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		CM2301536

母亲身心健康对婴儿成长的影响研究

摘要

本文根据母亲身体和心理指标以及婴儿行为特征和睡眠质量的数据，利用**Spearman 相关性分析**、**XGBoost 预测模型**、**非线性规划模型**、**Topsis-分位数评级模型**、**GBDT 分类模型**等方法完成母亲身心健康对婴儿成长的影响研究。在识别并处理母婴指标中的异常数据后，通过统计方法探究母亲身体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，并找出其中的规律。随后分别建立母亲身体和心理指标与婴儿行为特征以及睡眠质量的关联模型，以实现通过母亲身体和心理指标对婴儿行为特征以及睡眠质量的预测。最后，通过构建非线性规划模型，计算出能实现不同的婴儿行为特征和睡眠质量的治疗目标的同时使治疗费用降到最低的最优治疗方案及相应最低治疗费用。

针对问题一，本文首先进行数据清洗，识别并处理附件数据中的异常值，其次对婴儿行为特征指标和婴儿整晚睡眠时间指标进行量化处理。在完成数据预处理后，对各数据指标是否服从正态分布进行了检验，发现部分数据不符合正态分布则不适用于 Person 相关分析，故采用 **Spearman 相关分析**的方法进行检验，结果发现母亲身体和心理指标对婴儿行为特征指标和睡眠质量存在显著影响，随后通过热力图将这种关系可视化并归纳出其影响规律。

针对问题二，本文通过训练 **XGBoost 预测模型**来刻画婴儿的行为特征与母亲的身体指标与心理指标的关系，并引入逻辑回归模型、KNN 分类模型、随机森林模型、GBDT 分类模型等多个预测模型与 XGBoost 预测模型的预测优良性进行比对，以此来验证模型选择的准确性和可靠性，最后通过训练完成的 XGBoost 预测模型进行婴儿行为特征的预测。

针对问题三，本文将其转化为数学规划问题，结合问题二的 XGBoost 预测模型构建了**行为特征治疗策略非线性规划模型**，并给出了在能够实现婴儿行为特征向期望行为特征转变的同时将治疗费用降到最低的最优治疗方案及与之相对的最低治疗费用。

针对问题四，本文通过建立 **Topsis-分位数评级模型**对婴儿的睡眠质量进行优、良、中、差四分类的综合评判，再通过对逻辑回归模型、KNN 分类模型、随机森林模型、GBDT 分类模型、XGBoost 预测模型等多个预测模型的预测优良性进行比对，选取出**GBDT 分类模型**作为婴儿综合睡眠质量的最优预测模型，训练 GBDT 分类模型完成对婴儿综合睡眠质量与母亲的身体指标、心理指标的关联关系的刻画，最后实现对婴儿的综合睡眠质量的预测。

针对问题五，延续问题三的思路，结合问题四的 GBDT 分类模型，构建**行为特征&睡眠质量治疗策略非线性规划模型，**并给出了在能够实现婴儿行为特征和睡眠质量向期望的行为特征和睡眠质量转变的同时将治疗费用降到最低的最优治疗方案及与之相对的最低治疗费用。

关键词：Spearman 相关分析 XGBoost 预测模型 非线性规划模型 Topsis-分位数评级模型 GBDT 分类模型

一、 问题背景与分析

1.1 背景分析

母亲作为婴儿成长过程中的重要支持者，其身心健康状态对婴儿的成长和发展有着深远的影响。母亲不良的身心健康状况可能对婴儿的生理和心理的发展产生不良影响。深入研究母亲身心健康与婴儿成长之间的关系对于制定相应的干预措施和提供支持至关重要。

1.2 问题重述

根据题目所提供的数据资料，建立数学模型解决以下 5 个问题：

1.对附件中的数据进行统计分析，探究母亲的身体指标和心理指标是否对婴儿的行为特征和睡眠质量产生影响，若存在则将这种关系可视化并归纳出其影响规律。

2.构建婴儿的行为特征与母亲的身体指标和心理指标的关联模型，并利用该模型对 20 组婴儿行为特征信息进行预测，判断他们属于哪种类型（安静型、中等型、矛盾型）。

3.建立模型根据不同的期望行为特征干预效果，确定出相应治疗费用最低的行为特征治疗方案，并估计出实施这些方案所需的最少治疗费用。并以 238 号婴儿为例给出具体治疗方案及治疗费用。

4.对婴儿的睡眠质量进行综合评判，并建立婴儿综合睡眠质量与母亲的身体指标和心理指标之间的关联模型，并通过模型对最后 20 组婴儿的综合睡眠质量进行推断。

5.在问题三的基础上，兼顾不同的期望睡眠质量干预效果，确定出相应治疗费用最低的行为特征&睡眠质量治疗方案，比较与原先行为特征治疗方案的区别，并给实施这些方案所需的最少治疗费用。同样以 238 号婴儿为例给出具体治疗方案及治疗费用。

二、 问题分析

2.1 对于问题一的分析

问题一要求对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进行统计分析。首先进行数据清洗，识别出附件数据中异常值并进行筛选处理，其次对婴儿行为特征指标和婴儿整晚睡眠时间指标进行量化处理。在对数据进行预处理后，首先检验各数据指标是否服从正态分布，若服从正态分布则采用 Pearson 相关系数探究其规律，如若不服从则采用 spearman 相关系数的方法探究其规律，若存在关系可最后通过热力图来可视化分析。

2.2 对于问题二的分析

根据题目所给信息，将本问题转化为有监督的预测问题，通过训练适当的模型来刻画婴儿的行为特征与母亲的身体指标与心理指标的关系。可以引入多个预测模型进行预测优良性的比对，选取最优的预测模型进行训练，并利用训练好的预测模型对母亲身体指标和心理指标特征的重要性进行评分，最后得出婴儿行为特征预测的结果。

2.3 对于问题三的分析

根据题目所给信息，将本问题转化为数学规划问题，构建行为特征治疗策略数学规划模型，根据不同的期望行为特征干预效果，确定出相应治疗费用最低的行为特征治疗方案，并估计出实施这些策略所需的最少治疗费用。将行为特征为矛盾型的 238 号婴儿作为行为特征治疗对象，运用 Python 软件编写程序求解，得出最优治疗方案。

2.4 对于问题四的分析

将本问题转化为评价-预测问题，先通过 Topsis-分位评级模型对婴儿的睡眠质量进行优、良、中、差四分类的综合评判，再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对婴儿的综合睡眠质量进行预测。在得到婴儿综合睡眠质量的评级后我们采用均方根误差（RMSE）、平均绝对误差（MAE）和准确率作为评价指标，对逻辑回归模型、KNN 分类模型、随机森林模型、GBDT 分类模型和 XGBoost 模型等多种预测模型进行对婴儿睡眠质量预测效果的比较分析，以选出可信度最高的预测模型去预测婴儿睡眠质量。

2.5 对于问题五的分析

问题五要求在问题三的基础上兼顾不同的期望睡眠质量干预效果，因此问题五的行为特征&睡眠质量治疗策略数学规划模型是在问题三的行为特征治疗策略数学规划模型基础上的进一步深化。首先仍然是确立约束条件和构建目标函数，然后再对行为特征睡眠质量治疗策略数学规划模型进行求解。

三、 模型假设

1. CBTS、EPDS、HADS 得分真实反映患病程度，且治疗效果影响得分的变化程度最小为 0.1；
2. 附件提供的数据真实可信，是经过真实的实验所得的数据，且不存在人为的删改和破坏。
3. 假设除了本文提到的因素，没有其它因素影响母亲和婴儿各项指标之间的关系。

四、 符号说明

序号	符号	意义
1	BC_i	第 i 号婴儿的行为特征
2	ST_i	第 i 号以秒为单位计量的整晚睡眠时间
3	$cost_j$	j=1, 2, 3，分别表示 CBTS、EPDS、HADS 的单项治疗费用
4	$heat_j$	j=1, 2, 3，分别表示 CBTS、EPDS、HADS 的单项治疗效果(得分值)
5	BC_i^*	第 i 组经治疗后预测的婴儿行为特征
6	BC_desire	期望行为特征
7	L_i^*	第 i 组经治疗后预测的婴儿睡眠质量评级
8	L_desire	期望睡眠质量评级

五、 问题一的求解

根据题目所给信息，将本问题转化为对变量间关联关系的分析，在对数据进行清洗后，进行正态性的判别，从而相应选取 Pearson 相关系数或 Spearman 相关系数进行变量间相关性的分析，并通过热力图将其可视化。

5.1 数据预处理

5.1.1 数据清洗

利用 Python 软件对附件相关数据进行异常值的识别并进行清洗，结果发现母亲婚姻状况指标和婴儿整晚睡眠时间指标存在异常值，并且第 391-410 行存在数据空缺。根据题意，将附件中婚姻状况指标值大于 2 的数据行和婴儿整晚睡眠时间指标为 99:99 的数据行（共 10 行数据）删去，将仅预测时使用的编号 391-410 的数据单独取出，对剩余 380 组样本进行后续分析。

5.1.2 指标量化处理

1.对三种婴儿行为特征指标：安静型、中等型、矛盾型进行量化处理，结果如下：

$$BC_i = \begin{cases} 安静型 = 1 \\ 中等型 = 2 \\ 矛盾型 = 3 \end{cases}$$

其中BC_i表示第 i 号婴儿的行为特征。

2.将第 i 号整晚睡眠时间（时：分：秒）全部转化为以秒为单位的数值ST_i。

5.2 母亲身心指标对婴儿行为特征和睡眠质量的影响分析

通过作图可以发现部分母亲身体和心理指标以及婴儿行为特征和睡眠质量的数据分布并不服从正态分布，如图 5-1 所示。

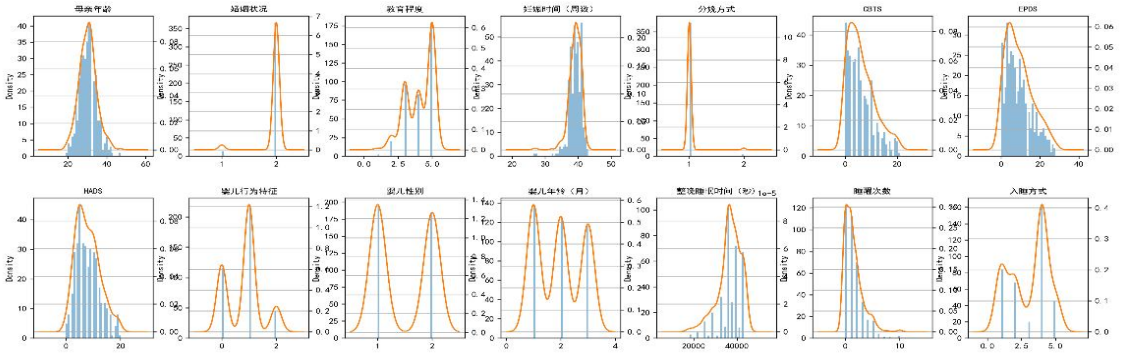

图 5-1 母亲身体和心理指标以及婴儿行为特征和睡眠质量的数据分布

因此采用可分析不服从正态分布的变量关联性的 Spearman 相关系数来探究母亲的身体指标和心理指标是否对婴儿的行为特征和睡眠质量产生影响[1]。通过对 Spearman 相关系数表(详见附录一)进行整理得到下表 5-1：

表 5-1 母亲身体和心理指标中对婴儿行为特征与睡眠质量显著影响的变量

显著性水平 α	婴儿行为特征	婴儿睡眠质量		
		整晚睡眠时间	睡醒次数	入睡方式
0.01	EPDS	EPDS	–	–
0.05	EPDS、CBTS、HADS	EPDS、CBTS、HADS	EPDS	–
0.1	EPDS、CBTS、HADS、母亲年龄	EPDS、CBTS、HADS	EPDS、妊娠时间	母亲年龄

