第九届湖南省研究生数学建模竞赛承诺书

我们仔细阅读了湖南省高校研究生数学建模竞赛的竞赛规则。

我们完全明白，在竞赛开始后参赛队员不能以任何方式（包括电话、电子邮件、网上咨询等）与队外的任何人（包括指导教师）研究、讨论与赛题有关的问题。

我们知道，抄袭别人的成果是违反竞赛规则的，如果引用别人的成果或其他公开的资料（包括网上查到的资料），必须按照规定的参考文献的表述方式在正文引用处和参考文献中明确列出。

我们完全清楚，在竞赛中必须合法合规地使用文献资料和软件工具，不能有任何侵犯知识产权的行为。否则我们将失去评奖资格，并可能受到严肃处理。

我们郑重承诺，严格遵守竞赛规则，以保证竞赛的公正、公平性。如有违反竞赛规则的行为，我们将受到严肃处理。

我们授权湖南省研究生数学建模竞赛组委会，可将我们的论文以任何形式进行公开展示（包括进行网上公示，在书籍、期刊和其他媒体进行正式或非正式发表等）。

所属学校和学院（请填写完整的全名）：国防科技大学系统工程学院

参赛队员（打印后签名）：1. 谭云雪

2. 张泽

3. 黎磊

指导教师或指导教师组负责人（打印后签名）：蒋平

日期： 2024 年 07 月 01 日

（请勿改动此页内容和格式。以上内容请仔细核对，如填写错误，论文可能被取消评奖资格。）

第九届湖南省研究生数学建模竞赛

题　目：基于混合学习模型的传感器数据特征提取与目标匹配一致性研究

摘要：针对问题一：设计了基于自监督学习的集成聚类算法对无标签样本进行聚类分析。首先，利用时域、频域数据采用**特征工程**对时间序列数据进行特征提取，构造了具有 **168 个属性维度的样本数据**。定义样本均衡度指标，传统聚类算法无法实现 12 类、每类 5 个样本的聚类指标，并且具有较大偏差。因此，我们在**集成聚类算法**的基础，引入了自监督学习。使用**简单对比学习框架（Simple Framework for Contrastive Learning，SimCLR）**进行样本增强，基于集成聚类算法对样本进行聚类。为了解决自监督学习模型的不稳定性问题，采样多轮次聚类的结果，以**赋予强关联性样本奖励**的形式，基于多轮聚类的结果构**造样本相似度矩阵**。然后，采用谱聚类**挖掘相似度矩阵中关联度高的子矩阵块**，将对应的样本划分为同一个活动类别。最后，为了均衡样本数量，利用所得的相似度矩阵提出了一种**样本均衡算法**，保证调整不同类别样本数量过程中，保证同一类别关联度高，不同类别关联度低。实验表明，相较于传统聚类算法，基于 SimCLR 的集成聚类算法在单次聚类（未进行相似度矩阵构造和各类别样本数量调整）结果上，样本均衡度指标明显更优。经过多次聚类构建相似度矩阵、挖掘关联子矩阵块后，以热力图的形式展现了不同活动类别之间的差异性。

针对问题二：设计了基于遗传算法的关键特征选择方法，采用多类分类器对样本进行类别判断。为解决高维特征小样本分类可能导致的维度灾难、过拟合等问题，采用遗传算法对 168 维进行关键特征提取，找到了拐点对应的关键特征，提高了模型的分类效果。同时，对比和验证了多种传统降维技术如主成分分析、t-分布随机邻域嵌入等的结果。使用问题一中聚类模型对已知标签的 10 人聚类后，各个类别子集划分的正确率在 80%以上，验证了问题一中方法的有效性，同时，发现聚类模型在**坐下（8）、站立（9）、坐电梯上（11）、坐电梯下（12）**这四个动作上混合程度较高，无法做到有效区分，其他动作分类较为准确，基于 SimCLR 模型的集合聚类算法所有人员活动类别的**准确度均在 80%以上**。且模型对于不同类别的混淆率较低，问题二中给出了不同类别的混淆矩阵

针对问题三：我们采用**基于传感器特征矩阵的 4 通道 CNN 模型**实现人物画像任务中对身高，体重，年龄等指标的**回归预测**，得到年龄、体重的预测偏差幅度均能达到±1 单位长度以内，具有较高的准确性和稳定性，对于身高的预测偏差相对较大，分析原因在于身高的影响较小，传感器数据预测误差大，因此，基于年龄和体重的预测值进行反向匹配，得到模型预测五位未知人员的 ID。在此基础上，为进一步验证结果的准确性以及与第二问模型的一致性，我们采用**时序矩阵 CNN 与特征矩阵 CNN** 对目标人物 ID 进行**分类预测**，并与基于**集成优化**的分类器分类结果相互验证，确定附件 5 的人员 **ID:10，7，6，9，13**，且模型的训练结果一致性为 100%，且稳定性较高，通过对比分析，综合确定五位未知人员的 **ID:10，7，6，9，13。**

综合以上问题思路，进行如下总结。论文通过改进聚类算法建立分类模型实现针对运动数据的**自监督学习任务**，通过特征工程与多分类器优化实现**有监督学习任务**，并进一步通过多种 CNN 回归预测模型与集成优化分类预测模型与实现了基于运动数据与目标**关键特征匹配**的人物画像任务。通过在多任务场景下综合运用上述多种模型，实现了三类模型的一致性，具有较好的迁移性与可应用型。

关键词：自监督聚类，多分类器判别，CNN，回归预测，特征提取，目标匹配

目录

1 问题综述

1.1 问题背景

随着信息时代的到来以及传感器技术的不断发展和日益成熟，基于传感器的运动状态识别目前已经有很多国内外学者进行了研究。主要的方向为基于视觉的运动状态识别和基于传感器的运动状态识别。前者主要是在受控环境中放置摄像头等传感器进行活动识别，但这种方法并不适用于个体记录其每天活动状态。后者主要是依靠放置在身体不同部位的可穿戴式传感器进行记录并识别状态。但额外的设备可能会限制人体的活动，妨碍日常生活。综合考虑到这些方法的数据收集过程较为繁琐复杂，同时还有额外的硬件成本，而集成了各类嵌入式传感器的智能手机则由于其便携、使用频率高的特点可以有效避免这些缺陷。因此基于智能手机内置传感器的人体运动状态已经成为一个研究热点。

目前已经有很多学者针对基于智能手机的传感器进行了研究，主要的模型仍集中于对各类传感器所收集的数据进行分析处理。文献[1–3]基于单个加速度传感器进行人体动作识别算法，但单个传感器数据所能识别的运动种类仍比较少，且不精确。文献[4]采用多传感器进行运动模式识别，包括 5 个加速度计和 2 只足底压力测试鞋，但是建立的模型对于仅靠智能手机内置传感器收集的数据并不通用。文献[5,6]基于智能手机内置的传感器进行人体运动状态识别，但是能够识别的种类较少，仅有 5、6 种。文献[7]所能够识别的种类较多，但是所构建的模型为有标签学习分类模型，并不能很好的使用于无标签分类问题。因此，我们将在仅依靠智能手机内置传感器的基础上，建立能够用于无标签学习分类和有标签学习分类的模型。

1.2 问题提出

基于传感器数据进行运动状态识别，往往由数据处理、特征提取、分类学习、模型训练、预测判别等内容构成。本题提出的三个问题依次从无标签分类学习、有标签分类学习、以及多任务回归预测问题。

(1) 问题 1: 无监督分类学习，建立基于自监督原则的聚类模型对数据进行特征学习与聚类。
(2) 问题 2: 有监督分类学习，建立适当的特征集合进行学习，并用训练好的判别模型对未知数据进行判别。
(3) 问题 3: 通过回归预测与分类预测，实现基于传感器数据的目标特征学习与匹配，应用于实际场景的人物画像需求。

1.3 资料条件

- 附件 1、2、3、5 中提供了不同实验人员各自的活动状态数据，其中每条数据包含加速度计、陀螺仪的三个坐标轴（acc_x, acc_y, acc_z, gyro_x, gyro_y, gyro_z）的时序数据，并以 100HZ 的频率进行记录若干条。
- 附件 1 中包含 3 名实验人员各自的 60 组数据，**并未给出每组数据对应的活动状态**。
- 附件 2 中包含 10 名实验人员各自的 60 组数据，并给出了每组数据对应的活动状态。
- 附件 3 中包含 1 名实验人员的 30 组数据，**并未给出每组数据对应的活动状态**。
- 附件 4 中包含 13 名实验人员的身高、体重和年龄。
- 附件 5 中包含 5 名**未知人员**的 30 组数据，**并未给出每组数据对应的活动状态**。

2 模型假设与符号说明

2.1 模型基本假设

(1) 假设不同人群的运动行为状态存在一定的相似性和差异性，可以通过模型学习进行分类识别；
(2) 加速度计和陀螺仪的三个轴向中，X 轴沿重力方向，Y 轴沿实验人员前进方向，Z 轴与 XY 平面垂直指向身体一侧；
(3) 假设不同的活动之间存在可区分的特征，以便于机器学习算法进行分类。
(4) 假设不同人群的身高、体重和年龄数据相互独立。

2.2 符号说明

本文定义了如下主要符号，其余符号在首次使用时进行了说明与解释。

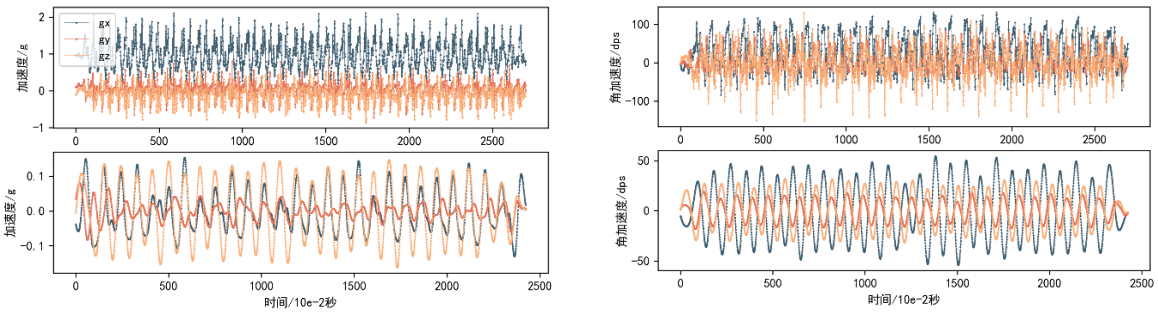
表1 符号说明

符号	含义	单位
α	线性加速度信号	
β	角加速度信号	
N	每组活动状态记录的信号数据数量	
f	传感器记录信号的频率	HZ
α^X、α^Y、α^Z、α^{Mag}	X，Y，Z 三轴线性加速度以及幅值	m/s^2
β^X、β^Y、β^Z、β^{Mag}	X，Y，Z 三轴角加速度以及幅值	$1/s^2$

3 数据预处理

3.1 数据清洗、滤波与去噪

由于题中所给的人员运动状态数据均由传感器采集，其中包含噪声，为避免噪声干扰特征提取，影响识别的准确率和泛化能力，对数据进行去噪处理。我们通过中值滤波器和 Butter-Worth 滤波器进行降噪预处理。



