33037.67。即当 238 号母亲的 $CBTS$ 指标治疗 4 分，$EPDS$ 指标治疗 13 分，$HADS$ 指标治疗 8 分时，其婴儿行为特征由矛盾型转为中等型，且综合睡眠质量为优的治疗花费最少为 33037.67 元。

当 x_4^* 为 13，x_5^* 为 19，x_6^* 为 9 时，目标函数 W_2^* 取得最小值 47483.67。即当 238 号母亲的 $CBTS$ 指标治疗 13 分，$EPDS$ 指标治疗 19 分，$HADS$ 指标治疗 9 分时，其婴儿行为特征由矛盾型转为安静型，且综合睡眠质量为优的治疗花费最少为 47483.67 元。

六、模型的评价与改进

6.1 模型的优点

- 通过将数据描述性统计化可以方便直观地观察分析，帮助我们直观、快捷地寻找到数据间的关系，寻找普适规律，使模型建立的数据信息更加可靠，更加贴近实际。在定性的描述之后进行定量的计算，使结果更加可靠；
- 通过对模型的层层检验和比较，使模型更加可靠的同时能适应更加复杂的实际情况，模型简洁实用，可移植性强；

6.2 模型的缺点

- 由于数据量过多，处理十分复杂，不同的模型所要求的数据不同，故我们未将建立的模型与其他模型进行对比；
- 问题三构建的规划模型，约束条件单一，忽略了一些因素；
- 模型求解时间过长，消耗计算资源过多；

6.3 模型的改进

- 丰富问题三和问题五的约束条件，尝试更多目标函数与约束条件，将结果进行横向对比，确定最优模型；
- 减少模型假设，提高模型的普遍性；

参考文献

[1] 熊先军, 李静湖, 王丽莉等. 心理咨询和心理治疗国内外管理基本情况分析 [J]. 中国医疗保险,2012(05):40-43.

[2] 马晰君. 多视角典型相关分析的算法及应用 [D]. 上海财经大学,2022.DOI:10.27296/d.cnki.gshcu.2022.000003.

[3] 王超学, 潘正茂, 董丽丽等. 基于改进 SMOTE 的非平衡数据集分类研究 [J]. 计算机工程与应用,2013,49(02):184-187+245.

[4] 雍凯. 随机森林的特征选择和模型优化算法研究 [D]. 哈尔滨工业大学,2011.

[5] 毛开富, 包广清, 徐驰. 基于非对称学习因子调节的粒子群优化算法 [J]. 计算机工程, 2010(19):188-190

[6] 赵静, 王婷, 牛东晓. 用于评价的改进熵权 TOPSIS 法 [J]. 华北电力大学学报,2004(03):68-70.

[7] 陈鹏宇. 线性无量纲化方法对比及反向指标正向化方法 [J]. 运筹与管理,2021,30(10):95-101.

[8] 国家卫生和计划生育委员会.0 岁～5 岁儿童睡眠卫生指南:WS/T 579-2017[S]. 北京: 中国标准出版社, 2018.