对 Spearman 相关系数表(详见附录一)进行可视化得到下图 5-2：

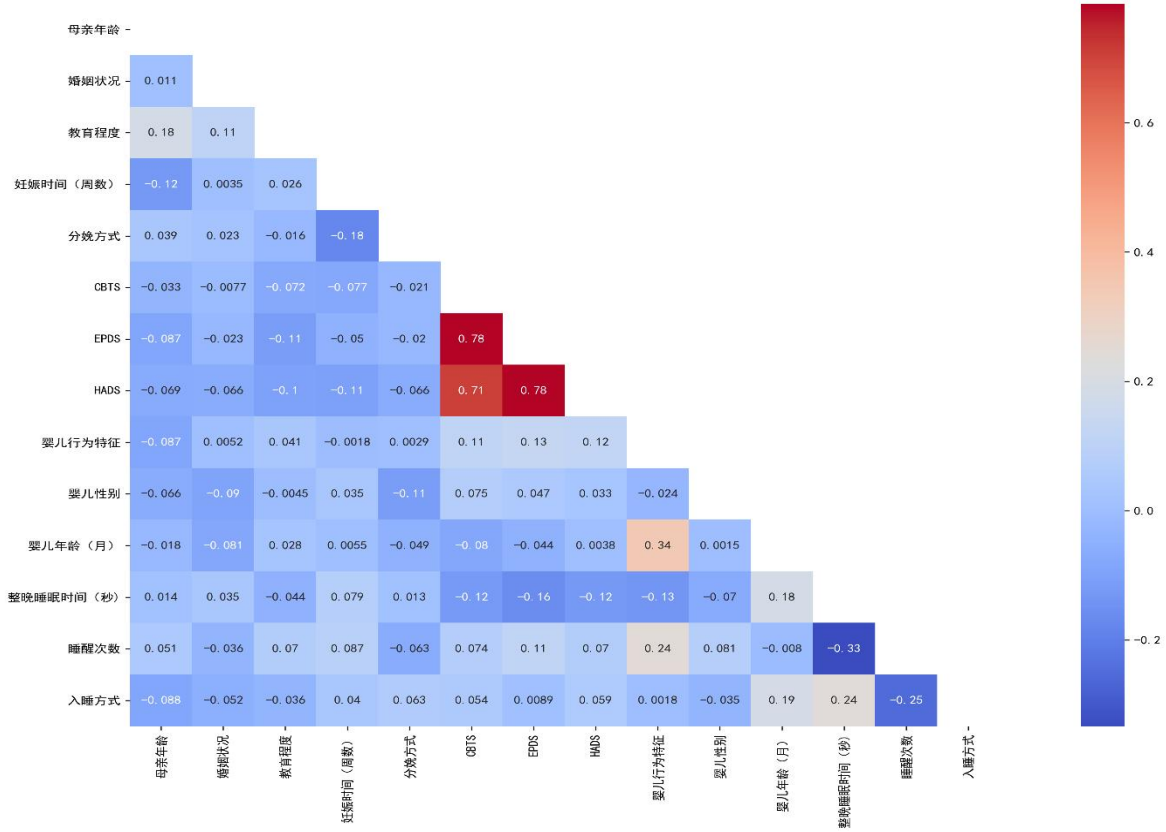

图 5-2 Spearman 相关系数热力图

由上表和上图可知，在 0.1 的显著性水平上，EPDS、CBTS、HADS 和母亲年龄均会对婴儿行为特征产生显著影响，其中母亲年龄与婴儿行为方式呈显著负相关，即母亲年龄越大婴儿越呈现安静向的行为特征，而 EPDS、CBTS、HADS 均与因而行为方式呈显著正相关，即母亲 EPDS、CBTS、HADS 得分越高，说明母亲心理指标水平越低，则婴儿越呈现矛盾向的行为特征。综上，母亲的身体指标和心理指标会对婴儿行为方式产生显著影响。

在 0.1 的显著性水平上，EPDS、CBTS、HADS 均与婴儿睡眠时间呈显著负相关，即母亲 EPDS、CBTS、HADS 得分越高，说明母亲心理指标水平越低，则婴儿相对睡眠时间越少，说明婴儿睡眠质量越差；EPDS、妊娠时间与婴儿睡醒次数呈显著正相关，即母亲 EPDS 得分越高、妊娠时间越长，则婴儿睡醒次数越多，说明婴儿睡眠质量越差；母亲年龄与婴儿入睡方式呈显著负相关，即母亲年龄越大，婴儿入睡越需要母亲的人为干预。综上，母亲的身体指标和心理指标会对婴儿睡眠质量产生显著影响。

六、 问题二模型的建立与求解

根据题目所给信息，将本问题转化为有监督预测问题，通过训练适当的模型来刻画婴儿的行为特征与母亲的身体指标与心理指标的关系，并通过训练完成的模型进行婴儿行为特征的预测。

6.1 XGBoost 模型的建立

集成决策树类的模型是机器学习模型中可以进行特征重要性度量的一类模型，其中 XGBoost 模型自提出以来备受关注，不仅众多学者对其展开深入研究与改进[2,3]，而且在工业界取得了不错的成果[4,5]。相较于传统的统计模型，XGBoost 无论在分类还是回归问题中均能取得较好效果[6]。XGBoost 算法的基本思想就是，不断地进行特征分裂来生长

一棵树，每一轮学习一棵树，其实就是去拟合上一轮模型的预测值与实际值之间的残差，当我们训练完成得到 K 棵树时，我们要预测一个样本的分数，其实就是根据这个 样本的特征，在每棵树中会落到对应的一个叶子节点，每个叶子节点就对应一个分数， 最后只需将每棵树对应的分数加起来就是该样本的预测值。

构建 XGBoost 算法的模型如式 6.1 所示，给定数据集$D = \{(x_i, y_i)\}$，XGBoost 进行 additive training，学习 K 棵树并对样本进行预测。

$$\tilde{y} = \Phi(x_i) = \sum_{k=1}^{K} f_k(x_i), f_k \in \Psi \tag{6.1}$$

其中，Ψ是假设空间，$f_k(x_i)$是回归树，数学表达式如式 6.2 所示。

$$\Psi = \{f(x) = \}(q: R^m \to T, w \in R^T) \tag{6.2}$$

其中，$q(x)$表示样本x分到某个叶子节点上，w是叶子节点的分数，所以$w_{q(x)}$表示 回归树对样本的预测值。构建模型函数空间中的目标函数为：

$$L(\Phi) = \sum_i l(\tilde{y}_l, y_i) + \sum_k \Omega(f_k) \tag{6.3}$$

其中，正则化项$\Omega(f) = \gamma T + \frac{1}{2}\lambda||w||^2$，$T$为叶子节点个数，$w$为叶子节点分数。 将目标函数使用二阶泰勒公式展开，经过第 t 次迭代后，模型的预测等于前 t-1 次 模型预测加第 t 棵树的预测，如式 6.4 所示。

$$\tilde{y}_l^{(t)} = \tilde{y}_l^{(t-1)} + f_l(x_i) \tag{6.4}$$

这时目标函数式 6.4 可以写作式 6.5：

$$L^{(t)} = \sum_{i=1}^{n} l\left(y_i, \tilde{y}_i^{(t-1)} + f_t(X_i)\right) + \Omega(f_t) \tag{6.5}$$

其中，$y_i, \tilde{y}_i^{(t-1)}$都已知，模型要学习的只有第 t 棵树f_t。 将误差函数在$\tilde{y}_l^{(t-1)}$处进行二阶泰勒展开，如 6.6 式所示：

$$L^{(t)} = \sum_{i=1}^{n} [l(y_i, \tilde{y}_i^{(t-1)} + g_i f_t(X_i) + \frac{1}{2} h_i f_t^2(X_i))] + \Omega(f_t) \tag{6.6}$$

其中，

$$\begin{aligned} g_i &= \partial_{\tilde{y}^{(t-1)}} l(y_i, \tilde{y}_l^{(t-1)}) \\ h_i &= \partial^2_{\tilde{y}^{(t-1)}} l(y_i, \tilde{y}_l^{(t-1)}) \end{aligned} \tag{6.7}$$

把公式中的常数项去掉，得到式 6.9：

$$L^{(t)} = \sum_{i=1}^{n} [l(g_i f_t(X_i) + \frac{1}{2} h_i f_t^2(X_i))] + \Omega(f_t) \tag{6.8}$$

把f_t，$\Omega(f_t)$写成树结构的形式，即把式 6.9 带入目标函数 6.8 中，得到式 6.10。

$$\begin{cases} f(x) = w_{q(x)} \\ \Omega(f) = \gamma T + \frac{1}{2}\lambda ||w||^2 \end{cases} \tag{6.9}$$

$$\tilde{L}^{(t)} = \sum_{i=1}^{n} [g_i w_{q(x_i)} + \frac{1}{2} h_i w_{q(x_i)}^2] + \gamma T + \frac{1}{2}\Sigma_{j=1}^{T} w_j^2 \tag{6.10}$$

定义 每个叶节点j上的样本集合为$I_j = \{i|q(x) = j\}$，则目标函数可以写成按叶节点累加的形式，如式 6.11 所示。

$$\begin{aligned} \tilde{L}^{(t)} &= [(\sum_{i\in I_j} g_i) w_j + \frac{1}{2}((\sum_{i\in I_j} h_i + \lambda) w_j^2)] + \gamma T \\ &= \sum_{j=1}^{T} [G_j w_j + \frac{1}{2}(H_j + \lambda) w_j^2] + \gamma T \end{aligned} \tag{6.11}$$

确定树的结构(即确定$q(x)$),为了使目标函数最小，可以令其导数为0，解得每个叶 节点的最终预测分数如式 6.13 所示。

$$w_j^* = -\frac{G_j}{H_j + \lambda} \tag{6.12}$$

将式 6.10 代入目标函数，得到最小损失如式 6.13 所示。

$$\tilde{L}^* = -\frac{1}{2}\Sigma_{j=1}^{T} \frac{G_j^2}{H_j + \lambda} + \gamma T \tag{6.13}$$

最终，算法不断迭代优化，目标函数降到最小就完成了模型的训练，其预测原理如图 6-1 所示，利用训练完成的模型进行特征筛选。

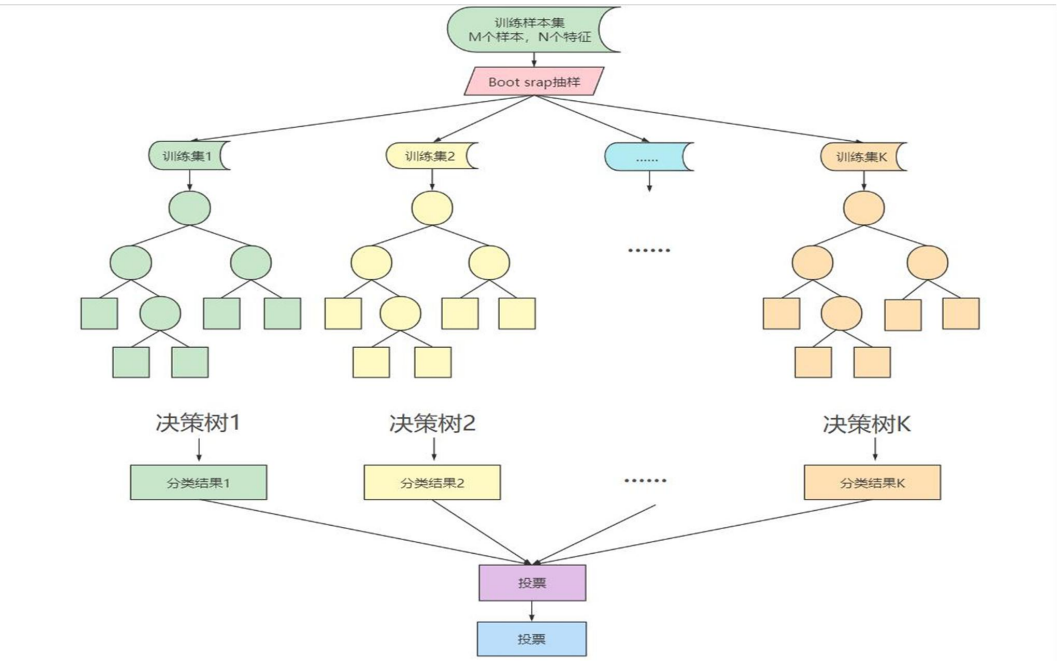

图 6-1 XGBoost 预测原理示意图

与很多人工智能模型不同的是，XGBoost 被创建后，可以相对直接地得到每个特征的重要性得分。一般来说，重要性分数衡量了该特征在模型构建中的价值。一个特征被越多的使用，它的重要性就相对越高。用xi表示第i个特征，则其得分的计算公式如式 6.14 所示。

$$\text{Score}(x_i) = \sum_{t=1}^{n} \frac{\text{Score}_t(x_i)}{n} \quad (6.14)$$