(a)加速度　　(b)角加速度

图1 降噪处理对比

3.2 数据扩展

题中提供的每名实验人员的数据类型仅有六种，分别为手机在三个轴向（X，Y，Z）的线性加速度和角加速度变化情况。对此，我们额外引入与三轴线性加速度和角加速度信号相对应的两个幅值信号：

$$x^{Mag}=\sqrt{\left(x^{X}\right)^{2}+\left(x^{Y}\right)^{2}+\left(x^{Z}\right)^{2}},\quad x=\alpha,\beta \tag{1}$$

该幅值信号的引入可以反应线性加速度和角加速度的总体变化趋势，在对各个轴信号进行单独分析的基础上添加了总体信息的分析，可以有效提高特征提取的鲁棒性[7]。

4 问题一：基于 SimCLR 的集成聚类算法

针对给定样本类别和类别数量的无标签分类问题，本节设计了基于自监督学习的集成聚类算法对无标签样本进行聚类分析。首先，对运动传感器采集的信号数据，从统计特征、时序、频域、和时-域等方面进行**特征工程**构造，挖掘数据的内在模式和信息特征；其次，引入了自监督学习的对比学习框架—**简单对比学习框架（Simple Framework for Contrastive Learning, SimCLR）**，从无标签数据中学习有意义的特征，对样本使用**集成聚类算法**聚类；然后，为了解决样本数据的高维特征以及 SimCLR 单次输出的差异带来的不稳定性，采用前述聚类框架对样本进行多轮聚类，基于聚类结果设计了样本相似度矩阵来表征不同样本之间的关联关系；最后，采用**谱聚类算法（Spectral Clustering）**挖掘相似度矩阵中具有较高关联度的子矩阵块，得到一个较为稳定的聚类结果，同时，设计了一种**样本均衡算法**来，根据相似度矩阵，在尽可能保证每类活动内部强相关的基础上，调整每个类别的样本数量，使之分布更加均衡。

所设计算法的有效性和聚类效果通过消融实验、单次聚类样本的均衡度、相似性矩阵以及热力图来描述和说明。值得注意的是，由于本节是主要解决的是无标签聚类问题，因此，各个样本之间的关联关系是本节挖掘的重点，所提算法的准确度将在问题（二）中进一步阐述。本节的组织架构为问题分析、特征工程、SimCLR 与集成聚类算法、实验结果分析。

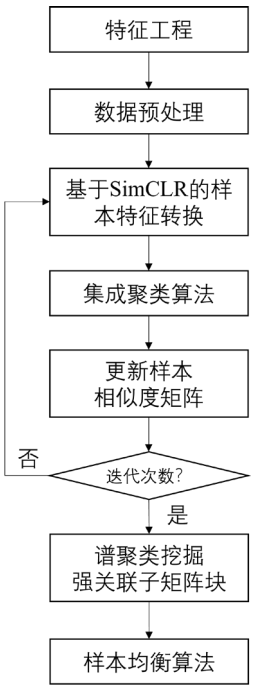


图2 问题一思路框架

4.2 问题分析

题目中给出了 3 名实验人员各自 60 组活动数据，要求将每个人的 60 组数据划分为 12 类，并且每类中有 5 组数据。但并未给出十二种活动状态所对应的数据特征，所以该问题本质上为无标签分类问题，即聚类问题。

为了达到更好的聚类效果，需要提取数据的隐含特征和内部结构，即特征工程。原始数据设计三轴加速度计和陀螺仪采集的时序数据，在此基础上，通过傅里叶变换能够将时序数据转换为频域数据，进一步能够得到时域和频域数据的统计特征。由于该领域已经进行了较为深入的研究，能够从大量的参考文献中参考借鉴相关指标。基于特征工程构造的特征中，可能存在部分作用重叠或者差异性较低的指标，因此需要对主要特征进行筛选，常用的方法有主成分分析等，这一步在数据预处理中完成。

考虑到现有特征的纬度过高、样本数量较少等特点，现有的经典聚类算法容易由于纬度爆炸而出现聚类失衡等问题，而且难以均衡不同活动样本之间的类别数量。而自监督学习作为一种不依赖于人工标注数据的训练方法，在深度特征预处理上具有很好的性能，能够利用数据本身的信息进行监督，扩大样本内部关系的差异性，输出更高维度的样本数据来提升聚类效果。同时，为了避免单一聚类模型效果失衡等情况的发生，我们使用多个聚类模型以获得更稳健和更准确的聚类效果。

进一步，为了刻画样本之间的关联关系和减少自监督学习对数据处理时出现波动或者聚类模型效果失衡的不利影响，基于采样的思想，对样本进行多轮次自监督学习和聚类，根据每一轮此次样本之间的聚合关系构建样本的相关性矩阵，当采样次数达到一定程度，那么相似度矩阵中也会包含稳定的样本关系。为了挖掘样本之间的关联关系，采用谱聚类算法找到具有较强关系的子矩阵块，每块矩阵对应了一类活动。

最后，由于聚类后并不能保证所有类别的样本数量相等，因此，需要对每个类别的样本进行调整。在这一步中，我们通过定义相关的距离指标，删除同一类别中相关性弱的样本，融合相关性强的样本，设计了样本均衡算法来满足题设的分类要求。

4.3 特征工程

将每组数据从 6 维扩充到 8 维后，我们获得了更充分的初始数据。但是这些数据中包含有大量的冗余信息，直接利用这些信息进行分类可能会导致模型复杂度过高、难以训练和泛化，而进行特征提取可以突出原始信号中与分类任务相关的关键信息，可以避免这些缺陷并提高模型的鲁棒性和分类的稳定性。

由于传感器记录的是实验人员数秒内的活动，因此每组数据都为时序数据。同时由于采样频率高，数据量较为充足，所以我们综合考虑数据的统计特征、时序特征、频域特征、时-域特征，对每组活动状态的 8 个维度数据分别建立以下数据特征：

速度变化量：对离散数据采取加权求和计算出速度的改变量。

$$dv_i^x = \sum_{n=1}^{N} x_j^i / f, \quad x = \alpha, \beta \quad i = X, Y, Z, Mag \tag{2}$$

加速度峰值：加速度传感器记录到的最大加速度值。

$$pk_i^x = \max\left\{\left|x_n^i\right|, n = 1, 2, \ldots N\right\}, \quad x = \alpha, \beta \quad i = X, Y, Z, Mag \tag{3}$$

均方根值：描述数据的波动或变化程度。

$$RMS_i^x = \sqrt{\frac{\sum_{n=1}^{N}\left(x_j^i\right)^2}{N}}, \quad x = \alpha, \beta \quad i = X, Y, Z, Mag \tag{4}$$

平均值：数据的平均值。

$$\bar{x}^i = \sum_{n=1}^{N} x_n^i, \quad x = \alpha, \beta \quad i = X, Y, Z, Mag \tag{5}$$

标准差：反映数据的波动程度。

$$\sigma_i^x = \sqrt{\frac{\sum_{n=1}^{N}\left(x_j^i - \overline{x}^i\right)}{N}}, \quad x = \alpha, \beta \quad i = X, Y, Z, Mag \tag{6}$$

最大值：数据的最大值。

$$x_{\max}^i = \max\left\{x_n^i, n = 1, 2, \ldots, N\right\}, \quad x = \alpha, \beta \quad i = X, Y, Z, Mag \tag{7}$$

最小值：数据的最小值。

$$x_{\min}^i = \min\left\{x_n^i, n = 1, 2, \ldots, N\right\}, \quad x = \alpha, \beta \quad i = X, Y, Z, Mag \tag{8}$$

极差：数据中最大值和最小值之间的差异。

$$R_i^x = x_{\max}^i - x_{\min}^i, \quad x = \alpha, \beta \quad i = X, Y, Z, Mag \tag{9}$$

偏度：表示数据分布相对于其均值的不对称程度。

$$Skewness_i^x = \frac{N}{(N-1)(N-2)}\sum_{n=1}^{N}\left(\frac{x_n^i - \overline{x}^i}{\sigma_i^x}\right)^3, \quad x = \alpha, \beta \quad i = X, Y, Z, Mag \tag{10}$$

峰度：描述数据分布的尾部厚度和尖峭程度。

$$Kurtosis_i^x = \frac{N(N+1)}{(N-1)(N-2)(N-3)}\sum_{n=1}^{N}\left(\frac{x_n^i - \overline{x}^i}{\sigma_i^x}\right)^4 - \frac{3(N-1)^2}{(N-2)(N-3)}, \quad x = \alpha, \beta \quad i = X, Y, Z, Mag \tag{11}$$

峰度因子：评估潜在尾部风险的可能性。

$$CrestFactor_i^x = \frac{\left|x_{\max}^i\right|}{RMS_i^x}, \quad x = \alpha, \beta \quad i = X, Y, Z, Mag \tag{12}$$

裕度因子：对潜在风险的容忍程度。

$$MarginFactor_i^x = \frac{\left|x_{\max}^i\right|}{P_{RMSi}^{\ x}}, \quad x = \alpha, \beta \quad i = X, Y, Z, Mag \tag{13}$$

波形因子：描述周期性信号的无量纲参数。

$$WaveformFactor_i^x = \frac{N \cdot RMS_i^x}{\sum_{n=1}^{N}\left|x_n^i\right|}, \quad x = \alpha, \beta \quad i = X, Y, Z, Mag \tag{14}$$

脉冲指数：分析数据的脉冲或短暂的数据剧烈变化。

$$Impulse_i^x = \frac{\max\left\{\left|x_n^i\right|, n = 1, 2, \ldots, N\right\}}{\sum_{n=1}^{N}\left|x_j^i\right|}, \quad x = \alpha, \beta \quad i = X, Y, Z, Mag \tag{15}$$

四分位频率：帮助识别数据的分布形状。

$$Q_{1i}^x = \frac{N+1}{4}x^i, \quad x = \alpha, \beta \quad i = X, Y, Z, Mag \tag{16}$$

$$Q_{2i}^{x}=\frac{2(N+1)}{4}x^{i},\quad x=\alpha,\beta\quad i=X,Y,Z,Mag \tag{17}$$

$$Q_{3i}^{x}=\frac{3(N+1)}{4}x^{i},\quad x=\alpha,\beta\quad i=X,Y,Z,Mag \tag{18}$$

频率变异系数：描述数据频率分布的离散程度。

$$fcv_{i}^{x}=\frac{\sigma_{i}^{x}}{f},\quad x=\alpha,\beta\quad i=X,Y,Z,Mag \tag{19}$$

频谱峰度：描述频谱的尖峭程度。

$$SpectralKurtosis_{i}^{x}=\frac{\sum_{n=1}^{N}\left(x_{n}^{i}-\overline{x}^{i}\right)^{4}/N}{\left(\sum_{n=1}^{N}\left(x_{n}^{i}-\overline{x}^{i}\right)^{2}/N\right)^{2}},\quad x=\alpha,\beta\quad i=X,Y,Z,Mag \tag{20}$$

频谱偏度：描述频谱分布的对称性。

$$SpectralSkewness_{i}^{n}=\frac{\sum_{n=1}^{N}\left(x_{n}^{i}-\overline{x}^{i}\right)^{3}}{N},\quad x=\alpha,\beta\quad i=X,Y,Z,Mag \tag{21}$$

频谱熵：量化频谱的复杂性和不确定性。

$$SpectralEntropy_{i}^{x}=-\sum_{n=1}^{N}x_{n}^{i}\log_{2}x_{n}^{i},\quad x=\alpha,\beta\quad i=X,Y,Z,Mag \tag{22}$$