附 录

附录 1：代码

问题二的代码：

```
%question_2.m文件
clc
clear
import imblearn.over_sampling.SMOTE;
%全部数据
data=xlsread("D:\shuweibei\第二问数据_1.xlsx");
%要预测的391-410的数据
pre_data=xlsread("D:\shuweibei\第二问数据_2.xlsx");
%使用Smote算法对不均衡数据进行过采样
% 计算安静型、中等型、矛盾型类型的数量
class_num = size(unique(data(:, end)),1);
class_cnt = histc(data(:, 9), class_num);
max_cnt = max(class_cnt);
smote_data = [];
%进行smote采样
for i=1:max_cnt
    class_data = data(data(:, end) == unique(data(:, end))(i), :);
    sample= datasample(class_data,1);
    houxuan = nearestneighbour(sample', class_data', 'NumberOfNeighbours');
    houxuan(:,1) = [] ;
    rn=ceil(rand(1)*(size(houxuan,2)));
    select_ind = houxuan(:,rn);
    g = class_data(select_ind,:);
    alpha = rand(1);
    snew = sample(1,:) + alpha.*(g-sample(1,:));
    smote_data = [smote_data;snew];
end
%将使用smote过采样得到的结果和要预测的数据组合
data=[smote_data(:,1:8);pre_data];
data_temp=data;
%对数据进行标准化
normal=[data(:,1),data(:,4),data(:,6:8)];
normal_data = (normal - mean(normal)) ./ std(normal);
%将标准化的部分数据替换以及标准化的数据
data=[normal_data(1:660,1),data(1:660,2:3),normal_data(1:660,2),data(1:660,5),
    normal_data(1:660,3:5),smote_data(1:660,9)];
pre_data=[normal_data(661:680,1),data_temp(661:680,2:3),normal_data(661:680,2),
    data_temp(661:680,5),normal_data(661:680,3:5)];
%设置输入变量以及label
x_data = data(:,1:8);
label = data(:,9);
%%使用贝叶斯方法对随机森林进行参数优化
% 设置优化的参数
vars = [
    optimizableVariable('NumPredictorsToSample', [1, 8], 'Type', 'integer');
    optimizableVariable('MinLeafSize', [1, 40], 'Type', 'integer');
    optimizableVariable('NumTrees', [10, 100], 'Type', 'integer');
    optimizableVariable('MaxNumSplits', [2, 10], 'Type', 'integer');
```

```
    ];
% 定义目标函数
minfn = @(T)kfoldLoss(fitcensemble(X_train, Y_train, 'Method', 'Bag', '
    NumLearningCycles', T.NumTrees, 'Learners', ...
    templateTree('MaxNumSplits', T.MaxNumSplits, 'NumPredictorstoSample', T.
        NumPredictorsToSample, 'MinLeafSize', T.MinLeafSize)));
% 使用贝叶斯优化来搜索最佳参数
results = bayesopt(minfn, vars, 'Verbose', 1, 'AcquisitionFunctionName', 'expected
    -improvement-plus');

% 提取最优参数
best_NumPredictorsToSample = results.XAtMinEstimatedObjective.
    NumPredictorsToSample;
best_MinLeafSize = results.XAtMinEstimatedObjective.MinLeafSize;
best_NumTrees = results.XAtMinEstimatedObjective.NumTrees;
best_MaxNumSplits = results.XAtMinEstimatedObjective.MaxNumSplits;
% 使用最优参数创建随机森林模型
rf = TreeBagger(best_NumTrees, x_data,label, 'Method', 'classification', '
    MinLeafSize', best_MinLeafSize, ...
    'NumPredictorsToSample', best_NumPredictorsToSample, 'MaxNumSplits',
        best_MaxNumSplits);
% 生成随机下标
indices = randperm(size(x_data,1));
% 20%的数据用于验证
test_index = indices(1:round(0.2 *size(x_data,1)));
data_test = x_data(te_index,:);
label_test = label(te_index);
% 预测测试集标签
y_pred = predict(rf, data_test);
% 计算准确率
right_rate = sum(str2double(y_pred) == label_test) / size(label_test,1);
disp(['验证集的准确率为: ' num2str(right_rate)]);
% 计算混淆矩阵以及各种指标
y_pred=str2double(y_pred);
y_true=label_test;
%创建混淆矩阵
confusion_matrix = confusionmat(y_true, y_pred);
confusionchart(confusion_matrix, {'安静型', '中等型','矛盾型'});
num_class = size(confusion_matrix, 1);
precision = zeros(num_class, 1);
recall = zeros(num_class, 1);
f1_score = zeros(num_class, 1);
for i=1:num_class
    TP = confusion_matrix(i,i);
    FP = sum(confusion_matrix(:,i)) - TP;
    FN = sum(confusion_matrix(i,:)) - TP;
    precision(i) = TP / (TP + FP);
    recall(i) = TP / (TP + FN);
    f1_score(i) = 2 * precision(i) * recall(i) / (precision(i) + recall(i));
end
% 输出每个类别的精度、召回率和F1分数
disp('每一类的精度: '), disp(precision)
disp('每一类的真正类率: '), disp(recall)
disp('每一类的F1分数: '), disp(f1_score)
% 预测的391-410的数据
```

```
pre_label= str2double(predict(rf, pre_data))
```