其中，n为决策树的个数，t为决策树的编号。

6. 2 XGBoost 模型的参数设置与特征重要性得分

我们将所有特征输入到 XGBoost 模型中，计算得到特征重要度得分。进一步依据 特征的得分，筛选出对于结果影响程度较大的特征，并用于后续 20 名婴儿行为特征的预测。XGBoost 模型的参数设置如表 6-1 所示。

表 6-1 模型的参数设置

模型	参数描述	参数值
XGBoost	XGBoost 中决策树的个数	525
	XGBoost 学习率	0. 2
	XGBoost 每个决策树的最大深度	5
	XGBoost 子样本的比例	0. 6
	XGBoost 每棵树随机选取的特征的比例	0. 9
	XGBoost 损失阈值	0. 2
	XGboost 的 L1 的正则项参数	0. 05
	XGboost 的 L2 的正则项参数	1

利用训练好的 XGBoost 模型对母亲身体指标和心理指标特征的重要性进行评分，绘制条形图如图 6-2 所示。

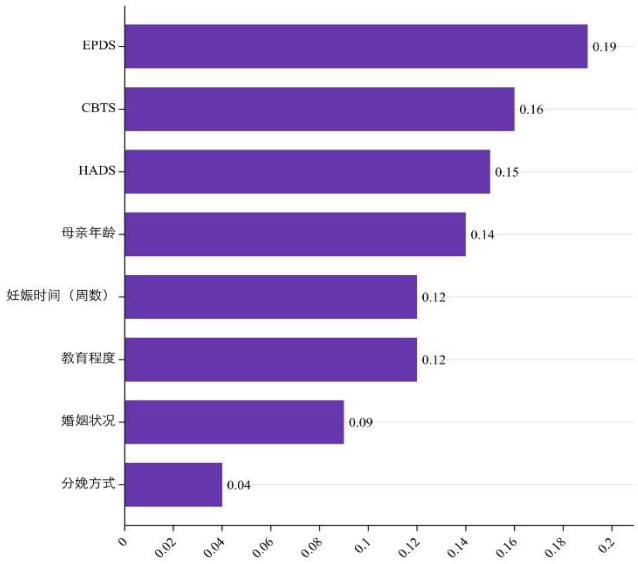

图 6-2 特征的重要性得分

由图 6-2 我们可以得知预测婴儿行为特征重要性最高的前四个指标分别为 EPDS、CBTS、HADS 和母亲年龄，对于问题一中该四个指标对婴儿行为特征的显著影响得到验证。

6.3 XGBoost 模型预测有效性评估

6.3.1 模型评估指标

为证明我们所构建的模型的有效性，我们采用均方根误差（Root Mean Squard Error，RMSE）、平均绝对误差（Mean Absolute Error，MAE）和准确率作为评价指标[7]。其中，均方根误差 RMSE 评价指标是用来衡量观测值同真实值之间的偏差；平均绝对误差 MSE 能更好地反映预测值误差的实际情况，值越小说明预测模型描述的实验数据就具有更好的精确度。准确率介于 0-1 之间，越接近 1，回归拟合效果越好。

均方根误差 RMSE 的公式如式 6.15 所示。

$$RMSE = \sqrt{\frac{1}{m}\Sigma_{i=1}^{m}(f_i - y_i)^2} \tag{6.15}$$

其中，m为实验的次数，f_i是模型预测值，y_i是样本真实值。平均绝对误差 MAE 的公式如式 6.16 所示。

$$MAE = \frac{1}{m}\Sigma_{i=1}^{m}|f_i - y_i| \tag{6.16}$$

其中，m为实验的次数，f_i是模型预测值，y_i是样本真实值。R^2的公式如式 6.17 所示：

$$\text{准确率}(accuracy) = \frac{1}{n}\sum_{i=1}^{n} I(f(x_i) = y_i) \tag{6.17}$$

其中，当$f_i = y_i$时$I(f(x_i) = y_i)$的值为 1，否则为 0.

6.3.2 模型有效性分析

为了证明所构建模型的有效性，我们同时对逻辑回归模型、KNN 分类模型、随机森林模型、GBDT 分类模型等多种预测模型进行预测效果的比较分析[8–11]。通过使用十折交叉验证方法，重复 10 次实验的平均结果如表 6-2 所示。

表 6-2 各模型婴儿行为特征的预测有效性比较分析

模型	RMSE	MAE	准确率
逻辑回归模型	0. 6689	0.4211	0.5789
KNN 分类模型	0. 8111	0.5526	0.5021
随机森林模型	0.7255	0. 4737	0.5526
GBDT 分类模型	0.7255	0.4211	0.6316
XGBoost 分类模型	0.6571	0.3847	0.6875

综合 RMSE、MAE 和准确率指标，可以得出 XGBoost 预测模型相对其他模型预测准确性更高的结论。因此我们认为：使用 XGBoost 模型预测得到的婴儿行为特征可信度较高。

6.4 问题二的求解

采用 XGBoost 模型对最后 20 组婴儿的行为特征进行预测，所得结果如下表所示：

表 6-3 最后 20 组婴儿行为特征预测结果

编号	婴儿行为特征	编号	婴儿行为特征
391	中等型	401	中等型
392	中等型	402	中等型
393	安静型	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	中等型	407	中等型
398	中等型	408	中等型
399	中等型	409	中等型
400	中等型	410	中等型

七、 问题三模型的建立与求解

根据题目所给信息，将本问题转化为数学规划问题，构建行为特征治疗策略数学规划模型，为婴儿行为特征治疗策略的制定提供科学依据。

7.1 行为特征治疗策略数学规划模型的建立

根据题目所给出的信息，可将问题三转化为数学规划问题，根据不同的期望行为特征干预效果，确定出相应治疗费用最低的行为特征治疗方案，并估计出实施这些策略所需的最少治疗费用。

7.1.1 约束条件的构建

由题可知 CBTS、EPDS、HADS 的治疗费用相对于患病程度的变化率均与治疗费用呈正比，则有：

$$\frac{dcost_j}{dheat_j} = k_j cost_j，\ j = 1，\ 2，\ 3 \tag{7.1}$$

则解微分方程得：

$$cost_j = C_j e^{k_j heat_j}，\ j = 1，\ 2，\ 3 \tag{7.2}$$

分别代入两个分数对应的治疗费用可解得：

$$\begin{aligned} \text{cost}_1 &= 200e^{0.8811\text{heat}_1} \\ \text{cost}_2 &= 500e^{0.6649\text{heat}_2} \\ \text{cost}_3 &= 300e^{0.7459\text{heat}_3} \end{aligned} \tag{7.3}$$

同时，由于治疗方案需要使治疗对象的行为特征转变为到期望行为特征，因此有：

$$BC_i^* = BC\_desire \tag{7.4}$$

其中，BC_i^*是第 i 组母亲经治疗改变 CBTS、EPDS、HADS 后用问题二中的 XGBoost 模型预测的婴儿行为特征，BC_desire为期望行为特征。

则行为特征治疗策略数学规划模型总的约束条件为：

$$\text{s.t.}\begin{cases} BC_i^* = BC\_desire \\ \text{cost}_1 = 200e^{0.8811\text{heat}_1} \\ \text{cost}_2 = 500e^{0.6649\text{heat}_2} \\ \text{cost}_3 = 300e^{0.7459\text{heat}_3} \\ 0 \le \text{heat}_1^i \le EPDS^i \\ 0 \le \text{heat}_2^i \le \text{CBTS}^i \\ 0 \le \text{heat}_3^i \le \text{HADS}^i \end{cases} \tag{7.5}$$

7.1.2 目标函数的构建

由题意可知，要通过治疗实现当前婴儿行为特征到期望行为特征的转变，同时要尽可能的减少所花费的治疗费用，由此可知目标函数：

$$min \sum_{k=1}^{3} cost_k \tag{7.6}$$

整理上述条件，可得行为特征治疗方案数学规划模型：

目标函数：

$$min \sum_{k=1}^{3} cost_k$$

约束条件：

$$\text{s.t.}\begin{cases} BC_i^* = BC\_desire \\ \text{cost}_1 = 200e^{0.8811\text{heat}_1} \\ \text{cost}_2 = 500e^{0.6649\text{heat}_2} \\ \text{cost}_3 = 300e^{0.7459\text{heat}_3} \\ 0 \le \text{heat}_1^i \le EPDS^i \\ 0 \le \text{heat}_2^i \le \text{CBTS}^i \\ 0 \le \text{heat}_3^i \le \text{HADS}^i \end{cases}$$

7.2 行为特征治疗策略数学规划模型的求解

选取 238 号婴儿作为行为特征治疗对象。基于假设 1 和以上列出的目标函数与约束条件，运用 Python 软件编写程序求解，可得出最优治疗方案如表 7-1 所示：

表 7-1 238 号婴儿最优治疗方案

期望婴儿行为特征	治疗后母亲心理指标		单项治疗费用（元）	总治疗费用（元）
中等型	CBTS	12	2811.90	22454.67
	EPDS	14	7145.29	
	HADS	17	12497.48	
安静型	CBTS	9	39534.04	313480.77
	EPDS	12	27011.23	
	HADS	13	246935.50	

由上表可知，若要将 238 号婴儿的行为特征从矛盾型转变为中等型，治疗费用最低的治疗方案是将母亲的 CBTS、EPDS、HADS 分别治疗到 12、14 和 17 分的水平以提高母亲的心理健康水平，此时最低的总治疗费用为 22454.67 元。

若要将 238 号婴儿的行为特征从矛盾型转变为安静型，则需将母亲的 CBTS、EPDS、HADS 分别治疗到 9、12 和 13 分的水平以提高母亲的心理健康水平，此时的总治疗费用为 313480.78 元。

八、 问题四模型的建立与求解

根据题意，将本问题转化为评价-预测问题，先通过 Topsis-分位评级模型对婴儿的睡眠质量进行优、良、中、差四分类的综合评判，再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对婴儿的综合睡眠质量进行预测。

8.1 Topsis−分位数评级模型的建立

Topsis-分位数评级模型先通过 TOPSIS 模型为每个对象进行评分并排序，随后以分位数为界限将每个对象进行类别划分，使每个类别中所含对象数基本相等，该方法不易受到极端值与主观划分的影响[12]。Topsis−分位数评级模型的建立步骤如下：

步骤 1：根据评价对象构造初始矩阵。

设由 m 个评价对象与 n 个评价指标建立起来的矩阵：

$$X = (x_{ij})_{m\times n} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{1n} & x_{2n} & \cdots & x_{nn} \end{bmatrix}, (i = 1,2\ldots, m; j = 1,2\ldots n) \quad (8.1)$$

其中，X 为初始矩阵，x_{ij}为第 i 个评价对象的第 j 个评价指标数据。

步骤 2：标准化决策矩阵（最小−最大标准化）

由于各指标的量纲，数量级均有差异，为消除量纲不同对评价结果的影响，需要对各指标进行标准化处理，本文采用极值法对数据进行处理。

当为正向指标时：

$$v_{ij} = \frac{x_{ij} - min\{x_{1j}, x_{2j}, \cdots x_{nj}\}}{max\{x_{1j}, x_{2j}, \cdots x_{nj}\} - min\{x_{1j}, x_{2j}, \cdots x_{nj}\}} \quad (8.2)$$