经过上述特征指标的构建，我们将每组的 8 维数据通过提取特征转化为 8×21=168 维的向量。每个实验人员的 60 组数据也转化为 60×168 的二维矩阵。

我们将采取自监督对比学习训练(Class Simple Constrastive Learning Representation, SimCLR)对数据进行聚类，从而将每位实验人员的活动数据进行分类。

4.4 SimCLR 与集成聚类算法

SimCLR 是一种用于自监督学习的对比学习框架。SimCLR 的核心思想是通过对比学习，从无标签数据中学习有意义的特征。集成聚类的思想类似集成学习，融合多个聚类模型的结果增强效果，相较于单一聚类模型具有较好的稳定性。除了上述两个主要模块以外，问题一较为重要的组件还有相似度矩阵的构建和样本均衡算法，下面对它们展开介绍。

4.4.1 SimCLR

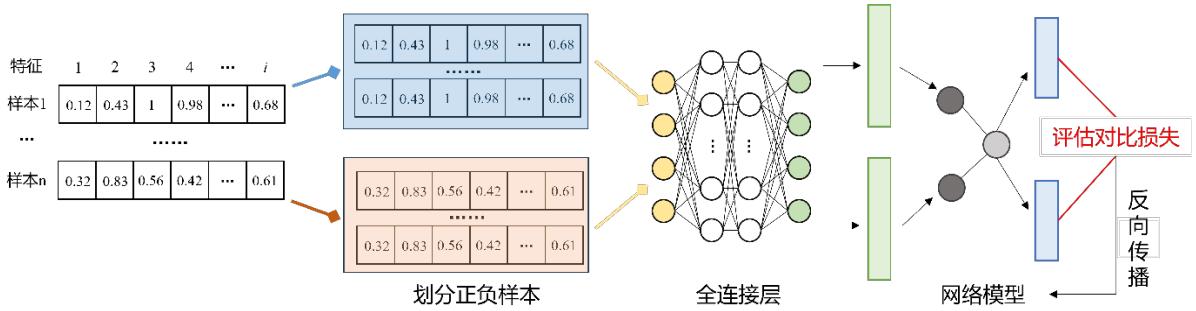


图3 SimCLR 框架

该方法的步骤为：

1）输入样本数据，该样本数据经过了前期的预处理、特征工程等处理；

2）数据增强：对每组数据进行变换，构建正负样本；

3）使用非线性全连接层进行映射；

4）计算增强后的数据经过映射后的余弦相似度；

5）通过反向传播算法更新编码器和投影头的参数，以最小化损失函数，并在训练过程中逐渐优化参数；

6）训练完成后将编码器输出的表征作为聚类参数的输入进行聚类。

其中，步骤 1）-5）在图 2 中进行了展示，划分为正负样本以后，通过多个全连接层的组合，使模型能够捕获输入数据中的复杂关系和模式，提取出更高层次的特征表示。结合非线性激活函数，增加模型的非线性特征提取能力，有助于处理复杂且非线性的数据分布，不同大小的输出维度，可以对输入数据进行降维或升维。

对比损失函数可以从两个角度来解释：1、当来自相同输入图像的增强图像投影相似时，对比损失减小；2、对于两个增强的正样本对 i 和 j，i 的对比损失试图在同一个 batch 中的其他图像(“负”样本)中识别出 j。

对正样本对 i 和 j 的损失的形式化定义为：

$$L=\sum_{i\in I}\sum_{j\in P(i)}-log\frac{exp(\mathbf{sim}(\mathbf{z}_i,\mathbf{z}_j)/\tau)}{\sum_{k\in A(i)}exp(\mathbf{sim}(\mathbf{z}_i,\mathbf{z}_k)/\tau)} \tag{23}$$

最终的损失是 batch 中所有正样本对损失的算术平均值：

$$\mathcal{L}=\frac{1}{2N}\sum_{k=1}^{N}\left[\ell(2k-1,2k)+\ell(2k,2k-1)\right] \tag{24}$$

特征工程构造的样本经过 SimCLR 映射转换后，会得到一组新的样本特征，该样本特征会把不同样本之间的差异性放大，为集成聚类聚类算法提供输入。

4.4.2 集成聚类算法

针对无标签数据进行分类已经有较多的算法研究，主要的方法有凝聚层次聚类（Agglomerative Hierarchical Clustering, AHC）算法、高斯混合模型（Gaussian Mixture Model, GMM）聚类算法、K-Means 聚类算法、谱聚类（Spectral Clustering, SC）算法等。集成聚类算法主要综合考虑多个聚类算法的效果，得出一个相对而言可能更加好的结果。本文集成算法中采用的聚类算法为上述 4 种。

AHC 算法：不需要预先指定数据的标签，而是基于数据点之间的相似度或距离来构建数据的层次结构。该方法的步骤为：

1）将每条数据视为一个独立的簇，并计算不同数据点之间的距离矩阵；

2）合并簇：选择最近的两个簇进行合并，并更新距离矩阵；

3）重复步骤 2，直到达到预定簇的数量；

GMM 模型：该算法将数据集看作是由多个高斯分布组成的混合模型，每个高斯分布代表一个聚类簇。该方法的步骤为：

1）选择聚类个数 k；

2）随机选择 k 个聚类中心，并初始化每个高斯分布的均值和协方差矩阵；

3）计算每个数据点属于每个高斯分布的后验概率；

4）根据期望步骤的结果，通过最大化似然函数来更新每个高斯分布的参数；

5）重复 3）和 4）直到模型参数收敛；

6）根据最终的高斯分布参数和后验概率将数据点分配到概率最大的聚类簇中。

K-Means 聚类算法：在没有预先标注的数据集中发现数据内在的分组结构，将数据集分成 K 个簇，使得每个数据点与其分配的簇中心之间的平方距离之和最小。该方法的步骤为：

1）随机选择 k 个数据点作为初始簇中心；

2）计算每个数据点与每个簇中心的距离，并将其分配给距离最近的簇；

3）重新计算每个簇的中心；

4）重复步骤 2）和 3），直到簇中心的变化小于某个阈值或达到预设的迭代次数；

SC 算法：该算法基于图论，将数据点视为图中的点，样本之间的相似度视为图中的边，通过分析数据点之间的相似性矩阵（亲和矩阵）的特征向量来进行聚类。该方法的步骤为：

1）构建相似性矩阵和度矩阵；

2）基于 1）中的两个矩阵构建拉普拉斯矩阵，并计算特征向量；

3）将特征向量进行聚类，并输出相应的聚类结果。

上述 4 种算法集成结果的方式与相似度矩阵的构建过程相关，将在下一节介绍，主要方式为对聚类结果进行评价，根据该结果不断更新样本之间的关联奖励。

4.4.3 相似度矩阵构建

由于 SimCLR 模型会存在一定的不稳定性，所以可能输出的新的样本特征可能放大部分样本特征或者错分样本特征，而集成聚类的效果又会被样本特征所影响。因此，合理地利用多次 SimCLR 和集成聚类的信息，不断记录、强化其中的频繁关系模式，最终能够得到一个相对稳定的样本相关性矩阵，该矩阵会记录样本与样本之间的关系，能够通过关联性挖掘等方式，发现其中的强相关的样本块。相似度矩阵的定义和构建如下：

本文所构建的相似度矩阵 ***Rel*** 是一个 $N\times N$ 的对称矩阵，其中 N 为样本数量，a_{1n} 为样本 1 和样本 n 最终的累计关联奖励，样本自身不能关联，对角线元素为 0：

$$Rel_{n\times n}=\begin{pmatrix} 0 & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & 0 \end{pmatrix} \tag{25}$$

相似度矩阵构建的流程如下：

步骤 1：初始化 SimCLR 的训练次数、学习率、温度等参数；

步骤 2：将预处理好的样本特征数据输入到 SimCLR，得到新的样本特征属性；

步骤 3：使用集成聚类算法对新的样本特征属性进行聚类；

步骤 4：对划分到同一子集的样本 i 和 j 给予奖励，该奖励累加到 a_{ij}；

步骤 5：终止条件判断，回到步骤 1，否则输出相似度矩阵.

为了挖掘样本关联关系的一般状态，在步骤 4 中我们对聚类结果给予一定的奖励，该奖励的定义如下：

$$reward = \begin{cases} r, & if\ k == 5, \\ r / k & if\ k > 5, \\ r / (10 - k) & otherwise \end{cases} \tag{26}$$

其中 k 为被划分为同一类活动的子集的数量大小。由于题设已经给出每类活动的实际的样本数量为 5，因此，此处的奖励设置与该信息相关。更一般情况下的设置应该为 r / k，被划分到同一子集的样本数量越少，划分错误的概率越小；具有强相关性的大的样本集合也不会因为奖励过小而造成采样失真，因为它们会被经常划分到同一个子集。

4.4.4 基于相似度矩阵的样本均衡算法

在上述步骤的基础上，得到相似度矩阵以后，可以使用谱聚类算法对矩阵进行挖掘，该过程与常规聚类算法一致，能够使用工具包完成，此处不再赘述。下面，将介绍一种样本均衡算法，该算法利用相似度矩阵对每个样本子集的数量进行调整，均衡样本集合数量的同时，保证内部具有较强的关联性。算法步骤如下：

步骤 1：将相似度矩阵、聚类结果作为输入，moreFive=[], lessFive=[], needAllocate=[]，satisfiedResults = []；

步骤 2：将聚类结果中样本数量超过 5 的添加到 moreFive，少于 5 的添加到 lessFive, 等于 5 的样本标签集合添加到 satisfiedResults；

步骤 3：计算 moreFive 中聚类结果子集的子相似度矩阵 subRel 的行和，挑选出最小的样本标签添加到 needAllocate 直到该聚类结果的数量为 5；

步骤 4:计算 needAllocate 中所有样本到 lessFive 各样本划分集合的相关性，将相关性最高的样本标签添加到对应的样本集合；

步骤 5：删除步骤 4 中从 needAllocate 选出的样本标签，判断 lessFive 中是否有数量等于 5 的样本划分，有则删除该样本划分，无则回到步骤 4 直到 lessFive 为空；

步骤 6：输出最终的聚类结果 satisfiedResults.