问题三的代码：

```
%fun.m文件

function y = fun(x)
%导入第二问贝叶斯寻优获得的最佳参数
    data238=[24 2 2  26.5   1   15  22  18];
    rf = TreeBagger(best_NumTrees, x_data,label, 'Method', 'classification', '
        MinLeafSize', best_MinLeafSize, ...
    'NumPredictorsToSample', best_NumPredictorsToSample, 'MaxNumSplits',
        best_MaxNumSplits);
    %对数据进行标准化
    normal=[data238(1),data238(4),data238(6:8)];
    normal_data = (normal - mean(normal)) ./ std(normal);
    pre_data=[normal_data(1),data_temp(2:3),normal_data(2),data_temp(5),
        normal_data(3:5)];
    pred_label=str2double(predict(rf, pre_data));
    %当预测结果为中等型，则不加惩罚项
    if pred_label==1
        y = 2612/3*x(1) + 695*x(2)^2 +2440*x(3)+1000;
    else
        %当预测的结果不为中等型时，增加一个惩罚项
        y=2612/3*x(1) + 695*x(2)^2 +2440*x(3)+1000+9999999;
    end
end

%question_3_xiugai.m

clear; clc
%设置粒子群的初始参数
particle_num = 30; % 粒子数量
x_num = 3; % 变量个数
%定义个体学习因子和社会学习因子
init_c1 = 2.5;
final_c1 = 0.5;
init_c2 = 1;
final_c2 = 2.25;
w = 0.9;
iter = 100;
%粒子的最大速度以及上下界约束条件
vmax = [1.5 2.2 1.8];
x_lower = [0 0 0];
x_upper = [15 22 18];

%初始化粒子的位置和速度并计算相应的适应度
x = zeros(particle_num,x_num);
for i = 1: x_num
    x(:,i) = x_lower(i) + (x_upper(i)-x_lower(i))*rand(particle_num,1);
end
v = -vmax + 2*vmax .* rand(particle_num,x_num);
%计算初始化例子的适应度
  fit = zeros(particle_num,1);
for i = 1:particle_num
```

```
    fit(i) = fun(x(i,:));
end
%记录最佳位置
best_p = x;
best_ind = find(fit == min(fit), 1);
best_g = x(best_ind,:);
best_fittness = ones(iter,1);
for k = 1:iter
        %计算非对称性学习因子
        c1 = init_c1 + (final_c1 - init_c1)*k/iter;
        c2 = init_c2 + (final_c2 - init_c2)*k/iter;
    %更新每个粒子的速度
    for i = 1:particle_num
        v(i,:) = w*v(i,:) + c1*rand(1)*(best_p(i,:) - x(i,:)) + c2*rand(1)*(best_g
            - x(i,:));
        x(i,:) = x(i,:) + v(i,:);
        %防止粒子超过限制条件
        for j = 1: x_num
            if x(i,j) < x_lower(j)
                x(i,j) = x_lower(j);
            elseif x(i,j) > x_upper(j)
                x(i,j) = x_upper(j);
            end
        end
        %重新计算粒子的适应度
        fit(i) = fun(x(i,:));
        %判定粒子的适应度是否最优
        if fit(i) < fun(best_p(i,:))
           best_p(i,:) = x(i,:);
        end
        if  fit(i) < fun(best_g)
            best_g = best_p(i,:);
        end
    end
    %每次迭代过后更新粒子的最优适应度
    best_fittness(k) =fun(best_g);
end
disp('粒子群搜索得到的最优结果为: '); disp(best_g)
disp('需要花费的最少费用为: '); disp(fun(best_g))
```