当为负向指标时：

$$v_{ij} = \frac{max\{x_{1j}, x_{2j}, \cdots x_{nj}\} - x_{ij}}{max\{x_{1j}, x_{2j}, \cdots x_{nj}\} - min\{x_{1j}, x_{2j}, \cdots x_{nj}\}} \quad (8.3)$$

其中，$max\{x_{1j}, x_{2j}, \cdots x_{nj}\}$为各项指标的最大值，$min\{x_{1j}, x_{2j}, \cdots x_{nj}\}$为各项指标的最小值。

步骤 3：定义理想解和反理想解

设决策问题有 m 个目标f_j(j = 1,2… m)，n 个可行解，并设该问题的规范化目标的正负理想解分别为是Z^+、Z^-，其中:

$$Z^+ = (maxv_{i1}, maxv_{i2}, \ldots maxv_{ij}) \quad (8.4)$$

$$Z^- = (minv_{i1}, minv_{i2}, \ldots minv_{ij}) \quad (8.5)$$

v_{ij}为第i个评价对象的第j个评价指标的标准化数据。

步骤 4：计算评价对象到理想解和反理想解的欧氏距离

用欧几里得范数作为距离的测度，则从任意可行解到的距离为

$$S_i^+ = \sqrt{\sum_{j=1}^{m}\left(Z_{ij} - Z_j^+\right)^2} \tag{8.6}$$

$$S_i^- = \sqrt{\sum_{j=1}^{m}\left(Z_{ij} - Z_j^-\right)^2} \tag{8.7}$$

式中，S_i^+为 i 到正最大值的接近度，S_i^-为 i 到负最大值的接近度。

步骤 5：计算 TOPSIS 得分

各个对象的 TOPSIS 得分为：

$$C_i = \frac{S_i^-}{S_i^- + S_i^+} \tag{8.8}$$

式中：C_i为第 i 个评价对象的 TOPSIS 得分，以贴近 0 或 1 的距离判断表现优劣，越贴近 1，表现越优，反之越贴近 0，表现越差。

步骤 6：确定分位数位置及分类界限

对 n 个对象按为归一化的得分C_i进行由大到小的排序，并选取 q 分位数的数值作为将 n 个对象分为 q 类的界限，第 p 分位数的位置W_p为：

$$W_p = \frac{(n+1) * p}{q} \tag{8.9}$$

步骤 7：确定类别

比较S_i与各个分位数的大小，得到对象 i 所属类别L_i

$$L_i = \begin{cases} 0，\ S_i < W_1 \\ 1，\ \leq S_i < W_3 \\ \cdots \\ p-2，\ W_{p-2} \leq S_i < W_{p-1} \\ p-1，\ W_{p-1} \leq S_i \end{cases} \tag{8.10}$$

其中 0，1，….p-1 分别为类别 1 到类别 p 的映射。

8.2 Topsis-分位数评级模型的求解

利用 Python 编程求解上述模型，计算出各个婴儿综合睡眠质量的 TOPSIS 得分，结果如图 8-1 所示。

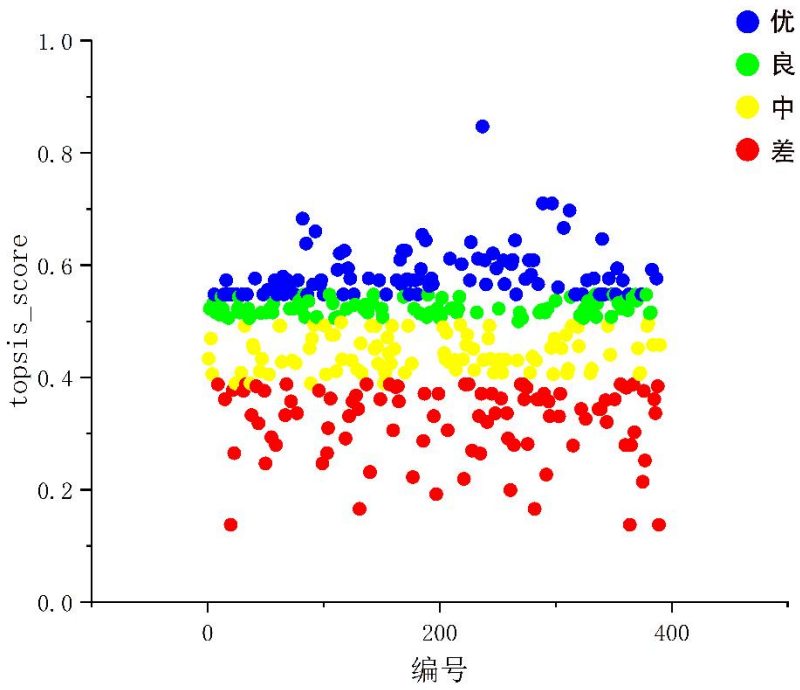

图 8-1 婴儿综合睡眠质量分布散点图

要将婴儿睡眠质量进行优、良、中、差四分类的综合评判，通过四分位的计算得到各婴儿 TOPSIS 得分的划分界限分别为 0.3891、0.4991 和 0.5481，由此得到表 8-1 婴儿睡眠质量综合评判（详见附录二）。

表 8-1 婴儿睡眠质量综合评判

睡眠质量评级	婴儿编号	睡眠质量 TOPSIS 得分	睡眠质量评级	婴儿编号	睡眠质量 TOPSIS 得分
优	237	0. 847217874	中	384	0. 457887352
	289	0. 710102051		331	0. 415090499
	297	0. 710102051		132	0. 460689112
	312	0. 697322174		204	0. 443310362
	82	0. 683123975		218	0. 493752371
	307	0. 666369644		371	0. 451941016
	93	0. 660407642		92	0. 493752371
	185	0. 654070103		62	0. 491942672
	340	0. 646884592		101	0. 491942672
	…	…		…	…
良	2	0. 522553836	差	294	0. 357464266
	5	0. 532320735		386	0. 336155625
	7	0. 515796524		50	0. 24692389
	8	0. 515796524		388	0. 384396885
	10	0. 522553836		55	0. 293399774
	11	0. 511343704		376	0. 376178512
	12	0. 543634995		9	0. 387640135
	13	0. 522553836		290	0. 371146943
	18	0. 506131593		360	0. 279694165
	…			…	…

8.3 婴儿睡眠质量预测模型的选择

在得到婴儿综合睡眠质量的评级后，我们需要训练适当的模型来刻画婴儿的行睡眠质量与母亲的身体指标与心理指标的关系，并通过训练完成的模型进行婴儿综合睡眠质量的预测。

采用与问题二相同的方法，我们采用均方根误差（RMSE）、平均绝对误差（MAE）和准确率作为评价指标，对逻辑回归模型、KNN 分类模型、随机森林模型、GBDT 分类模型和 XGBoost 模型等多种预测模型进行对婴儿睡眠质量预测效果的比较分析，以选出可信度最高的预测模型去预测婴儿睡眠质量。

通过使用十折交叉验证方法，重复 10 次实验的平均结果如表 8-2 所示。

表 8-2 各模型婴儿睡眠质量的预测有效性分析

模型	MASE	MAE	准确率
逻辑回归模型	1. 2817	0.8571	0.4286
KNN 分类模型	1. 1495	0.75	0.4643
随机森林模型	1.1495	0. 6785	0.5714
GBDT 分类模型	1.1650	0.6429	0.6429
XGBoost 分类模型	1.1180	0.6071	0.6429

综合 MASE、MAE 和准确率指标，可以得出 GBDT 分类模型相对其他模型预测准确性更高的结论。因此我们认为：使用 GBDT 分类模型预测得到的婴儿睡眠质量可信度较高。

8.4 最后 20 组婴儿的综合睡眠质量预测结果

采用 GBDT 模型对最后 20 组婴儿的综合睡眠质量进行预测，所得结果如表 8-3 所示：

表 8-3 最后 20 组婴儿行为特征预测结果

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

九、 问题五模型的建立与求解

根据题目所给信息，本题仍可以转化为数学规划问题，在问题三的基础上，兼顾不同的期望睡眠质量干预效果，构建行为特征&睡眠质量治疗策略数学规划模型，为确定行为特征&睡眠质量治疗方案提供科学依据。

9.1 行为特征&睡眠质量治疗策略数学规划模型的建立

问题五要求在问题三的基础上兼顾不同的期望睡眠质量干预效果，因此问题五的行为特征&睡眠质量治疗策略数学规划模型是在问题三的行为特征治疗策略数学规划模型基础上的进一步深化。

9.1.1 约束条件的构建

在问题三的约束条件的基础上，问题五要求兼顾不同的睡眠质量干预效果，因此需增加约束条件使得治疗对象的睡眠质量达到期望评级：

$$L_i^* = L\_desire \tag{9.1}$$

其中，L_i^*是第 i 组母亲经治疗改变 CBTS、EPDS、HADS 后用问题四中的 GBDT 分类模型模型预测的婴儿睡眠质量评级，L_desire为期望睡眠质量评级。

则行为特征&睡眠质量治疗策略数学规划模型总的约束条件为：

$$\text{s.t.}\begin{cases} BC_i^* = BC\_desire \\ L_i^* = L\_desire \\ \text{cost}_1 = 200e^{0.8811\text{heat}_1} \\ \text{cost}_2 = 500e^{0.6649\text{heat}_2} \\ \text{cost}_3 = 300e^{0.7459\text{heat}_3} \end{cases} \tag{9.2}$$

9.1.2 目标函数的构建

由题意可知，要通过治疗实现当前婴儿行为特征和睡眠质量到期望行为特征和睡眠质量的改变，同时要尽可能的减少所花费的治疗费用，由此可知目标函数：

$$min \sum_{k=1}^{3} cost_k \tag{9.3}$$

整理上述条件，可得行为特征治疗方案数学规划模型：

目标函数：

$$min \sum_{k=1}^{3} cost_k \tag{9.4}$$

约束条件：

$$\text{s.t.}\begin{cases} BC_i^* = BC_{desire} \\ L_i^* = L\_desire \\ \text{cost}_1 = 200e^{0.8811\text{heat}_1} \\ \text{cost}_2 = 500e^{0.6649\text{heat}_2} \\ \text{cost}_3 = 300e^{0.7459\text{heat}_3} \\ 0 \le \text{heat}_1^i \le EPDS^i \\ 0 \le \text{heat}_2^i \le \text{CBTS}^i \\ 0 \le \text{heat}_3^i \le \text{HADS}^i \end{cases} \tag{9.5}$$

9.2 行为特征&睡眠质量治疗策略数学规划模型的求解

选取 238 号婴儿作为行为特征治疗对象。基于假设 1 和以上列出的目标函数与约束条件，运用 Python 软件编写程序求解，可得出最优治疗方案如表 9-1 所示：

表 9-1 兼顾睡眠质量后 238 号婴儿最优治疗方案的调整

期望婴儿行为特征	原治疗后母亲心理指标		调整治疗后母亲心理指标		原总治疗费用（元）	调整后总治疗费用（元）
中等型	CBTS	12	CBTS	12	22454.67	56621.49
	EPDS	14	EPDS	13		
	HADS	17	HADS	13		
安静型	CBTS	9	CBTS	9	313480.78	338987.38
	EPDS	12	EPDS	11		
	HADS	13	HADS	13		

由上表可知，在兼顾要将 238 号婴儿的综合睡眠质量评级为优之后，若要将 238 号婴儿的行为特征从矛盾型转变为中等型同时使总治疗费用最低，治疗方案应调整为将母亲的 CBTS、EPDS、HADS 分别治疗到 12、13 和 13 分的水平以提高母亲的心理健康水平，此时最低的总治疗费用为 56621.49 元。

在兼顾要将 238 号婴儿的综合睡眠质量评级为优之后，若要将 238 号婴儿的行为特征矛盾型转变为安静型同时使总治疗费用最低，则需将母亲的 CBTS、EPDS、HADS 分别治疗到 9、11、13 分的水平以提高母亲的心理健康水平，此时的总治疗费用为 338987.38 元。