在步骤 3 中，我们定义子相似度矩阵的行和为聚类结果中每个元素与同一划分中其他元素的关联强度，对于样本数量超过 5 的聚类集合，删除关联强度最小的样本直到满足数量要求。而后，需要考虑被删除样本的重划分。假设 needAllocate = [3, 4, 17], lessFive = [[5, 9, 2], [6, 8, 1, 7]]，含义为需要重新划分的样本为 3、4 和 17，样本数量少于 5 个的聚类结果有两个，计算上述 3 个样本到这两个集合的相关程度，该相关程度的定义为样本到集合中样本相似度的均方和，因此需要计算 3×2 个数值，假设最终结果为[[0, 12], [34, 4.2], [198, 35]]，则代表样本 2 与聚类结果[6, 8, 1, 7]具有很强的关联特性，因此将其分到该集合中。

通过上述步骤，能够调整样本数量不等于 5 的子集，使得被删除的样本与之前样本集合的关联性足够小，添加的样本与之后的样本集合关联性足够大。

4.5 实验结果分析

在本节中，进行各个实验对所提出算法的有效性进行验证，实验包括与基本聚类算法的对比实验、消融实验、参数的敏感性分析等，其中对比试验缺少样本标签，拟从样本相关性，各个类别的样本数量是否均衡等方面进行评价。

4.5.1 实验设置

由于现有研究中，聚类算法已经很成熟，实验方案的设置如下：首先使用预处理好的数据使用几种基本的聚类算法（集成聚类模型中的四类分别进行测试），观察样本的类内、类间距离等情况。定义样本均衡度指标如下：

$$B_a = \sum_{l=1}^{12} (num_l - 5)^2 \tag{27}$$

4.5.2 实验结果与分析

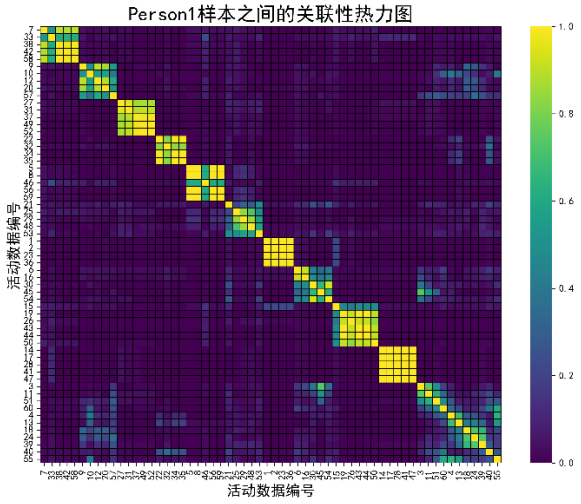
对比算法的结果：

表2 问题 1 中各算法分类结果的均衡度

数据集\算法	K-Means	AHC	GMM	SC	SimCLR
Person1	82	62	68	54	**8**
Person2	136	126	128	64	**2**
Person3	158	84	84	52	**10**

从上表结果可见，经过聚类后，能够将 60 个数据稳定的区分接近 12×5 的只有 SimCLR 算法，明显优于其他聚类算法。需要说明的是，此处的 SimCLR 算法没有使用 4.3.4 中所提到的样本均衡算法调整各个类别的样本数量，而是直接通过对比学习提取的特征进行多轮次聚类后得到的统计结果挖掘出的数据。

该对比实验验证了自监督学习在样本增强方面的作用，需要特别说明的是，特征工程过程中，没有构造关于每个类别为 5 个样本这类属性，仅对时域和频域信号进行了处理。因此，有理由相信，经过 SimCLR 架构得出的样本特征能够分区不同样本的差异性。聚类结果的准确度将在第二问中进行对比验证。基于自监督学习集成聚类所得的样本关联矩阵的热力图如图所示，将被归为同一类活动的数据按照集合顺序排列。



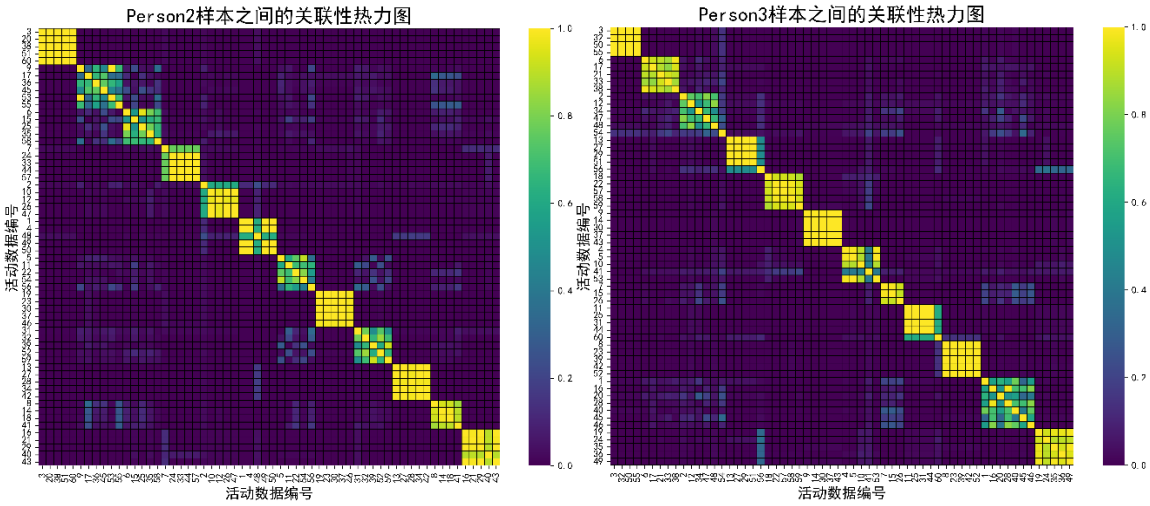


图4 基于 SimCLR 的集成聚类的样本相似度矩阵热力图

从热力图中可以明显看出，经过多轮聚类结果统计出的相关性矩阵的热力图各个模块之间具有较为明显的区分度，类内的相关性很高，但也有部分数据存在一定的混淆。在 Person1 数据集中，右下角有 10 个活动之间没有很强的关联性，其他活动可以很明显的做出区分。Person2 中，10 个活动类别具有很明显的差异性，热力值完全与其他类别有区分度。Person3 中有 1 个动作与另外一个动作相似，但是能够做出区分。

通过上述结果我们可以得出，要求分辨的 12 个动作中，有两个动作的区别度没有明显差异，容易造成混淆。但是不同人员的动作模式在这两个动作中也会表现出差异，可能会很明显，但也可能难以区分。

表3 问题 1 的分类结果

分类	Person1	Person2	Person3
第 1 类活动	1, 2, 15, 23, 36	1, 4, 48, 49, 50	1, 16, 28, 40, 46
第 2 类活动	3, 11, 51, 55, 60	2, 10, 12, 26, 47	2, 12, 34, 47, 48
第 3 类活动	4, 13, 18, 24, 39	3, 20, 38, 51, 60	3, 20, 32, 50, 55
第 4 类活动	5, 8, 46, 56, 59	5, 11, 22, 54, 56	4, 5, 10, 41, 53
第 5 类活动	6, 16, 30, 45, 54	6, 15, 25, 35, 58	6, 17, 21, 33, 38
第 6 类活动	7, 33, 38, 42, 58	7, 24, 33, 44, 57	7, 15, 26, 45, 54
第 7 类活动	9, 10, 12, 20, 57	8, 14, 17, 18, 41	8, 23, 39, 42, 52
第 8 类活动	14, 17, 28, 41, 47	9, 36, 45, 53, 55	9, 14, 30, 37, 43
第 9 类活动	19, 26, 43, 44, 50	13, 27, 28, 34, 42	11, 25, 31, 44, 60
第 10 类活动	21, 25, 29, 48, 53	16, 21, 29, 40, 43	13, 27, 29, 51, 56
第 11 类活动	22, 32, 34, 35, 40	19, 23, 30, 37, 46	18, 22, 57, 58, 59
第 12 类活动	27, 31, 37, 49, 52	31, 32, 39, 52, 59	19, 24, 35, 36, 49

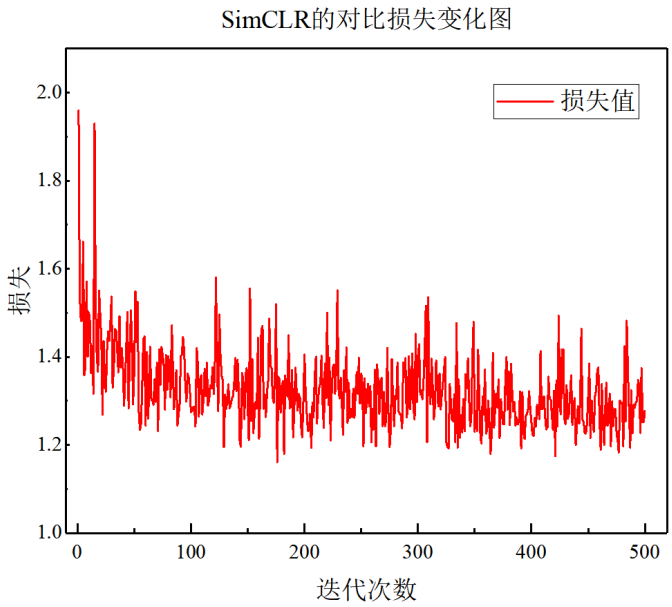


图5 基于 SimCLR 的集成聚类的样本相似度矩阵热力图

SimCLR 在某次训练过程中的对比损失随迭代进程如上，可以看到，最开始的损失值较高，此时，神经网络折射的样本空间的正负样本区分度较小，到了后期，损失值趋于稳定。

5 问题（二）基于遗传算法的 KNN 分类学习模型

问题二要求使用带标签的数据训练出一个判别模型用来预测未知数据。首先根据问题一的特征工程来构建出 600×168 的二维矩阵数据，同时增添每条数据的活动标签，从而得到 600×169 的二维矩阵训练数据。使用全部的数据进行训练时，通过多次 K 折交叉验证充分利用数据信息训练模型，为提高模型的泛化性，采用遗传算法进行特征筛选，从 168 维特征中提取出关键特征集合，同时通过比较不同分类器的准确率效果，选出最佳的 KNN 分类器训练模型，并预测未知数据。

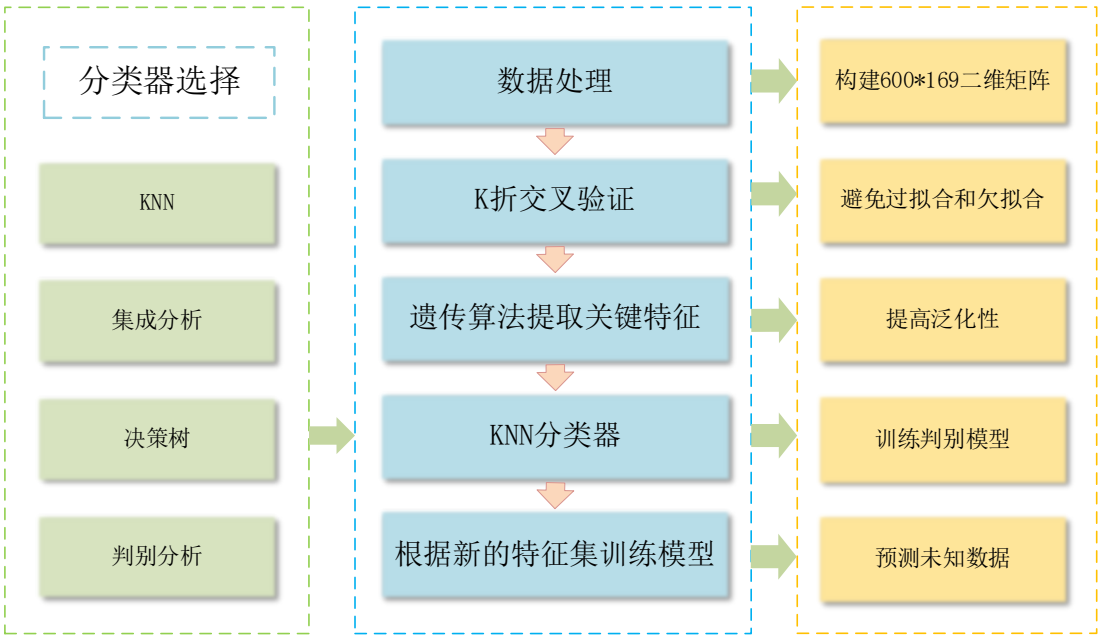


图6 问题二流程框图

5.1 问题分析

附件二为 10 个实验人员已知活动状态的 60 组实验数据，要求通过这些带标签数据训练出判别模型。这相当于建立有标签学习的分类模型，通过已知标签的数据集来训练模型，以便对未知数据进行分类。