问题四的代码:

```
%Question3.m文件

clc;
clear;
two_years=xlsread("D:\MATLAB\special_for_Matlab\2023_test\所有路线流量数据处理结果.
    xlsx");
all_data=xlsread("D:\MATLAB\special_for_Matlab\2023_test\predict_data.xlsx");
all_end9=xlsread("D:\MATLAB\special_for_Matlab\2023_test\data_end9.xlsx");
%各路线的阈值
max_produce=two_years(:,3);
%用一个月的平均值代替每天的
for i=1:size(all_end9,1)%平均运到5的货量
    all_end9(i,2)=sum(all_end9(i,3:end))/31;
```

```
end
for i=1:size(all_data,1)
    all_data(i,34)=sum(all_data(i,3:end))/31;
end
%第34列换成他还能承载多少货物
all_data(:,34)=max_produce;
%将跟5有关联的结点按照平均值降序排序
all_end9=sortrows(all_end9,-2);
cumulative_quantity=0;%货物累计量
change_num=0;%线路变化量
% 初始化粒子的位置和速度
x = zeros(n,narvs);
for i = 1: narvs
    x(:,i) = x_lower(i) + (x_upper(i)-x_lower(i))*rand(n,1);
end
v = -v_max + 2*v_max .* rand(n,narvs);
%计算粒子的适应度
fitness = zeros(n,1); % 初始化这n个粒子的适应度全为0
for i = 1:n
    fitness(i) = fun_1(x(i,:)); % 调用fun_1函数来计算适应度
end
best_x = x;
ind = find(fitness == min(fitness), 1);
best_g = x(ind,:);
for i=1:31
    for d = 1:max_iterations
        tem = best_g;
        for i = 1:n %对离子的速度和适应度进行更新
            f_i = fitness(i);
            f_avg = sum(fitness)/n;
            f_min = min(fitness);
            if f_i <= f_avg
                w = w_min + (w_max - w_min)*(f_i - f_min)/(f_avg - f_min);
            else
                w = w_max;
            end
            v(i,:) = factor * (w*v(i,:) + a1*rand(1)*(best_x(i,:) - x(i,:)) + a2*
                rand(1)*(best_g - x(i,:))); % 更新第i个粒子的速度
            x(i,:) = x(i,:) + v(i,:); %对第i个粒子的位置进行更新
            for j = 1: narvs %对那些超出范围的粒子进行更新
                if x(i,j) < x_lower(j)
                    x(i,j) = x_lower(j);
                elseif x(i,j) > x_upper(j)
                    x(i,j) = x_upper(j);
                end
            end
            fitness(i) = fun_1(x(i,:));
            if fitness(i) < fun_1(best_x(i,:))%判断是否需要更新粒子的最佳位置
               best_x(i,:) = x(i,:);
            end
            if  fitness(i) < fun_1(best_g)
                best_g = best_x(i,:);
            end
        end
        best_fitness(d) = fun_1(best_g);
```

```
        delta_fitness = abs(fun_1(best_g) - fun_1(tem)); % 计算相邻两次迭代适应度的
            变化量
    end
end
```