十、 模型的评价、改进与推广

10.1 模型的评价

10.1.1 模型优点

1. 采用 Spearman 相关系数探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关系。Spearman 相关系数是一种灵活和鲁棒的统计指标，适合用于测量非线性关系，特别是在观察到的数据不满足正态分布或存在异常值的情况，在本题不满足正态分布的条件下得到充分合理应用。

2. 采用 XGBoost 模型，并通过比对其他机器学习模型(KNN 分类模型、随机森林模型、GDBT 分类模型），多角度验证及分析，避免了单个模型预测可能存在的偶然性，并提高了数据精度，使婴儿的行为特征信息更加可信。

3. Topsis-分位数评级模型简单易懂，易于理解和实施。也能够同时考虑多个评估指标，对不同指标加权求和，综合评估候选方案的表现并进行分类。且模型本身不易受极端值和主观分界的影响。

10.1.2 模型缺点

1.Spearman 相关系数仅基于变量的相对排序，它可能会丢失变量的一些信息。这意味着它无法全面捕捉到变量之间的更复杂的关系。

2. XGBoost 模型有很多可调整的参数和超参数，如学习率、树的深度、叶子节点权重等。这使得模型的参数调整复杂。此外与一些其他模型相比，由于 XGBoost 是基于梯度提升框架的，每个迭代周期都需构建并训练一棵新的决策树，这导致其训练时间较长。

10.2 模型的改进

对于 XGBoost 模型可以对其参数调整优化：XGBoost 模型中有许多参数和超参数，可以通过使用调参技巧如网格搜索、随机搜索等方法来寻找最佳配置。这可以提升模型的性能和稳定性。此外还可以提前停止策略：通过设置合适的迭代次数和使用提前停止策略，可以在训练过程中提前终止模型，以减少训练时间和防止过拟合。

参考文献

[1] Rebekić A, Lončarić Z, Petrović S, et al. Pearson’s or Spearman’s correlation coefficient-which one to use?[J]. Poljoprivreda, 2015, 21(2): 47-54.[J]. .

[2] 宋玲玲,王时绘,杨超,盛潇.改进的 XGBoost 在不平衡数据处理中的应用研究[J].计算机科学,2020,47(06):98-103.[J]. .

[3] Wang Chen,Deng Chengyuan,Wang Suzhen. Imbalance-XGBoost: leveraging weighted and focal losses for binary label-imbalanced classification with XGBoost[J]. Pattern Recognition Letters,2020(prepublish).[J]. .

[4] 黄卿,谢合亮.机器学习方法在股指期货预测中的应用研究——基于 BP 神经网络、SVM 和 XGBoost 的比较分析[J].数学的实践与认识,2018,48(08):297-307.[J].

[5] Amir Bahador Parsa, Ali Movahedi, Homa Taghipour, Sybil Derrible, Abolfazl (Kouros) Mohammadian. Toward safer highways, application of XGBoost and SHAP for real-time accident detection and feature analysis[J]. Accident Analysis and Prevention,2020,136.[J]. .

[6] 郑列,穆新宇.改进的 XGBoost 模型在短租房价格预测中的应用[J].湖北工业大学学报,2021,36(02):104-109.[J]..

[7] 谢秋霞,贾立,陈琪婷,尹燕旻,MenentiMassimo.闪电河流域农牧交错带微波遥感土壤水分产品评价[J].遥感学报,2021,25(04):974-989.[J]..

[8] Domínguez-Almendros S, Benítez-Parejo N, Gonzalez-Ramirez A R. Logistic regression models[J]. Allergologia et immunopathologia, 2011, 39(5): 295-305.[J]. .

[9] Guo G, Wang H, Bell D, et al. KNN model-based approach in classification[C]//On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE: OTM Confederated International Conferences, CoopIS, DOA, and ODBASE 2003, Catania, Sicily, Italy, November 3-7, 2003. Proceedings. Springer Berlin Heidelberg, 2003: 986-996.[J]. .

[10] Biau G. Analysis of a random forests model[J]. The Journal of Machine Learning Research, 2012, 13: 1063-1095.[J]. .

[11] Yu Z, Wang Z, Zeng F, et al. Volcanic lithology identification based on parameter-optimized GBDT algorithm: A case study in the Jilin Oilfield, Songliao Basin, NE China[J]. Journal of Applied Geophysics, 2021, 194: 104443.[J]. .

[12] Wang X, Feng S, Wang L. A Interval Number Ranking Method Based on Interval Quantile and TOPSIS for Decision Problems[C]//Fuzzy Sets and Operations Research 9. Springer International Publishing, 2019: 375-383.[J].

附录：

附录一

介绍：下表为问题一的 Spearman 相关系数表

	母亲年龄	婚姻状况	教育程度	妊娠时间（周数）	分娩方式	CBTS	EPDS	HADS	婴儿行为特征	婴儿性别	婴儿年龄（月）	整晚睡眠时间（秒）	睡醒次数	入睡方式
母亲年龄	1 (0.000***)	0.011 (0.833)	0.183 (0.000***)	-0.124 (0.016**)	0.039 (0.453)	-0.033 (0.522)	-0.087 (0.090*)	-0.069 (0.178)	-0.087 (0.089*)	-0.066 (0.202)	-0.018 (0.734)	0.017 (0.747)	0.051 (0.323)	-0.088 (0.085*)
婚姻状况	0.011 (0.833)	1 (0.000***)	0.11 (0.032**)	0.004 (0.946)	0.023 (0.661)	-0.008 (0.882)	-0.023 (0.662)	-0.066 (0.203)	0.005 (0.919)	-0.09 (0.080*)	-0.081 (0.115)	0.019 (0.706)	-0.036 (0.488)	-0.052 (0.313)
教育程度	0.183 (0.000***)	0.11 (0.032**)	1 (0.000***)	0.026 (0.615)	-0.016 (0.760)	-0.072 (0.163)	-0.114 (0.026**)	-0.1 (0.051*)	0.041 (0.426)	-0.004 (0.931)	0.028 (0.589)	-0.058 (0.256)	0.07 (0.172)	-0.036 (0.487)
妊娠时间（周数）	-0.124 (0.016**)	0.004 (0.946)	0.026 (0.615)	1 (0.000***)	-0.18 (0.000***)	-0.077 (0.133)	-0.05 (0.336)	-0.106 (0.039**)	-0.002 (0.972)	0.035 (0.497)	0.006 (0.914)	0.07 (0.176)	0.087 (0.089*)	0.04 (0.441)
分娩方式	0.039 (0.453)	0.023 (0.661)	-0.016 (0.760)	-0.18 (0.000***)	1 (0.000***)	-0.021 (0.683)	-0.02 (0.700)	-0.066 (0.201)	0.003 (0.956)	-0.112 (0.029**)	-0.049 (0.337)	0.015 (0.766)	-0.063 (0.221)	0.063 (0.218)
CBTS	-0.033 (0.522)	-0.008 (0.882)	-0.072 (0.163)	-0.077 (0.133)	-0.021 (0.683)	1 (0.000***)	0.781 (0.000***)	0.71 (0.000***)	0.114 (0.027**)	0.075 (0.142)	-0.08 (0.118)	-0.128 (0.013**)	0.074 (0.150)	0.054 (0.295)
EPDS	-0.087 (0.090*)	-0.023 (0.662)	-0.114 (0.026**)	-0.05 (0.336)	-0.02 (0.700)	0.781 (0.000***)	1 (0.000***)	0.784 (0.000***)	0.132 (0.010***)	0.047 (0.363)	-0.044 (0.389)	-0.173 (0.001***)	0.112 (0.029**)	0.009 (0.863)
HADS	-0.069 (0.178)	-0.066 (0.203)	-0.1 (0.051*)	-0.106 (0.039**)	-0.066 (0.201)	0.71 (0.000***)	0.784 (0.000***)	1 (0.000***)	0.123 (0.017**)	0.033 (0.523)	0.004 (0.942)	-0.122 (0.017**)	0.07 (0.172)	0.059 (0.248)
婴儿行为特征	-0.087 (0.089*)	0.005 (0.919)	0.041 (0.426)	-0.002 (0.972)	0.003 (0.956)	0.114 (0.027**)	0.132 (0.010***)	0.123 (0.017**)	1 (0.000***)	-0.024 (0.638)	0.344 (0.000***)	-0.118 (0.021**)	0.245 (0.000***)	0.002 (0.971)
婴儿性别	-0.066 (0.202)	-0.09 (0.080*)	-0.004 (0.931)	0.035 (0.497)	-0.112 (0.029**)	0.075 (0.142)	0.047 (0.363)	0.033 (0.523)	-0.024 (0.638)	1 (0.000***)	0.001 (0.977)	-0.049 (0.345)	0.081 (0.116)	-0.035 (0.492)
婴儿年龄（月）	-0.018 (0.734)	-0.081 (0.115)	0.028 (0.589)	0.006 (0.914)	-0.049 (0.337)	-0.08 (0.118)	-0.044 (0.389)	0.004 (0.942)	0.344 (0.000***)	0.001 (0.977)	1 (0.000***)	0.183 (0.000***)	-0.008 (0.876)	0.189 (0.000***)
整晚睡眠时间（秒）	0.017 (0.747)	0.019 (0.706)	-0.058 (0.256)	0.07 (0.176)	0.015 (0.766)	-0.128 (0.013**)	-0.173 (0.001***)	-0.122 (0.017**)	-0.118 (0.021**)	-0.049 (0.345)	0.183 (0.000***)	1 (0.000***)	-0.318 (0.000***)	0.232 (0.000***)
睡醒次数	0.051 (0.323)	-0.036 (0.488)	0.07 (0.172)	0.087 (0.089*)	-0.063 (0.221)	0.074 (0.150)	0.112 (0.029**)	0.07 (0.172)	0.245 (0.000***)	0.081 (0.116)	-0.008 (0.876)	-0.318 (0.000***)	1 (0.000***)	-0.255 (0.000***)
入睡方式	-0.088 (0.085*)	-0.052 (0.313)	-0.036 (0.487)	0.04 (0.441)	0.063 (0.218)	0.054 (0.295)	0.009 (0.863)	0.059 (0.248)	0.002 (0.971)	-0.035 (0.492)	0.189 (0.000***)	0.232 (0.000***)	-0.255 (0.000***)	1 (0.000***)

注：***、**、*分别代表 1%、5%、10%的显著性水平，括号内为 p 值

附录二

介绍：下表为问题四的婴儿睡眠质量综合评判表

编号	topsis_score	睡眠质量评级	编号	topsis_score	睡眠质量评级
237	0. 847217874	优	115	0. 498201508	中
289	0. 710102051	优	380	0. 496855787	中
297	0. 710102051	优	218	0. 493752371	中
312	0. 697322174	优	92	0. 493752371	中
82	0. 683123975	优	62	0. 491942672	中
307	0. 666369644	优	101	0. 491942672	中
93	0. 660407642	优	145	0. 491942672	中
185	0. 654070103	优	141	0. 491942672	中
340	0. 646884592	优	63	0. 491942672	中
188	0. 64461527	优	32	0. 491942672	中
265	0. 64461527	优	379	0. 491942672	中
227	0. 640979608	优	159	0. 491942672	中
85	0. 638669452	优	345	0. 491942672	中
118	0. 625906432	优	173	0. 491942672	中
168	0. 625906432	优	243	0. 491942672	中
171	0. 625906432	优	316	0. 491942672	中
114	0. 621110071	优	314	0. 491942672	中
246	0. 621110071	优	319	0. 489669077	中
209	0. 611691484	优	203	0. 489669077	中
233	0. 611691484	优	206	0. 47987455	中
166	0. 609940714	优	224	0. 475755153	中
263	0. 60940613	优	309	0. 475755153	中
239	0. 608772733	优	109	0. 475755153	中
277	0. 608772733	优	107	0. 475755153	中
281	0. 608772733	优	155	0. 47109117	中