问题（1）中要求使用问题一的分类模型对附件二中的 10 名人员的所有数据进行分类，同时比较与判别模型的结果。

问题（2）要求使用判别模型对附件三中的数据进行分类，即涉及到模型分类。这就要求对于判别模型既要有较高的准确性，又要有较好的泛化性，因此要在模型的训练上做改进。

在训练模型时，充分利用数据集是保证模型效果的前提，使用 K 折交叉验证法将所有数据集都作为测试集验证效果，可以避免欠拟合问题的出现。但 K 折的划分基于固定排序的数据，如果进行多次 K 折法并且每次均对数据进行随机排序，则可以避免单次 K 折交叉验证所带来的偶然性。

为提高判别模型对未知数据的泛化性，需要在问题一特征工程的 168 维数据基础上进行有效筛选，提取关键特征来训练模型。而可能的特征组合数量过于庞大，故采用遗传算法进行快速筛选。在递增的特征数量约束下，寻找最佳的特征组合。并根据不同特征数量约束下的模型判别准确率表现，得到合适的关键特征数量。

带标签数据的分类算法已有较多研究，通过比较分类效果和时间，选取性能表现较好的 K-近邻算法作为分类器。并根据得到的关键特征来训练模型，并对未知数据进行判别。

5.2 模型建立

5.2.1 数据处理

构建判别模型，同时用判别模型对无标签数据进行分类，即要求我们在训练模型时，要把握好训练集与测试集的划分，从而保证模型既有较高的准确性，又有较好的泛化性。

此问旨在构建有标签分类学习的判别模型。附件二所给的数据类别同样是来自加速度计和陀螺仪的三轴（X，Y，Z）加速度信号，为此，我们额外引入与三轴线性加速度和角加速度信号相对应的两个幅值信号将数据从 6 维扩充至 8 维。

同样考虑采用问题一中提出的特征集将每组数据转化为向量，这将在训练模型时提供便利。因此，本问中的数据从来自 10 名实验人员的有标签的 60 组活动状态数据转化为 600×168 的二维矩阵，标准化后再添加状态分类标签列，从而转化为 600×169 的二维矩阵。

5.2.2 特征降维技术

特征降维（Dimensionality Reduction）是一种数据预处理技术，通过减少数据集中的特征数量，从而简化数据表示，保留重要信息，同时丢弃冗余或不重要的特征。特征降维在数据分析和机器学习中具有重要作用。本节将介绍我们在实验过程中使用的几种降维技术，并给出可视化结果对当前数据的分布等进行分析。

主成分分析（Principal Component Analysis，PCA）是一种常用的线性降维技术，通过生成新的正交特征（称为主成分），保留数据集中的大部分方差信息，从而减少特征维度。其主要流程包括：

1）去中心化处理，使数据的均值为零；

2）计算协方差矩阵；

3）对协方差矩阵进行特征值分解，得到特征值和特征向量；

4）按特征值的大小降序排列，选取前k个特征向量，构成新的特征子空间；

5）原始数据投影到新的特征子空间，得到降维后的数据。

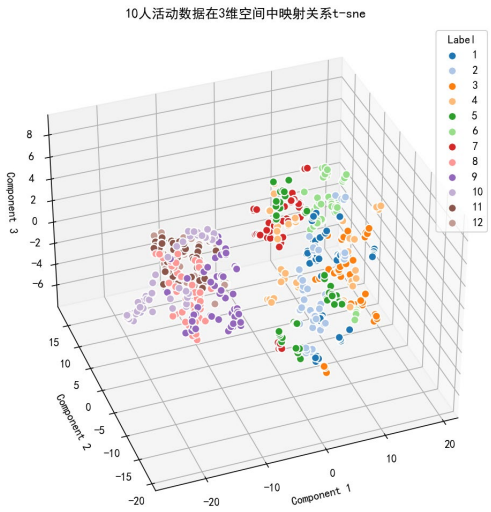
统一流形近似与投影（Uniform Manifold Approximation and Projection）是一种非线性降维技术，它利用流形学习和图论的方法，将高维数据有效地嵌入到低维空间中。主要的流程步骤包括：

1）数据标准化或归一化，以确保不同特征对距离计算的影响一致；

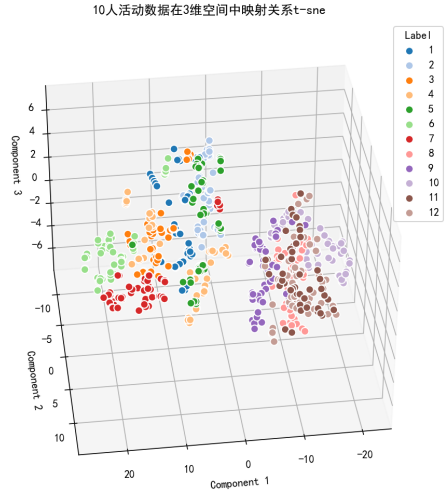
2）计算数据点之间的邻近关系，生成加权近邻图；

3）寻找低维嵌入，使其在低维空间中尽可能地保留高维空间的邻近关系。

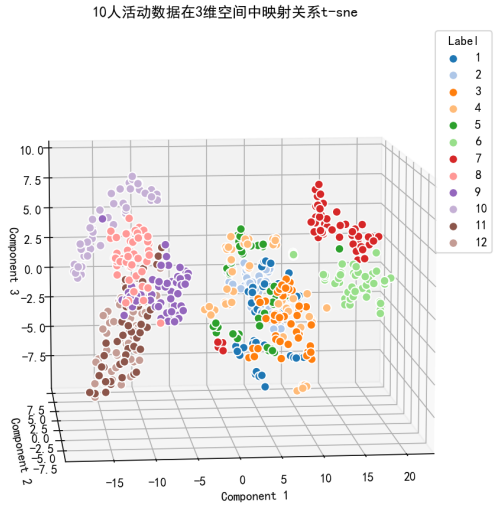
t-分布随机邻域嵌入（t-distributed Stochastic Neighbor Embedding）是一种用于降维和数据可视化的非线性技术，特别适合高维数据的低维表示。



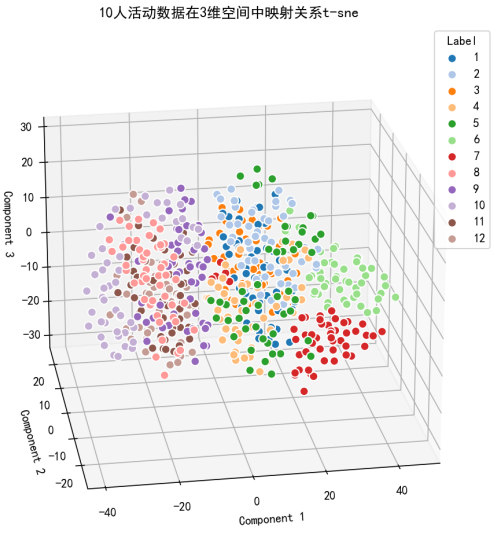
（a）3 维度空间映射 perplexity=15



（b）3 维度空间映射 perplexity=25



（c）3 维度空间映射 perplexity=50



（d）3 维度空间映射 perplexity=100

图7 问题二流程框图

经过 T-SNE 映射后，在三维空间进行可视化，可以看出数据具有很明显的分布特征。图（a）（b）（c）中动作 1-7 和动作 8-12 具有明显的差别，能够很明显的区分开。图（c）中动作 10、7 和 6 也能够与其他动作有区别。动作 11 和动作 12，动作 8 和动作 9 始终交叉在一起。

5.2.3 K 折交叉验证构建训练集和测试集

为尽量减少不同训练集和测试集对模型准确性的影响，我们采用具有较低风险过拟合和欠拟合优点的 K 折交叉验证（K-fodl cross-validation）进行分类训练。

K 折交叉验证步骤：1）将完整数据集随机打乱，并拆分为大小相近且互斥的 K 个子集；2）依次将每折子集作为测试集，其余 K-1 折子集作为训练集；3）使用训练集来训练模型，并使用测试集来评估模型的性能；4）将 K 次模型的准确率取平均值或中位数来作为模型性能的估计，并输出结果。

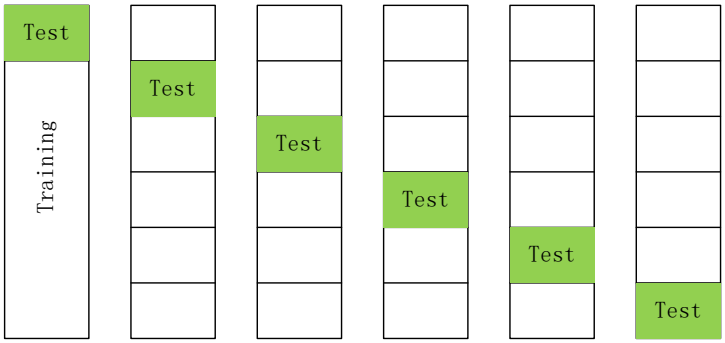


图8 K 折交叉验证示意图

在算法具体实现时，K 折交叉验证将所有数据随机打乱后划分为 K 个互斥子集，尽管将每个子集的其他所有互斥子集作为训练集依次训练，但对于数据的随机打乱仍有一定的偶然性。因此，我们采取多次 K 折交叉验证，每次均重新随机打乱数据进行交叉验证，提高训练模型的稳定性。

关于分类算法已有较多研究，主要的方法有：KNN、回归树集成、支持向量机、神经网络、决策树、判别分析等。考虑到特征指标数量和分类标签较多，我们选择效率多类分类器进行模型训练。与其他算法的实验对比将会在下一节展示。

5.2.4 基于遗传算法的特征集合筛选

在问题一中已经选取了较多的特征指标，尽管问题一的结果已经足够理想，但是对于有标签数据来说，过多的特征指标将会导致判别模型的过拟合，从而在样本数据外的泛化性较差。因此，我们考虑从 168 维特征指标中提取部分关键特征指标用来训练模型，在保证判别准确性的基础上，提高分类泛化性。

在不限定特征指标数量的前提下，所有可能的候选特征集合数量为 $\sum_{i=1}^{168} C_{168}^{i} \approx 3.7\times10^{50}$。由于候选集合数量庞大，故考虑采用遗传算法进行特征集和筛选，依次对特征数量设置不同的限制，并通过遗传算法在该限制下选取特征。以多次 K 折交叉验证的平均准确率作为适应度函数，选择出在不同限制下最佳的特征指标集合。并通过比较不同特征数量限制下判别模型的准确率，选择出最佳的特征集合来训练判别模型。

遗传算法通过模拟生物进化过程中的遗传和变异机制，在候选解空间中搜索最优解。基本步骤为：1）初始化，随机生成初始种群，并计算个体适应度；2）选择：根据个体适应度越高，选择概率越大来挑选父代个体；3）交叉变异：对选中的父代个体进行交叉

变异产生新的子代个体，并计算适应度；4）更新种群：通过精英策略保留最优个体作为新的种群；5）迭代终止：如果达到指定迭代次数，则终止算法并输出适应度最高的个体及结果，否则返回步骤 2）。