问题四的代码：

```
%fun_2.m文件

function [weight] = fun_2(data)
    %熵权法计算权值
    temp = zeros(1,size(data,2));
    for i = 1:size(data,2)
        x = data(:,i);
        p = x / sum(x);
        e = -sum(p .* mylog(p)) / log(size(data,1));
        temp(i) = 1- e;
    end
    %归一化，得到topsis运算得到的权值
    weight = temp ./ sum(temp);
end


%question_4_kmean.m

clc;clear;close all;
data=xlsread("D:\shuweibei\第四问_kmean++.xlsx");
%使用kmean++确定初始聚类中心
%聚类个数
cluster_num=4;
%产生一个初始聚类中心
init_1=randi([1,size(data,1)]);
cluster_center=zeros(cluster_num,2);
cluster_center(1,:)=data(init_1,:);
%初始化距离矩阵
distance=zeros(size(data,1),cluster_num)+999999;
%计算每个样本与当前已有聚类中心的最短距离
for i=1:cluster_num-1
    for j=1:i
        %将第j个中心复制size(data,1)份便于计算
        temp=repmat(cluster_center(j,1),size(data,1),1);
        %计算每个点到第j个中心的距离
        distance(:,j)=abs(temp(1)-data(:,1),2);
    end
    %选取第i+1个聚类中心
    min_distance=min(distance');
    rate=min_distance/sum(min_distance);
    cum_rate=cumsum(rate);
    %产生一个随机数
    rand_1=rand();
    newin=1;
    fitin=1;
    %使用轮赌法找到聚类中心
    while newin<=1
        if(rand_1)<cum_rate(fitin)
```

```
            cluster_center(i+1)=data(fitin);
            newin=newin+1;
        else
            fitin=fitin+1;
        end
    end
end

%进行聚类
[m,n]=size(data);
%定义聚类参数
iter_max=1000;
deta=0.001;
iter_num=0;
while(iter_num<iter_max)
    iter_num=iter_num+1;
    for i=1:cluster_num%计算每个数据点到每个聚类中心的距离
        distance=(data-repmat(cluster(i,1),m,1));
        new_distance(:,i)=abs(sum(distance'));
    end
    %找到当前点最近的聚类中心，并把他分配给当前的聚类中心
    [~,index_cluster]=min(new_distance');
    %更新聚类中心
    for j=1:cluster_num
       new_cluster(j,1)=mean(data(find(index_cluster==j),1));
    end
    %判断聚类中心是否变化
    if (sqrt(sum((new_cluster-cluster).^2))>deta)
        cluster=cluster_new;
    else
        break;
    end
end


%question_4_predict.m

clc
clear
import imblearn.over_sampling.SMOTE;
%全部数据
data=xlsread("D:\shuweibei\第四问数据.xlsx");
%要预测的391-410的数据
pre_data=xlsread("D:\shuweibei\第四问数据_2.xlsx");
%使用Smote算法对不均衡数据进行过采样
% 计算安静型、中等型、矛盾型类型的数量
class_num = size(unique(data(:, end)),1);
class_cnt = histc(data(:, 9), class_num);
max_cnt = max(class_cnt);
smote_data = [];
%进行smote采样
for i=1:max_cnt
    class_data = data(data(:, end) == unique(data(:, end))(i), :);
    sample= datasample(class_data,1);
    houxuan = nearestneighbour(sample', class_data', 'NumberOfNeighbours');
```

```
    houxuan(:,1) = [] ;
    rn=ceil(rand(1)*(size(houxuan,2)));
    select_ind = houxuan(:,rn);
    g = class_data(select_ind,:);
    alpha = rand(1);
    snew = sample(1,:) + alpha.*(g-sample(1,:));
    smote_data = [smote_data;snew];
end
%将使用smote过采样得到的结果和要预测的数据组合
data=[smote_data(:,1:8);pre_data];
data_temp=data;
%对数据进行标准化
normal=[data(:,1),data(:,4),data(:,6:8)];
normal_data = (normal - mean(normal)) ./ std(normal);
%将标准化的部分数据替换以及标准化的数据
data=[normal_data(1:660,1),data(1:660,2:3),normal_data(1:660,2),data(1:660,5),
    normal_data(1:660,3:5),smote_data(1:660,9)];
pre_data=[normal_data(661:680,1),data_temp(661:680,2:3),normal_data(661:680,2),
    data_temp(661:680,5),normal_data(661:680,3:5)];
%设置输入变量以及label
x_data = data(:,1:8);
label = data(:,9);
%%使用贝叶斯方法对随机森林进行参数优化
% 设置优化的参数
vars = [
    optimizableVariable('NumPredictorsToSample', [1, 8], 'Type', 'integer');
    optimizableVariable('MinLeafSize', [1, 40], 'Type', 'integer');
    optimizableVariable('NumTrees', [10, 100], 'Type', 'integer');
    optimizableVariable('MaxNumSplits', [2, 10], 'Type', 'integer');
    ];
% 定义目标函数
minfn = @(T)kfoldLoss(fitcensemble(X_train, Y_train, 'Method', 'Bag', '
    NumLearningCycles', T.NumTrees, 'Learners', ...
    templateTree('MaxNumSplits', T.MaxNumSplits, 'NumPredictorstoSample', T.
        NumPredictorsToSample, 'MinLeafSize', T.MinLeafSize)));
% 使用贝叶斯优化来搜索最佳参数
results = bayesopt(minfn, vars, 'Verbose', 1, 'AcquisitionFunctionName', 'expected
    -improvement-plus');

% 提取最优参数
best_NumPredictorsToSample = results.XAtMinEstimatedObjective.
    NumPredictorsToSample;
best_MinLeafSize = results.XAtMinEstimatedObjective.MinLeafSize;
best_NumTrees = results.XAtMinEstimatedObjective.NumTrees;
best_MaxNumSplits = results.XAtMinEstimatedObjective.MaxNumSplits;
% 使用最优参数创建随机森林模型
rf = TreeBagger(best_NumTrees, x_data,label, 'Method', 'classification', '
    MinLeafSize', best_MinLeafSize, ...
    'NumPredictorsToSample', best_NumPredictorsToSample, 'MaxNumSplits',
        best_MaxNumSplits);
% 生成随机下标
indices = randperm(size(x_data,1));
% 20%的数据用于验证
test_index = indices(1:round(0.2 *size(x_data,1)));
data_test = x_data(te_index,:);
```

```
label_test = label(te_index);
% 预测测试集标签
y_pred = predict(rf, data_test);
% 计算准确率
right_rate = sum(str2double(y_pred) == label_test) / size(label_test,1);
disp(['验证集的准确率为: ' num2str(right_rate)]);
% 计算混淆矩阵以及各种指标
y_pred=str2double(y_pred);
y_true=label_test;
%创建混淆矩阵
confusion_matrix = confusionmat(y_true, y_pred);
confusionchart(confusion_matrix, {'差', '中','差','优'});
num_class = size(confusion_matrix, 1);
precision = zeros(num_class, 1);
recall = zeros(num_class, 1);
f1_score = zeros(num_class, 1);
for i=1:num_class
    TP = confusion_matrix(i,i);
    FP = sum(confusion_matrix(:,i)) - TP;
    FN = sum(confusion_matrix(i,:)) - TP;
    precision(i) = TP / (TP + FP);
    recall(i) = TP / (TP + FN);
    f1_score(i) = 2 * precision(i) * recall(i) / (precision(i) + recall(i));
end
% 输出每个类别的精度、召回率和F1分数
disp('每一类的精度: '), disp(precision)
disp('每一类的真正类率: '), disp(recall)
disp('每一类的F1分数: '), disp(f1_score)
% 预测的391-410的数据
pre_label= str2double(predict(rf, pre_data))

%question_4_topsis.m

clc;clear;
data=xlsread("D:\shuweibei\第四问_topsis.xlsx");
[n,m] = size(data);
%首先对区间型数据正向化
num_1 = size(data(:,1),1);
M = max([13-min(x),max(x)-18]);
posit_x = zeros(num_1,1);
for i = 1: num_1
   if data(i,1) < a
       posit_x(i) = 1-(13-data(i,1))/M;
   elseif data(i,1) > b
       posit_x(i) = 1-(data(i,1)-18)/M;
   else
       posit_x(i) = 1;
   end
end
data(:,1)=posit_x;
%对极小型数据正项化
data(:,2)=max(data(:,2)) - data(:,2);
% 对正向化后的数据进行标准化
normal_data = data ./ repmat(sum(data.*data) .^ 0.5, n, 1);
%通过熵权法确定权值
```

```
sq_weight=fun_2(nornal_data);
%计算得分
postive_D = sum([(normal_data - repmat(max(normal_data),n,1)) .^ 2 ] .* repmat(
    weight,n,1) ,2) .^ 0.5;
negative_D = sum([(normal_data - repmat(min(normal_data),n,1)) .^ 2 ] .* repmat(
    weight,n,1) ,2) .^ 0.5;
score = negative_D ./ (postive_D+negative_D);
disp('最后的得分为：')
[sorted_S,index] = sort(score ,'descend')
```