255	0. 608522864	优	3	0. 469208823	中
260	0. 606346961	优	299	0. 469208823	中
219	0. 601648209	优	90	0. 469208823	中
262	0. 601648209	优	242	0. 469208823	中
121	0. 594430807	优	132	0. 460689112	中
249	0. 594430807	优	39	0. 458374917	中
353	0. 594430807	优	384	0. 457887352	中
184	0. 592951621	优	223	0. 457887352	中
112	0. 591750968	优	390	0. 457887352	中
383	0. 591750968	优	320	0. 45568721	中
279	0. 583159204	优	146	0. 455684752	中
65	0. 579632672	优	296	0. 452774766	中
139	0. 576481686	优	371	0. 451941016	中
333	0. 576481686	优	305	0. 451941016	中
41	0. 576280586	优	88	0. 451941016	中
123	0. 576280586	优	40	0. 451941016	中
193	0. 576280586	优	144	0. 450963239	中
346	0. 576280586	优	161	0. 450963239	中
387	0. 576280586	优	229	0. 450963239	中
172	0. 575205722	优	204	0. 443310362	中
182	0. 575205722	优	156	0. 443310362	中
274	0. 575205722	优	347	0. 440338955	中
16	0. 573104707	优	280	0. 437053714	中
58	0. 573104707	优	244	0. 433239175	中
69	0. 573104707	优	205	0. 433239175	中
78	0. 573104707	优	47	0. 433239175	中
98	0. 573104707	优	1	0. 433239175	中
148	0. 573104707	优	116	0. 43222165	中
163	0. 573104707	优	220	0. 43222165	中
175	0. 573104707	优	252	0. 43222165	中
179	0. 573104707	优	257	0. 43160714	中
226	0. 573104707	优	213	0. 431099505	中
327	0. 573104707	优	238	0. 431099505	中
358	0. 573104707	优	267	0. 430383403	中
61	0. 566300467	优	283	0. 430383403	中
167	0. 566300467	优	124	0. 430383403	中
194	0. 566300467	优	73	0. 430383403	中
256	0. 566300467	优	64	0. 427590604	中
285	0. 566300467	优	208	0. 427590604	中
91	0. 56594217	优	76	0. 425475383	中
97	0. 56594217	优	158	0. 42493143	中
240	0. 56594217	优	250	0. 42493143	中
191	0. 562946286	优	176	0. 42493143	中
70	0. 56076301	优	147	0. 42493143	中
302	0. 56076301	优	331	0. 415090499	中
52	0. 555925091	优	311	0. 415090499	中
6	0. 548058984	优	129	0. 414213562	中
14	0. 548058984	优	214	0. 414213562	中
17	0. 548058984	优	230	0. 413320851	中
19	0. 548058984	优	45	0. 410859357	中
21	0. 548058984	优	111	0. 410859357	中
25	0. 548058984	优	278	0. 408546252	中

30	0. 548058984	优	254	0. 408546252	中
34	0. 548058984	优	170	0. 408546252	中
48	0. 548058984	优	29	0. 408546252	中
60	0. 548058984	优	373	0. 408546252	中
66	0. 548058984	优	247	0. 408546252	中
75	0. 548058984	优	330	0. 407516141	中
86	0. 548058984	优	153	0. 407516141	中
100	0. 548058984	优	133	0. 407516141	中
117	0. 548058984	优	310	0. 407140297	中
126	0. 548058984	优	298	0. 405843439	中
154	0. 548058984	优	372	0. 405843439	中
174	0. 548058984	优	4	0. 405843439	中
181	0. 548058984	优	102	0. 405409652	中
266	0. 548058984	优	53	0. 405409652	中
317	0. 548058984	优	152	0. 389441485	中
318	0. 548058984	优	89	0. 389441485	中
323	0. 548058984	优	37	0. 389441485	中
338	0. 548058984	优	24	0. 389441485	中
341	0. 548058984	优	68	0. 388057826	差
342	0. 548058984	优	225	0. 388057826	差
349	0. 548058984	优	272	0. 388057826	差
352	0. 548058984	优	9	0. 387640135	差
363	0. 548058984	优	157	0. 387640135	差
374	0. 548058984	优	367	0. 387640135	差
83	0. 547500608	良	137	0. 387640135	差
105	0. 547500608	良	222	0. 387640135	差
143	0. 547500608	良	270	0. 387640135	差
190	0. 547500608	良	33	0. 387640135	差
334	0. 547500608	良	356	0. 387640135	差
350	0. 547500608	良	388	0. 384396885	差
369	0. 547500608	良	42	0. 384396885	差
378	0. 547500608	良	164	0. 384396885	差
12	0. 543634995	良	162	0. 38301539	差
27	0. 543634995	良	361	0. 381945134	差
169	0. 543634995	良	275	0. 381945134	差
217	0. 543634995	良	22	0. 377869086	差
313	0. 543634995	良	96	0. 376853364	差
202	0. 541625083	良	376	0. 376178512	差
366	0. 541625083	良	49	0. 376178512	差
300	0. 536985294	良	31	0. 376178512	差
370	0. 536985294	良	290	0. 371146943	差
79	0. 536009763	良	245	0. 371146943	差
87	0. 536009763	良	236	0. 371146943	差
328	0. 536009763	良	187	0. 371146943	差
359	0. 536009763	良	199	0. 371146943	差
5	0. 532320735	良	304	0. 371146943	差
57	0. 532320735	良	128	0. 367648613	差
108	0. 532320735	良	106	0. 362676898	差
142	0. 532320735	良	253	0. 362676898	差
336	0. 532320735	良	351	0. 36129785	差
354	0. 532320735	良	15	0. 361151561	差
136	0. 529453322	良	284	0. 361151561	差

127	0. 52945194	良	287	0. 361151561	差
2	0. 522553836	良	273	0. 361151561	差
10	0. 522553836	良	385	0. 361151561	差
13	0. 522553836	良	149	0. 361151561	差
26	0. 522553836	良	343	0. 359551751	差
35	0. 522553836	良	294	0. 357464266	差
54	0. 522553836	良	125	0. 357464266	差
71	0. 522553836	良	165	0. 357464266	差
74	0. 522553836	良	72	0. 357464266	差
80	0. 522553836	良	339	0. 344039294	差
81	0. 522553836	良	130	0. 344007209	差
120	0. 522553836	良	337	0. 344007209	差
150	0. 522553836	良	322	0. 344007209	差
178	0. 522553836	良	386	0. 336155625	差
198	0. 522553836	良	248	0. 336155625	差
210	0. 522553836	良	77	0. 336155625	差
211	0. 522553836	良	258	0. 336155625	差
212	0. 522553836	良	67	0. 332831095	差
216	0. 522553836	良	38	0. 332831095	差
251	0. 522553836	良	122	0. 331136684	差
293	0. 522553836	良	295	0. 331136684	差
325	0. 522553836	良	195	0. 331136684	差
332	0. 522553836	良	234	0. 331136684	差
357	0. 522553836	良	303	0. 330537643	差
135	0. 521591915	良	326	0. 326236413	差
362	0. 519872613	良	241	0. 320655282	差
288	0. 519552189	良	344	0. 320655282	差
7	0. 515796524	良	44	0. 318518086	差
8	0. 515796524	良	104	0. 30954164	差
28	0. 515796524	良	207	0. 305908335	差
51	0. 515796524	良	160	0. 305908335	差
56	0. 515796524	良	368	0. 302677826	差
138	0. 515796524	良	55	0. 293399774	差
192	0. 515796524	良	119	0. 29120837	差
215	0. 515796524	良	259	0. 29120837	差
232	0. 515796524	良	186	0. 287083986	差
269	0. 515796524	良	276	0. 2814611	差
286	0. 515796524	良	360	0. 279694165	差
291	0. 515796524	良	264	0. 279694165	差
329	0. 515796524	良	59	0. 279614622	差
382	0. 515796524	良	365	0. 279614622	差
46	0. 514777715	良	315	0. 278860678	差
183	0. 514239181	良	228	0. 269601819	差
381	0. 514239181	良	23	0. 265243416	差
11	0. 511343704	良	103	0. 265243416	差
321	0. 510405694	良	235	0. 264406082	差
84	0. 507805692	良	377	0. 25219881	差
94	0. 507805692	良	50	0. 24692389	差
113	0. 507805692	良	99	0. 24692389	差
151	0. 507805692	良	140	0. 23140718	差
189	0. 507805692	良	292	0. 226887474	差
201	0. 507805692	良	177	0. 222470795	差

335	0. 507805692	良	221	0. 219345093	差
348	0. 507805692	良	375	0. 214194534	差
18	0. 506131593	良	261	0. 199189346	差
200	0. 506131593	良	197	0. 192141275	差
324	0. 506131593	良	282	0. 165783344	差
36	0. 505903347	良	131	0. 165783344	差
110	0. 505903347	良	364	0. 137554734	差
271	0. 505903347	良	389	0. 137554734	差
268	0. 5	良	20	0. 137554734	差

附录三

介绍：该 NOTEBOOK 代码用于解决数据预处理与问题 1

```
#导入相关库
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import matplotlib
import seaborn as sns
import warnings

warnings.filterwarnings("ignore")                    #忽略警告信息
plt.rcParams['axes.unicode_minus'] = False
plt.rcParams['font.sans-serif'] = ['SimHei']
plt.rcParams['savefig.dpi'] = 300
myfont = matplotlib.font_manager.FontProperties(fname=r"simhei.ttf")
#导入数据
df=pd.read_excel(r"附件.xlsx")
#数据查看及其预处理
# 查看数据的结构
print(df.shape)
# 查看数据的信息
df.info()
df.describe()
df['婴儿行为特征'].value_counts()
df['整晚睡眠时间（时：分：秒）'].value_counts()

df[df['婚姻状况']>2]['婚姻状况'].value_counts()

df.drop(df[df['婚姻状况']>2].index, inplace=True)
df.drop(df[df['整晚睡眠时间（时：分：秒）']=='99:99'].index, inplace=True)

print(df[df['婚姻状况']>2]['婚姻状况'].value_counts(),df['整晚睡眠时间（时：分：秒）'].value_counts())

df['整晚睡眠时间（时：分：秒）']=df['整晚睡眠时间（时：分：秒）'].iloc[:366].apply(lambda x:(x.hour * 60 + x.minute) * 60 )

rename_list={'整晚睡眠时间（时：分：秒）':'整晚睡眠时间（秒）'}
df.rename(rename_list,axis=1,inplace=True)
# 数据展示
```

```
df.head(3).T

#保存处理完的数据
df.to_csv('预处理后的数据.csv')

#问题 1
df_1=df.iloc[:-20,1:]
#对婴儿行为特征进行编码，分 3 类，0 为安静、1 中等、2 矛盾
labe_bbaction={'安静型':0,'中等型':1,'矛盾型':2,}
df_1['婴儿行为特征']=df_1['婴儿行为特征'].map(labe_bbaction)
df_1.describe()

#数据分布图
plt.figure(figsize=(20,8),dpi=300)
plt.subplots_adjust(wspace =0.3, hspace =0.3)
n=-1
for i in df_1.columns:
    s=df_1[i]
    n+=1
    plt.subplot(2,7,n+1)
    s.hist(bins=30,alpha = 0.5)
    s.plot(kind = 'kde', secondary_y=True)
    plt.title(i,fontproperties=myfont)
    plt.grid()
#     plt.legend()
plt.savefig('数据分布图.png') # 保存图片
plt.show()

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from scipy import stats


for i in range(df_1.shape[1]):
    df_1_1=df_1.iloc[:,i]
    u = df_1_1.mean()  # 计算均值
    std =df_1_1.std()  # 计算标准差

# kstest 方法中的参数分别是：待检验的数据，检验方法（这里设置成 norm 正态分布），均值与标准差
# 返回两个值：statistic → D 值，pvalue → P 值
# 当 p 值大于 0.05，说明待检验的数据符合为正态分布
    result = stats.kstest(df_1_1, 'norm', (u, std))
    print(df_1_1.name,result,)
    print('是否正态:',result[1]>0.05)

# 多变量正态性检验是一种正态性检验，它确定给定的一组变量是否来自于正态分布。
from pingouin import multivariate_normality

# perform the Multivariate Normality Test
multivariate_normality(df_1, alpha=.05)
```

```
df_1.corr(method='spearman')


plt.figure(figsize=(22, 12), dpi=300)
#画半个
mask = np.zeros_like(df_1.corr(method='spearman'))
mask[np.triu_indices_from(mask)] = True
sns.heatmap(df_1.corr(method='spearman'),mask=mask                                    ,square=True,
annot=True,cmap='coolwarm' )
plt.title('问题 1 热力图',fontproperties=myfont)
plt.savefig('问题 1 热力图.png',bbox_inches = 'tight') #  保存图片
plt.show()
```