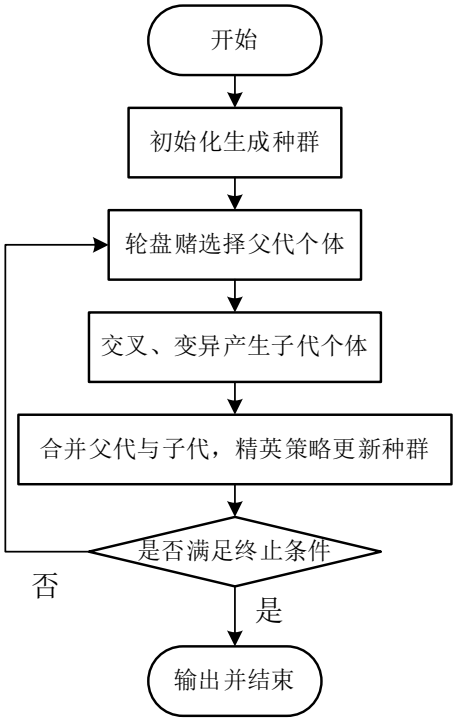


图9 遗传算法流程图

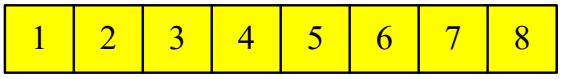


图10 不重复整数编码方式示意图

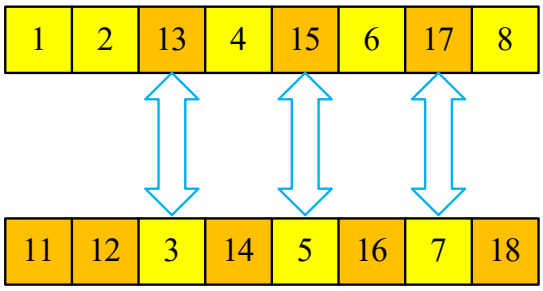


图11 个体之间的交叉示意图

图12 个体变异示意图

5.3 实验结果分析

5.3.1 分类器效果比较与特征筛选

我们分别进行多次实验，通过对特征数量施加限制，使用不同分类器作为适应度函数，在每个分类器中进行多次 K 折交叉验证，以平均准确率作为适应度值，使用遗传算法进行搜索，得到不同特征数量限制下的最佳特征集合，并比较不同分类算法的效果。具体结果如下：

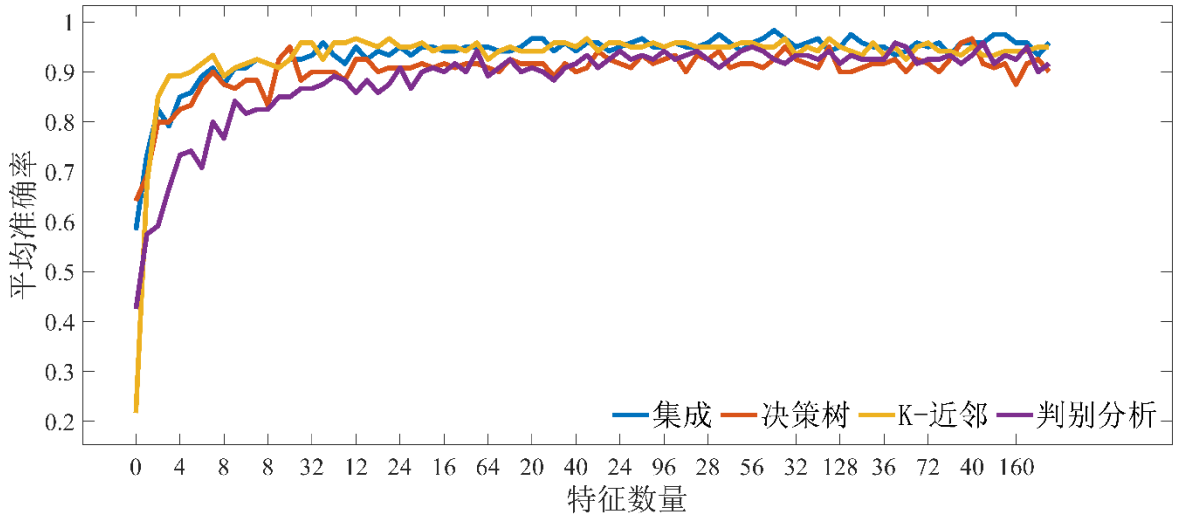


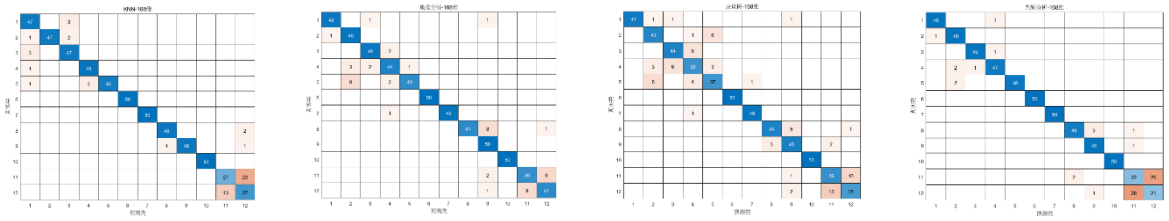
图13 不同分类器在不同特征数量下效果比较

从图中结果可以看出回归树集成与 K-近邻算法效果要优于判别分析和决策树分类。由于需要减少特征数量以避免判别模型的过拟合，所以我们选择在低维度更加准确的 KNN 算法作为分类器。所选出的最佳特征集合为

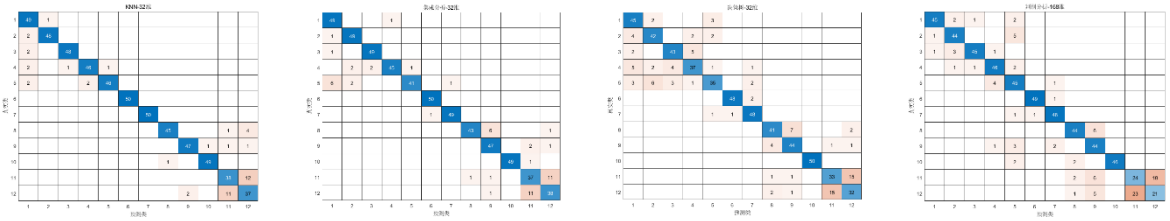
表4 最佳特征集合和平均准确率

最佳特征数量	平均准确率	特征集合
32	95.8%	dv_Y^β、dv_{Mag}^β、pk_Y^β、$\bar{\alpha}^X$ $\bar{\alpha}^Y$、σ_Z^α、$\alpha_{\max}^Z$、$\alpha_{\min}^Z$、R_{Mag}^α、R_{Mag}^β、$Skewness_X^\alpha$、$Skewness_Z^\alpha$、$Skewness_Y^\beta$、$Skewness_Z^\beta$、$Kurtosis_{Mag}^\alpha$、$Kurtosis_{Mag}^\beta$、$CrestFactor_Y^\beta$、$MarginFactor_X^\alpha$、$MarginFactor_Y^\alpha$、$WaveformFactor_Z^\beta$、Q_{1Y}^α、Q_{1Z}^β、Q_{2Y}^α、Q_{2Z}^α、Q_{2Mag}^α、Q_{2Z}^β、Q_{3Mag}^β、fcv_{Mag}^β、$SpectralKurtosis_{Mag}^\alpha$、$SpectralSkewness_{Mag}^\alpha$、$SpectralEntropy_Z^\alpha$、$SpectralEntropy_{Mag}^\alpha$

根据上述特征集合构建新的训练集，即 600×(32+1)的二维矩阵。为比较 32 维特征与 168 维特征之间的区别，我们比较两种维度下的验证混淆矩阵。



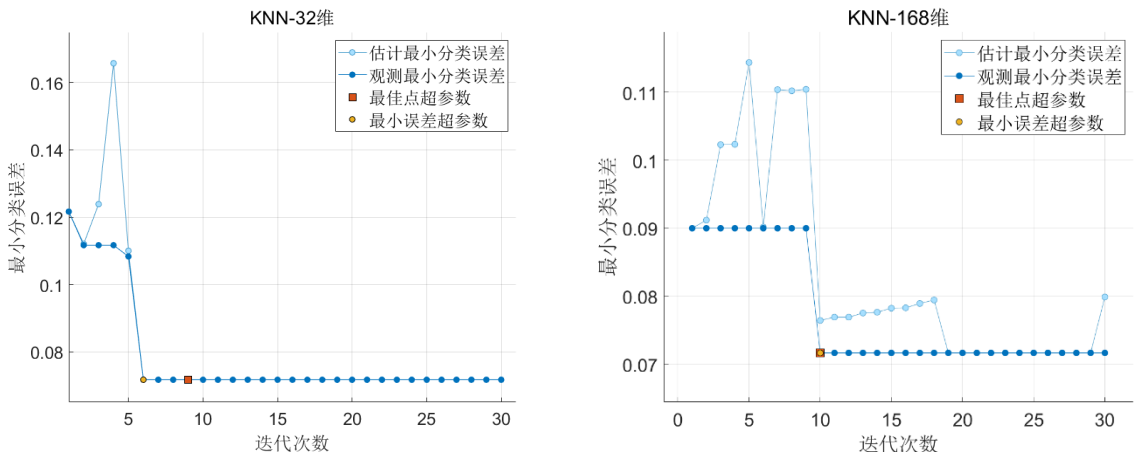
(a)KNN-168 维　(b)集成分析-168 维　(c)决策树-168 维　(d)判别分析-168 维



(e)KNN-32 维　(f)KNN-32 维　(g)决策树-32 维　(h)判别分析-32 维

图14 不同分类器特征筛选前后验证混淆矩阵对比

可以看出 32 维特征与 168 维特征的训练模型在 KNN 分类算法下效果接近，但是在集成分析、决策树和判别分析三种方法上则会出现比较大的偏差。下面具体比较 32 维特征与 168 维特征的 KNN 模型训练的最小分类误差图，迭代次数为 30 次。



(a)KNN-32 维　　(b)KNN-168 维

图15 KNN 模型训练 32 维与 168 维数据的最小分类误差图对比

从上述图中可以看出使用 32 维数据和 168 维数据进行训练的 KNN 模型效果。32 维数据训练的判别模型可以更快得到最小误差。

5.3.2 问题（1）：分类模型与判别模型结果对比

下面使用问题一中构建的分类模型对实验人员 Person4-Person13 进行分类，计算对每名人员活动状态分类的准确率，并将结果与问题二提出的判别模型进行比较。

表5 分类模型与判别模型结果对比

实验人员	分类模型准确率	判别模型准确率
Person4	96.7%	100%
Person5	80.0%	100%
Person6	90.0%	100%
Person7	86.7%	100%
Person8	88.3%	100%
Person9	93.3%	100%
Person10	88.3%	100%
Person11	88.3%	100%
Person12	90.0%	100%
Person13	93.3%	100%