问题五的代码：

```
%question_5.m

clear; clc
%设置粒子群的初始参数
particle_num = 50; % 粒子数量
x_num = 3; % 变量个数
%定义个体学习因子和社会学习因子
init_c1 = 2.5;
final_c1 = 0.5;
init_c2 = 1;
final_c2 = 2.25;
w = 0.9;
iter = 100;
%粒子的最大速度以及上下界约束条件
vmax = [1.5 2.2 1.8];
x_lower = [0 0 0];
x_upper = [15 22 18];

%初始化粒子的位置和速度并计算相应的适应度
x = zeros(particle_num,x_num);
for i = 1: x_num
    x(:,i) = x_lower(i) + (x_upper(i)-x_lower(i))*rand(particle_num,1);
end
v = -vmax + 2*vmax .* rand(particle_num,x_num);
%计算初始化例子的适应度
  fit = zeros(particle_num,1);
for i = 1:particle_num
    fit(i) = fun_3(x(i,:));
end
%记录最佳位置
best_p = x;
best_ind = find(fit == min(fit), 1);
best_g = x(best_ind,:);
best_fittness = ones(iter,1);
for k = 1:iter
        %计算非对称性学习因子
        c1 = init_c1 + (final_c1 - init_c1)*k/iter;
        c2 = init_c2 + (final_c2 - init_c2)*k/iter;
    %更新每个粒子的速度
    for i = 1:particle_num
        v(i,:) = w*v(i,:) + c1*rand(1)*(best_p(i,:) - x(i,:)) + c2*rand(1)*(best_g
            - x(i,:));
```

```
        x(i,:) = x(i,:) + v(i,:);
        %防止粒子超过限制条件
        for j = 1: x_num
            if x(i,j) < x_lower(j)
                x(i,j) = x_lower(j);
            elseif x(i,j) > x_upper(j)
                x(i,j) = x_upper(j);
            end
        end
        %重新计算粒子的适应度
        fit(i) = fun_3(x(i,:));
        %判定粒子的适应度是否最优
        if fit(i) < fun_3(best_p(i,:))
           best_p(i,:) = x(i,:);
        end
        if  fit(i) < fun_3(best_g)
            best_g = best_p(i,:);
        end
    end
    %每次迭代过后更新粒子的最优适应度
    best_fittness(k) =fun_3(best_g);
end
disp('粒子群搜索得到的最优结果为: '); disp(best_g)
disp('需要花费的最少费用为: '); disp(fun_3(best_g))
```