附录四

介绍：该 NOTEBOOK 代码用于解决问题 2

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import matplotlib
import seaborn as sns
import warnings

warnings.filterwarnings("ignore")                    #忽略警告信息
plt.rcParams['axes.unicode_minus'] = False
plt.rcParams['font.sans-serif'] = ['SimHei']
plt.rcParams['savefig.dpi'] = 300
myfont = matplotlib.font_manager.FontProperties(fname=r"simhei.ttf")

#建模
from imblearn.over_sampling import SMOTE
from sklearn.preprocessing import scale,LabelEncoder #用于数据预处理模块的缩放器、标签编
码
from sklearn.model_selection import train_test_split #数据集分类器 用于划分训练集和测试集
from sklearn.metrics import classification_report,accuracy_score #评估预测结果
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import GradientBoostingClassifier
from xgboost import XGBClassifier
from sklearn.model_selection import train_test_split,GridSearchCV
from sklearn.metrics import r2_score
from sklearn.metrics import mean_squared_error
from sklearn.metrics import mean_absolute_error

df=pd.read_csv(r"预处理后的数据.csv").iloc[:,1:]
df_2=df.iloc[:,0:10]
```

```
labe_bbaction={'安静型':0,'中等型':1,'矛盾型':2,}
df_2['婴儿行为特征']=df_2['婴儿行为特征'].map(labe_bbaction)

def mapfun1(x):
    if x<=19:
        return 1
    elif x>19 and x<34:
        return 2
    elif x>=34:
        return 3
df_2['母亲年龄'] = df_2['母亲年龄'].map(mapfun1)

def mapfun3(x):
    if x<=10:
        return 1
    elif x>=11 and x<=20:
        return 2
    elif x>20:
        return 3
def mapfun_HADS(x):
    if x<=7:
        return 1
    elif x>=8 and x<=10:
        return 2
    elif x>=11 and x<=14:
        return 3
    elif x>14:
        return 4
def mapfun_EPDS(x):
    if x<=8:
        return 1
    elif x>=9 and x<=13:
        return 2
    elif x>13:
        return 3
def mapfun_CBTS(x):
    if x<=3:
        return 1
    elif x>=4 and x<=6:
        return 2
    elif x>=7 and x<=12:
        return 3
    elif x>=13 and x<=24:
        return 4
df_2['CBTS'] = df_2['CBTS'].map(mapfun_CBTS)
df_2['EPDS'] = df_2['EPDS'].map(mapfun_EPDS)
df_2['HADS'] = df_2['HADS'].map(mapfun_HADS)

col_all = list(df_2.columns) # 全部特征
for i in col_all:
    print(i,"特征的各数据出现次数：\n",df_2[i].value_counts()) # “\n”代表换行
value_counts()可设置参数 ascending=True|False 对结果升序|降序
```

```
        plt.figure(figsize=(10,4)) # 新增一个画布
        #plt.hist(data[i],color="darkcyan") # 对该变量绘制直方图 选个自己喜欢的颜色参数~
        sns.countplot(x=i,data=df_2)
        plt.show()
        print("--"*30,"\n") #分割线


del df_2['分娩方式']
del df_2['婚姻状况']
df_2

# 训练的数据
X=df_2.iloc[:-20,1:7]
y=df_2['婴儿行为特征'].iloc[:-20,].astype(int)
#要预测的数据
df_2_predict=df_2.iloc[-20:,1:7]

#对数据集进行划分 test_size 参数设置为 0.1 取其中 10%的数据作为测试集
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.1)

#模型一：逻辑回归
# model_LR = LogisticRegression() # 实例化
model_LR= LogisticRegression(solver='liblinear')
model_LR = model_LR.fit(X_train,y_train) # 拟合

y_pred_LR = model_LR.predict(X_test) #预测
y_pred_LR=pd.DataFrame(y_pred_LR) # 转化为 DataFrame 形式
y_pred_LR.head() # 预览前五行
y_pred_LR.value_counts() # 预测结果数据分布
y_pred_LR.hist(color="darkcyan") # 绘制直方图查看分布
plt.title('LR',fontsize=14)

print('准确度:',accuracy_score(y_test,y_pred_LR)) #评估准确度
print('R2:',r2_score(y_test,y_pred_LR))#R
print('RMSE:',np.sqrt(mean_squared_error(y_test,y_pred_LR)))
print('MAE:',mean_absolute_error(y_test,y_pred_LR))
print(classification_report(y_test,y_pred_LR)) #输出分类预测报告

#模型二：KNN
model_KNN = KNeighborsClassifier(n_neighbors=10,p=1,weights="distance") # 实力化逻辑回
归模型
model_KNN = model_KNN.fit(X_train,y_train) # 拟合

y_pred_KNN = model_KNN.predict(X_test)
y_pred_KNN=pd.DataFrame(y_pred_KNN) # 转化为 DataFrame 形式

y_pred_KNN.head() # 预览前五行
y_pred_KNN.value_counts() # 预测结果数据分布
y_pred_KNN.hist(color="darkcyan") # 绘制直方图查看分布
plt.title('KNN',fontsize=14)
```

```
print('准确度:',accuracy_score(y_test,y_pred_KNN)) #评估准确度
print('R2:',r2_score(y_test,y_pred_KNN))#R
print('RMSE:',np.sqrt(mean_squared_error(y_test,y_pred_KNN)))
print('MAE:',mean_absolute_error(y_test,y_pred_KNN))
print(classification_report(y_test,y_pred_KNN))

#模型三：随机森林
model_RF = RandomForestClassifier(n_estimators=80,max_depth=8) # 实力化逻辑回归模型
model_RF = model_RF.fit(X_train,y_train) # 拟合

y_pred_RF = model_RF.predict(X_test)
y_pred_RF=pd.DataFrame(y_pred_RF) # 转化为 DataFrame 形式
y_pred_RF.head() # 预览前五行
y_pred_RF.value_counts() # 预测结果数据分布
y_pred_RF.hist(color="darkcyan") # 绘制直方图查看分布
plt.title('随机森林',fontsize=14)

print('准确度:',accuracy_score(y_test,y_pred_RF)) #评估准确度
print('R2:',r2_score(y_test,y_pred_RF))#R
print('RMSE:',np.sqrt(mean_squared_error(y_test,y_pred_RF)))
print('MAE:',mean_absolute_error(y_test,y_pred_RF))
print(classification_report(y_test,y_pred_RF))


#模型四：GBDT 分类
model_GBDT = GradientBoostingClassifier(n_estimators=100,max_depth=4,learning_rate=0.1)
model_GBDT = model_GBDT.fit(X_train,y_train) # 拟合

y_pred_GBDT = model_GBDT.predict(X_test)
y_pred_GBDT=pd.DataFrame(y_pred_GBDT) # 转化为 DataFrame 形式

y_pred_GBDT.head() # 预览前五行
y_pred_GBDT.value_counts() # 预测结果数据分布
y_pred_GBDT.hist(color="darkcyan") # 绘制直方图查看分布
plt.title('GBDT',fontsize=14)

# 评估
print('准确度:',accuracy_score(y_test,y_pred_GBDT)) #评估准确度
print('R2:',r2_score(y_test,y_pred_GBDT))#R
print('RMSE:',np.sqrt(mean_squared_error(y_test,y_pred_GBDT)))
print('MAE:',mean_absolute_error(y_test,y_pred_GBDT))
print(classification_report(y_test,y_pred_GBDT))

模型五 :XGB 分类
model_XGB = XGBClassifier(eval_metric= 'mlogloss',objective='multi:softmax',
                                                    use_label_encoder=False,
                                                    learning_rate =0.1,
                                                    n_estimators=100,
                                                    gamma=0,
                                                    subsample=0.8,
```

```
                                                colsample_bytree=0.8,
                                                nthread=4,
                                                scale_pos_weight=1,
                                                seed=27,
                                                verbose=False)
model_XGB.fit(X_train,y_train)
y_pred_XGB = model_XGB.predict(X_test)
y_pred_XGB=pd.DataFrame(y_pred_XGB) # 转化为 DataFrame 形式

y_pred_XGB.head() # 预览前五行
y_pred_XGB.value_counts() # 预测结果数据分布
y_pred_XGB.hist(color="darkcyan") # 绘制直方图查看分布
plt.title('XGB',fontsize=14)

# 评估
print('准确度:',accuracy_score(y_test,y_pred_XGB)) #评估准确度
print('R2:',r2_score(y_test,y_pred_XGB))#R
print('RMSE:',np.sqrt(mean_squared_error(y_test,y_pred_XGB)))
print('MAE:',mean_absolute_error(y_test,y_pred_XGB))
print(classification_report(y_test,y_pred_XGB))
```

附录五

介绍：该 NOTEBOOK 代码用于解决问题三

```
import numpy as np

# 定义费用计算函数
def calculate_cost(cbts_heat, epds_heat, hads_heat):
    cbts_cost = 200 * np.exp(0.8811 * cbts_heat)
    epds_cost = 500 * np.exp(0.6649 * epds_heat)
    hads_cost = 300 * np.exp(0.7459 * hads_heat)
    return cbts_cost + epds_cost + hads_cost

baby_238_features = X_train.loc[237, :]
baby_238_features = baby_238_features.values.reshape(1, -1)  # 转换为二维数组

# 当前行为特征为矛盾型的婴儿治疗费用
cbts_score = baby_238_features[0][2]
epds_score = baby_238_features[0][0]
hads_score = baby_238_features[0][1]

# 使用步长 0.1 搜索降低得分方案，使行为特征变为中等型
desired_behavior = 1
min_cost = calculate_cost(cbts_score, epds_score, hads_score)
min_cbts_heat = 0
min_epds_heat = 0
min_hads_heat = 0

for cbts_heat in np.arange(cbts_score, 0, -0.1):
    for epds_heat in np.arange(epds_score, 0, -0.1):
        for hads_heat in np.arange(hads_score, 0, -0.1):
            temp_baby_features = baby_238_features.copy()  # 创建临时变量用于更新特征
            temp_baby_features[0][2] = 15 - cbts_heat
```

```
                temp_baby_features[0][0] = 22 - epds_heat
                temp_baby_features[0][1] = 18 - hads_heat
                cost = calculate_cost(cbts_heat, epds_heat, hads_heat)
                if cost < min_cost:
                    predicted_behavior = xgb_model.predict(temp_baby_features)[0]  # 在临时
变量上预测行为特征
                    if predicted_behavior == desired_behavior:
                        min_cost = cost
                        min_cbts_heat = cbts_heat
                        min_epds_heat = epds_heat
                        min_hads_heat = hads_heat

print("使行为特征变为中等型的最少治疗费用:", min_cost)
print("调整方案：CBTS 降低到{}, EPDS 降低到{}, HADS 降低到{}".format(15 - min_cbts_heat,
22 - int(min_epds_heat), 18 - int(min_hads_heat)))

import numpy as np

# 定义费用计算函数
def calculate_cost(cbts_heat, epds_heat, hads_heat):
    cbts_cost = 200 * np.exp(0.8811 * cbts_heat)
    epds_cost = 500 * np.exp(0.6649 * epds_heat)
    hads_cost = 300 * np.exp(0.7459 * hads_heat)
    return cbts_cost + epds_cost + hads_cost

baby_238_features = X_train.loc[237, :]
baby_238_features = baby_238_features.values.reshape(1, -1)  # 转换为二维数组