通过对比可以发现利用 Person4-Person13 的带标签数据进行分类学习的判别模型准确率高达 100%，说明模型训练的有效性。同时问题一中的分类模型的结果准确率均在80%以上，甚至最高可达 96.7%，说明问题一所采取的方法的有效性。详细的分类结果详见附录。分类后的 Person12 和 Person13 之间的关联关系矩阵的热力图如下所示，可以明显看出动作 8 与动作 9、动作 11 和动作 12 的样本之间具有很高的重叠度，所以，聚类模型在这两个动作上可能存在一定的混淆。同时，不同人员在运动性动作方面还存在差异，如 Peron13 与 Person8、Person10，在运动动作上，区分度各不相同其他结果可见附件图表。

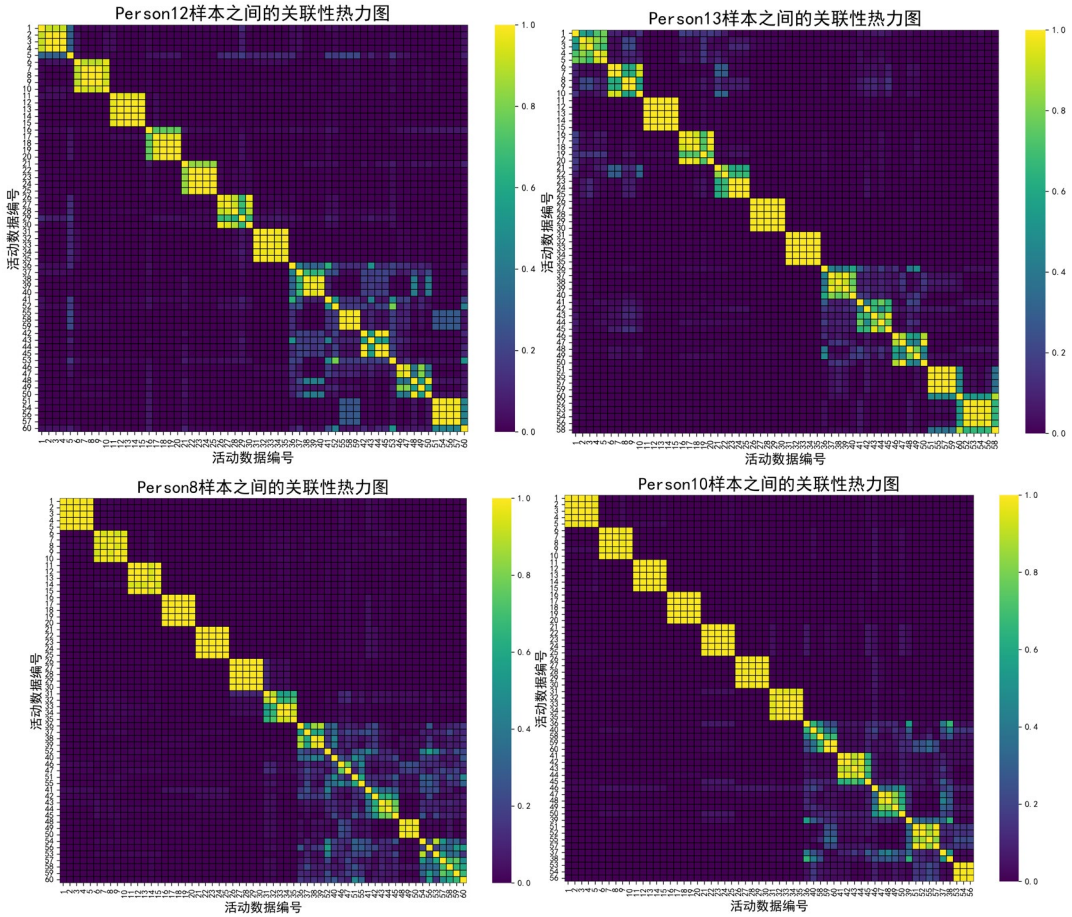


图16 聚类结果样本之间关联矩阵的热力图

5.3.3 问题（2）附件 3 中数据状态判别结果

采用判别模型对附件 3 中数据进行判别得到结果如下：

表6 附件 3 判别结果

活动类型	判别状态	活动类型	判别状态	活动类型	判别状态
SY1	5	SY11	12	SY21	6
SY2	1	SY12	7	SY22	2
SY3	7	SY13	3	SY23	8
SY4	11	SY14	3	SY24	5
SY5	7	SY15	3	SY25	2
SY6	10	SY16	1	SY26	9
SY7	2	SY17	4	SY27	8
SY8	6	SY18	2	SY28	2
SY9	7	SY19	8	SY29	6
SY10	10	SY20	12	SY30	5

各个状态数量及占比如下：

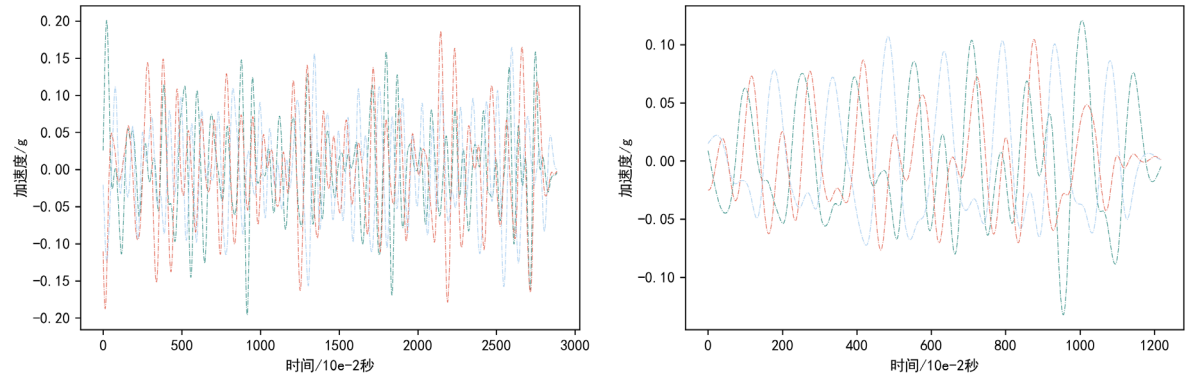
表7 附件 3 判别结果分布

活动状态	数量	百分比	活动状态	数量	百分比
1	2	6.67%	7	4	13.00%
2	5	16.67	8	3	10.00%
3	3	10.00%	9	1	3.33%
4	1	3.33%	10	2	6.67%
5	3	10.00%	11	1	3.33%
6	3	10.00%	12	2	6.67%

6 问题（三）分析与建模

6.1 问题分析

基于问题一与问题二的特征工程与分类器模型训练的结果，我们得到了同一人员在不同运动状态下的关键特征集合，并通过关键特征集合作为输入，利用 KNN 分类器实现了训练集上 12 类人员运动状态标签的高准确率，并验证了分类器模型与问题一的聚类器模型的一致性。然而，根据常识，不同实验人员在做出同一个运动状态时，往往因为身体条件以及个人习惯会造成运动速度，运动幅度的差异，以向前走为例，不同年龄，身高的人员，其运动速度、步幅以及重心的浮动数据均会产生较大差异，因此，通过运动数据进行人物信息的预测具有一定的合理性。如图 17 所示，人员 4 步行下楼时的垂直加速度变化相较于人员 10 幅度和频率较大。



（a）人员 4 步行下楼的加速度　　（b）人员 10 步行下楼的加速度

图17 不同人员在同一动作下的加速度数据波形差异

通过上述分析。初步确定在同一状态下的人员活动数据存在差异性。在此基础上，进一步说明人员年龄、身高、体重与活动数据的相关性。在给定三类人员信息相互独立的假设下，我们可以将**多任务预测问题**转换成单任务预测。区别于问题儿的活动状态预测，由于活动状态为离散标签，因此属于分类问题，而身高，年龄，体重等属于连续变量，因此属于**多标签回归问题**。在建立回归模型的基础上，题目要求建立人物画像模型，并给出附件 5 中的五名未知人员的 ID。

基于上述分析，将问题三初步确定为两项子任务：

1）人员类别 $ID_i(i=4,...,13)$ 与活动状态 $T_j(j=1,...,12)$ 两个分类型变量影响下的活动状态数据的差异性分析。其中在同一状态下的不同人员的差异性可以通过单因素方差分析继续，而人员类别与活动状态的相关性分析则可以通过双因素方差分析进行衡量。

2）基于活动状态数据的人物特征回归预测。通过输入活动状态数据，搭建回归模型进行训练，并利用训练模型对五名未知人员的 ID、年龄、身高、体重进行预测，给出五名人员的真实 ID。

6.2 双因素方差分析进行差异性检验

问题三额外给出 13 名实验人员的身高、体重和年龄数据，既要求验证不同人员的同一活动状态是否存在差异，又要求验证活动状态数据与实验人员的身高、体重和年龄是否具有相关性。由于每名实验人员都有独特的身高、体重、年龄组合，因此考虑使用实验人员 ID 号来代表这三个数据的组合。检验活动状态与实验人员身高、体重、年龄之间的差异性以及相关性，则可以转化为检验活动状态与实验人员 ID 的差异性和相关性。对于两个定类数据之间的差异性分析，我们采用双因素方差分析进行验证。

双因素方差分析：同时探讨两个独立的因素对实验结果变异的独立影像以及他们之间的交互作用。这里以活动状态作为因素 A，人员 ID 作为因素 B。响应变量为每名实验人员在每种活动状态下的 5 次采样数据。

由于在问题二中进行的关键特征提取是基于活动状态标签进行的，但是本问中人员数据的标签在活动状态基础上额外增添了人员的身高、体重和年龄数据，因此之前的 32 个关键特征集合并不适用于此问，所以我们仍采用问题一特征工程中的 168 维特征指标。将每组采样数据重构为一个 168 维的向量，从而得到 600×168 的二维矩阵，并进行标准化。

6.2.1 主成分分析压缩特征数量

但由于向量在代表实验结果时仍不便于计算，故采用主成分分析法进行特征压缩，通过正交变换将一组可能相关的变量转换为一组线性不相关的变量，并将特征数据投影到主成分上计算得分。主成分分析法通过从数据中提取出最重要的特征，达到简化数据复杂性的效果，同时尽可能的保留了原始数据中的信息。主要步骤为：1）数据标准化处理；2）计算协方差矩阵；3）求解特征值和特征向量；4）选择主成分并构造主成分得分。