# 当前行为特征为矛盾型的婴儿治疗费用
cbts_score = baby_238_features[0][2]
epds_score = baby_238_features[0][0]
hads_score = baby_238_features[0][1]

# 使用步长 0.1 搜索降低得分方案，使行为特征变为安静型
desired_behavior = 0
min_cost = calculate_cost(cbts_score, epds_score, hads_score)
min_cbts_heat = 0
min_epds_heat = 0
min_hads_heat = 0

for cbts_heat in np.arange(cbts_score, 0, -0.1):
    for epds_heat in np.arange(epds_score, 0, -0.1):
        for hads_heat in np.arange(hads_score, 0, -0.1):
            temp_baby_features = baby_238_features.copy()  # 创建临时变量用于更新特征
            temp_baby_features[0][2] = 15 - cbts_heat
            temp_baby_features[0][0] = 22 - epds_heat
            temp_baby_features[0][1] = 18 - hads_heat
            cost = calculate_cost(cbts_heat, epds_heat, hads_heat)
            if cost < min_cost:
                predicted_behavior = xgb_model.predict(temp_baby_features)[0]  # 在临时
变量上预测行为特征
                if predicted_behavior == desired_behavior:
```

```
                    min_cost = cost
                    min_cbts_heat = cbts_heat
                    min_epds_heat = epds_heat
                    min_hads_heat = hads_heat

print("使行为特征变为中等型的最少治疗费用:", min_cost)
print("调整方案：CBTS 降低到{}, EPDS 降低到{}, HADS 降低到{}".format(15 - min_cbts_heat,
22 - int(min_epds_heat), 18 - int(min_hads_heat)))
```

附录六

介绍：该 NOTEBOOK 代码用于解决问题四

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import matplotlib
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")                    #忽略警告信息
plt.rcParams['axes.unicode_minus'] = False
plt.rcParams['font.sans-serif'] = ['SimHei']
plt.rcParams['savefig.dpi'] = 300
myfont = matplotlib.font_manager.FontProperties(fname=r"simhei.ttf")
#建模
from imblearn.over_sampling import SMOTE
from sklearn.preprocessing import scale,LabelEncoder #用于数据预处理模块的缩放器、标签编
码
from sklearn.model_selection import train_test_split #数据集分类器 用于划分训练集和测试集
from sklearn.metrics import classification_report,accuracy_score #评估预测结果
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import GradientBoostingClassifier
from xgboost import XGBClassifier
from sklearn.model_selection import train_test_split,GridSearchCV
from sklearn.metrics import r2_score
from sklearn.metrics import mean_squared_error
from sklearn.metrics import mean_absolute_error
from collections import Counter
#最优模型
import joblib

df=pd.read_excel(r"睡眠质量数据.xlsx").iloc[:,1:]
df_2=df.iloc[:,[0,2,3.,5,6,7,-1]]
df_2
#  训练的数据
X=df_2.iloc[:-20,:-1]
y=df_2['睡眠质量等级'].iloc[:-20,].astype(int)
#要预测的数据
df_2_predict=df_2.iloc[-20:,:-1]
```

```
# 综合采样（先过采样再欠采样）
from imblearn.combine import SMOTETomek
kos = SMOTETomek(random_state=0)    # 综合采样
X,y = kos.fit_resample(X, y)
print(Counter( y))
model_GBDT = GradientBoostingClassifier(n_estimators=100,max_depth=4,learning_rate=0.1)
model_GBDT = model_GBDT.fit(X_train,y_train) # 拟合

y_pred_GBDT = model_GBDT.predict(X_test)
y_pred_GBDT=pd.DataFrame(y_pred_GBDT) # 转化为 DataFrame 形式

y_pred_GBDT.head() # 预览前五行
y_pred_GBDT.value_counts() # 预测结果数据分布
y_pred_GBDT.hist(color="darkcyan") # 绘制直方图查看分布
plt.title('GBDT',fontsize=14)

# 评估
print('准确度:',accuracy_score(y_test,y_pred_GBDT)) #评估准确度
print('R2:',r2_score(y_test,y_pred_GBDT))#R
print('RMSE:',np.sqrt(mean_squared_error(y_test,y_pred_GBDT)))
print('MAE:',mean_absolute_error(y_test,y_pred_GBDT))
print(classification_report(y_test,y_pred_GBDT))
model_XGB = XGBClassifier(eval_metric= 'mlogloss',objective='multi:softmax',
                                                    use_label_encoder=False,
                                                    learning_rate =0.1,
                                                    n_estimators=100,
                                                    gamma=0,
                                                    subsample=0.8,
                                                    colsample_bytree=0.8,
                                                    nthread=4,
                                                    scale_pos_weight=1,
                                                    seed=27,
                                                    verbose=False)
model_XGB.fit(X_train,y_train)
y_pred_XGB = model_XGB.predict(X_test)
y_pred_XGB=pd.DataFrame(y_pred_XGB) # 转化为 DataFrame 形式

y_pred_XGB.head() # 预览前五行
y_pred_XGB.value_counts() # 预测结果数据分布
y_pred_XGB.hist(color="darkcyan") # 绘制直方图查看分布
plt.title('XGB',fontsize=14)

# 评估
print('准确度:',accuracy_score(y_test,y_pred_XGB)) #评估准确度
print('R2:',r2_score(y_test,y_pred_XGB))#R
print('RMSE:',np.sqrt(mean_squared_error(y_test,y_pred_XGB)))
print('MAE:',mean_absolute_error(y_test,y_pred_XGB))
print(classification_report(y_test,y_pred_XGB))
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
```

```
from sklearn.metrics import accuracy_score, classification_report

# Assuming you have X (feature matrix) and y (target vector) already defined

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.1)

# Create and fit the k-NN model with k=5 (you can choose a different value for k)
model_knn = KNeighborsClassifier(n_neighbors=5)
model_knn.fit(X_train, y_train)

# Predict on the test set
y_pred_knn = model_knn.predict(X_test)

# Convert predictions to DataFrame
y_pred_knn = pd.DataFrame(y_pred_knn)

# Preview first five rows of predictions
y_pred_knn.head()

# Check data distribution of predicted results
y_pred_knn.value_counts()

# Plot histogram to visualize the distribution
y_pred_knn.hist(color="darkcyan")
plt.title('k-Nearest Neighbors', fontsize=14)

# Evaluation
print('准确度:', accuracy_score(y_test, y_pred_knn))    # Evaluate accuracy
print(classification_report(y_test, y_pred_knn))    # Generate classification report
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report

# Assuming you have X (feature matrix) and y (target vector) already defined

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.1)

# Create and fit the Logistic Regression model
model_logreg = LogisticRegression()
model_logreg.fit(X_train, y_train)

# Predict on the test set
y_pred_logreg = model_logreg.predict(X_test)

# Convert predictions to DataFrame
y_pred_logreg = pd.DataFrame(y_pred_logreg)

# Preview first five rows of predictions
y_pred_logreg.head()
```

```
# Check data distribution of predicted results
y_pred_logreg.value_counts()

# Plot histogram to visualize the distribution
y_pred_logreg.hist(color="darkcyan")
plt.title('Logistic Regression', fontsize=14)

# Evaluation
print('准确度:', accuracy_score(y_test, y_pred_logreg))    # Evaluate accuracy
print(classification_report(y_test, y_pred_logreg))    # Generate classification report
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, r2_score, mean_squared_error, mean_absolute_error,
classification_report
import matplotlib.pyplot as plt

# Create and fit the Random Forest model
model_RF = RandomForestClassifier(n_estimators=100, max_depth=4)
model_RF = model_RF.fit(X_train, y_train)

y_pred_RF = model_RF.predict(X_test)
y_pred_RF = pd.DataFrame(y_pred_RF) # Convert to DataFrame

y_pred_RF.head() # Preview the first five rows
y_pred_RF.value_counts() # Predicted result data distribution
y_pred_RF.hist(color="darkcyan") # Plot histogram to visualize the distribution
plt.title('Random Forest', fontsize=14)

# Evaluation
print('Accuracy:', accuracy_score(y_test, y_pred_RF))
print('R2:', r2_score(y_test, y_pred_RF))
print('RMSE:', np.sqrt(mean_squared_error(y_test, y_pred_RF)))
print('MAE:', mean_absolute_error(y_test, y_pred_RF))
print(classification_report(y_test, y_pred_RF))
#20 组（编号 391-410 号）婴儿的睡眠质量等级
df_predict=pd.DataFrame(model_XGB.predict(df_2_predict))
labe_bbaction={0:'差',1:'中等',2:'良好',3:'优秀'}
df_predict
df_predict_GBDT=pd.DataFrame(model_GBDT.predict(df_2_predict))
df_predict_GBDT
df_predict[0]=df_predict[0].map(labe_bbaction)
df_predict.to_csv('问题 4 预测结果.csv')
#保存最优模型
import joblib
#  将模型保存为文件
joblib.dump(model_GBDT, 'model_GBDT_问题 4.pkl')
joblib.dump(model_XGB, 'model_XGB_问题 4.pkl')
```

附录七

介绍：该 NOTEBOOK 代码用于解决问题五

```
import numpy as np
import pickle

# 定义费用计算函数
def calculate_cost(cbts_heat, epds_heat, hads_heat):
    cbts_cost = 200 * np.exp(0.8811 * cbts_heat)
    epds_cost = 500 * np.exp(0.6649 * epds_heat)
    hads_cost = 300 * np.exp(0.7459 * hads_heat)
    return cbts_cost + epds_cost + hads_cost

# 读取 GBDT 模型
model_filename = "model_GBDT_问题 4.pkl"
with open(model_filename, "rb") as f:
    gbdt_model = pickle.load(f)

baby_238_features = X_train.loc[231, :]
baby_238_features = baby_238_features.values.reshape(1, -1)  # 转换为二维数组

# 当前行为特征为矛盾型的婴儿治疗费用
cbts_score = baby_238_features[0][2]
epds_score = baby_238_features[0][0]
hads_score = baby_238_features[0][1]
sleep_quality_score = baby_238_features[0][3]  # 假设睡眠质量是第四个特征

# 使用步长 0.1 搜索降低得分方案，使行为特征和睡眠质量变为中等型
desired_behavior = 1
desired_sleep_quality = 3
min_cost = calculate_cost(cbts_score, epds_score, hads_score)
min_cbts_heat = 0
min_epds_heat = 0
min_hads_heat = 0
min_sleep_quality_heat = 0

for cbts_heat in np.arange(cbts_score, 0, -0.1):
    for epds_heat in np.arange(epds_score, 0, -0.1):
        for hads_heat in np.arange(hads_score, 0, -0.1):
            for sleep_quality_heat in np.arange(sleep_quality_score, 0, -0.1):
                temp_baby_features = baby_238_features.copy()  # 创建临时变量用于更新
特征
                temp_baby_features[0][2] = 15 - cbts_heat
                temp_baby_features[0][0] = 22 - epds_heat
                temp_baby_features[0][1] = 18 - hads_heat
                temp_baby_features[0][3] = 10 - sleep_quality_heat
                cost = calculate_cost(cbts_heat, epds_heat, hads_heat)
                if cost < min_cost:
                    predicted_behavior = xgb_model.predict(temp_baby_features)[0]  # 在
临时变量上预测行为特征
                    predicted_sleep_quality = desired_sleep_quality - int(sleep_quality_heat)
                    if predicted_behavior == desired_behavior and predicted_sleep_quality
== desired_sleep_quality:
```

```
                                min_cost = cost
                                min_cbts_heat = cbts_heat
                                min_epds_heat = epds_heat
                                min_hads_heat = hads_heat
                                min_sleep_quality_heat = sleep_quality_heat

print("使行为特征和睡眠质量变为中等型的最少治疗费用:", min_cost)
print("调整方案：CBTS 降低到{}, EPDS 降低到{}, HADS 降低到{}, 睡眠质量降低到{}".format(15 - min_cbts_heat, 22 - int(min_epds_heat), 18 - int(min_hads_heat), 10 - int(min_sleep_quality_heat)))
```