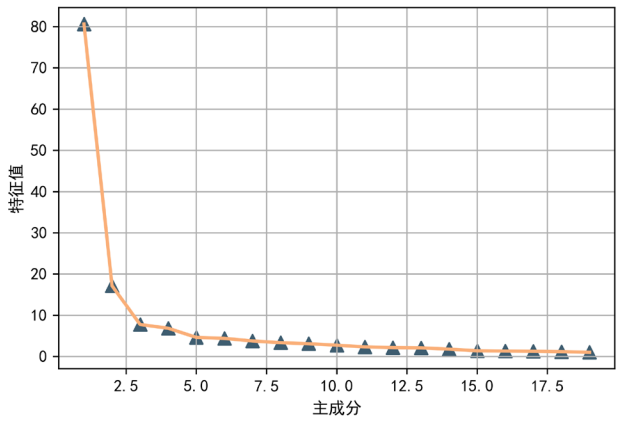


图18 主成分贡献率

从图中可以得出，仅第一主成分的方差贡献率就已经达到 80%，故可以使用主成分分析法对数据进行降维，并计算出主成分得分。

建立双因素方差分析模型如下：

表8 双因素方差分析模型

	Person4	…	Person13
向前走	(3.85,1.10,2.26,1.41,4.04)	…	(3.04,3.61,5.30,4.18,3.13)
向左走	(5.74,3.12,1.37,2.04,1.13)	…	(2.12,-0.01,1.13,1.45,0.62)
向右走	(4.27,2.28,2.37,2.34,3.04)	…	(0.85,0.22,0.23,0.05,0.86)
步行上楼	(1.67,0.99,0.85,7.48,3.23)	…	(2.90,1.19,2.02,4.27,2.14)
步行下楼	(2.00,1.42,1.07,5.12,4.14)	…	(2.60,2.60,4.84,7.08,5.60)
向前跑	(22.43,19.23,21.55,16.79,22.68)	…	(29.97,26.66,27.86,29.38,27.68)
跳跃	(13.93,8.13,8.86,8.83,6.87)	…	(21.30,21.49,20.78,20.32,23.64)
坐下	(-7.80,-7.51,-7.74,-7.91,-7.95)	…	(-8.15,-7.80,-7.48,-8.00,-8.02)
站立	(-7.75,-6.76,-5.14,-5.26,-6.23)	…	(-7.30,-7.23,-6.34,-6.53,-6.18)
躺下	(-7.22,-9.97,-8.38,-7.41,-9.63)	…	(-9.25,-7.94,-10.12,-10.26,-7.60)
乘坐电梯向上移动	(-7.78,-7.50,-7.72,-7.56,-7.79)	…	(-8.55,-9.11,-9.20,-9.45,-8.52)
乘坐电梯向下移动	(-7.39,-7.74,-7.74,-7.72,-8.03)	…	(-9.04,-8.54,-8.80,-8.59,-8.77)

其中表格列标题为运动状态，共包含 12 个水平，行标题为人员 ID，共包含 10 个水平。由于对数据进行了标准化处理，故计算主成分得分时存在负数，这表明该因素的水平与响应变量之间存在负相关关系，但并不影响实验结果。

表9 基于第一主成分的“运动状态-人员 ID”双因素方差分析表

OLS Regression Results

Dep. Variable:	第一主成分		R-squared:		0.240
Model:	OLS		Adj. R-squared:		0.236
Method:	Least Squares		F-statistic:		62.62
Date:	Fri, 12 Jul 2024		Prob (F-statistic):		3.31e-35
Time:	01:45:42		Log-Likelihood:		-2085.8
No. Observations:	600		AIC:		4180.
Df Residuals:	596		BIC:		4197.
Df Model:	3				
	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	8.5876	2.135	4.023	0.000	4.395	12.780
T	-1.3021	0.290	-4.489	0.000	-1.872	-0.732
ID	-0.0371	0.238	-0.156	0.876	-0.504	0.430
T:ID	0.0035	0.032	0.107	0.915	-0.060	0.067
Omnibus:	225.805		Durbin-Watson:			0.265
Prob(Omnibus):	0.000		Jarque-Bera (JB):			612.158
Skew:	1.935		Prob(JB):			1.18e-133
Kurtosis:	6.083		Cond. No.			451.

通过双因素方差分析结果可以看出，活动状态主效应显著，即同一名实验人员的不同活动状态差别显著。但是同一状态下，人员的主效应不显著，即同一活动状态下，不同人员的数据并不具有显著性差异。同时两者的交互效应也不显著。因此**通过主成分分析进行降维后的双因素方差分析结果说明不同人员的同一活动状态不存在差异**。分析原因有以下两点：

1、**主成分分析对数据维数压缩过大**，从 168 维的向量压缩为 1 个值，可能损失较多信息从而导致主效应和交互效应的不显著。

2、初始数据维数过多，**可能存在部分高度相关数据，干扰主效应以及交互效应的检验**。

6.2.2 逐个特征分析相关性

基于以上分析，选择将 168 维特征数据逐一作为响应进行方差分析，并统计各个运动状态下具有显著性差异的指标数量：

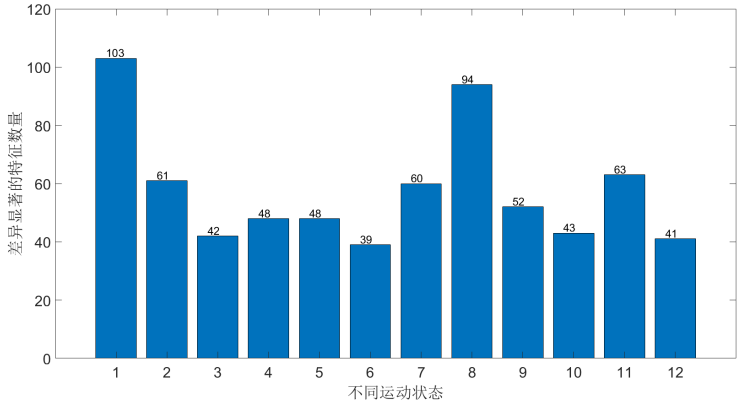


图19 168 维特征作为响应变量的活动状态主效应显著性数量图

上图可以看出将 168 维特征数据依次作为响应变量进行双因素方差分析，对于每个活动状态都可以找到一批特征体现人员主效应的显著性差异。这说明同一活动下，不同人员的活动存在差异。同时证明上述分析中主成分分析导致数据信息损失严重的推断。

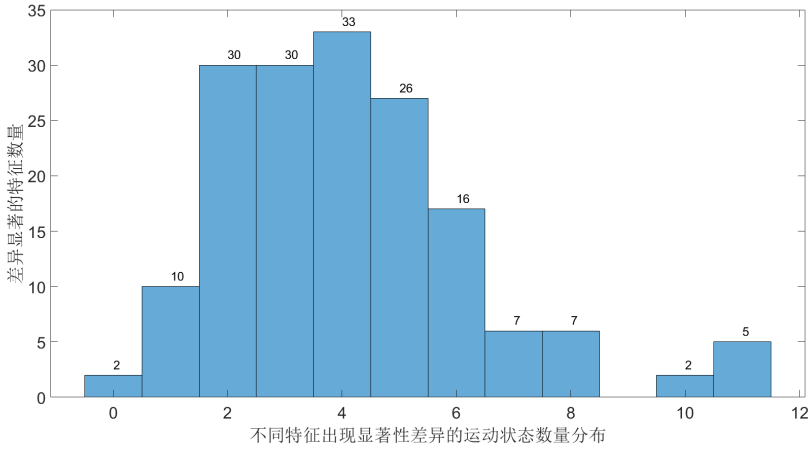


图20 168 维特征作为响应变量体现显著性差异的运动状态数量

上图表明 168 维数据在所有活动中体现显著性差异的数量分布，可以发现有 7 个特征指标在 10 个以上的运动状态中均体现显著性差异。也存在 2 个指标在所有运动状态下均不表现显著性差异，说明这 2 个指标可以考虑放弃。具体的指标信息详见附件。

6.3 基于 CNN 与集成学习的多任务回归预测

在指标的方差分析结果的基础上，我们认为问题一建立的 168 维特征集可以全部保留，以充分利用信号在特征上的差异性进行训练。我们通过搭建四通道的 CNN 卷积神经网络进行 ID 分类任务学习与（身高，体重，年龄）属性的回归任务学习。CNN 可以处理时间序列数据、非时序空间矩阵数据。通过建立多通道，多输出层的网络结构，可以在不同的输入数据集上进行多任务学习。问题的建模与求解框架如图 21 所示，对于初始题目中所给的在人员运动过程中采集的六维时间序列数据，可以构造为一个在一个维度上具有时间相关性，在另一个维度上具有空间相关性的**时间序列矩阵（图 2.a）**，将该矩阵作为输入，可以构建时序 CNN 对活动数据对应的人员 ID 标签进行分类任务学习与预测；进一步将时序矩阵压缩，构造 168×1 维的特征向量，并根据预测变量的特点，将不同的特征向量拼接，得到**特征矩阵(图 2.b)**，并基于特征矩阵的输入构建空间 CNN 对人员 ID 标签进行分类预测。在此基础上，通过集成学习算法，在已知样本集上进行训练，并对未知样本集进行预测，将两种分类预测模型所得结果进行对比，最终确定所属人员的 ID 编号。

为了实现基于人物身高，体重，年龄的人物画像功能，需要对三类连续型变量进行回归预测。通过采用多通道的 CNN，利用样本矩阵重构、分层聚类、模糊标签等处理方法，得到适用于不同标签预测的样本切片特征矩阵，将其为输入，训练得到对于身高、体重、年龄的回归预测值。

最后，通过 ID 与三类信息的匹配度、预测结果的稳定性等指标，对模型的能力进行评估，说明模型的合理性。

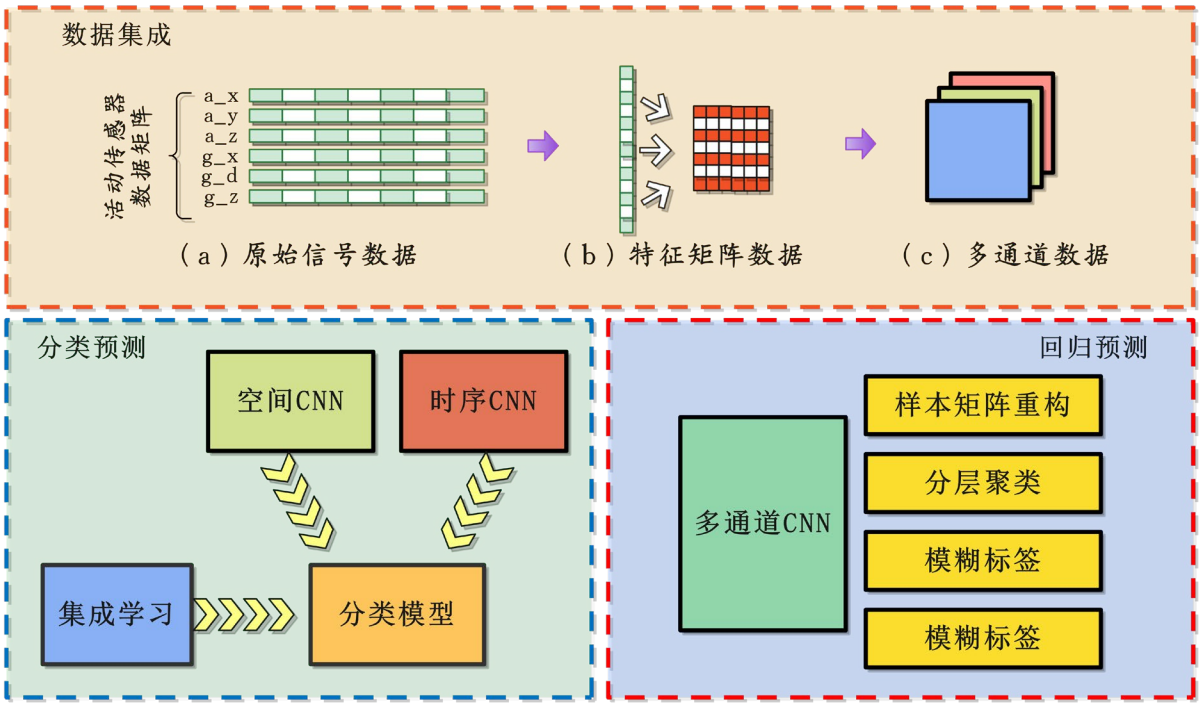
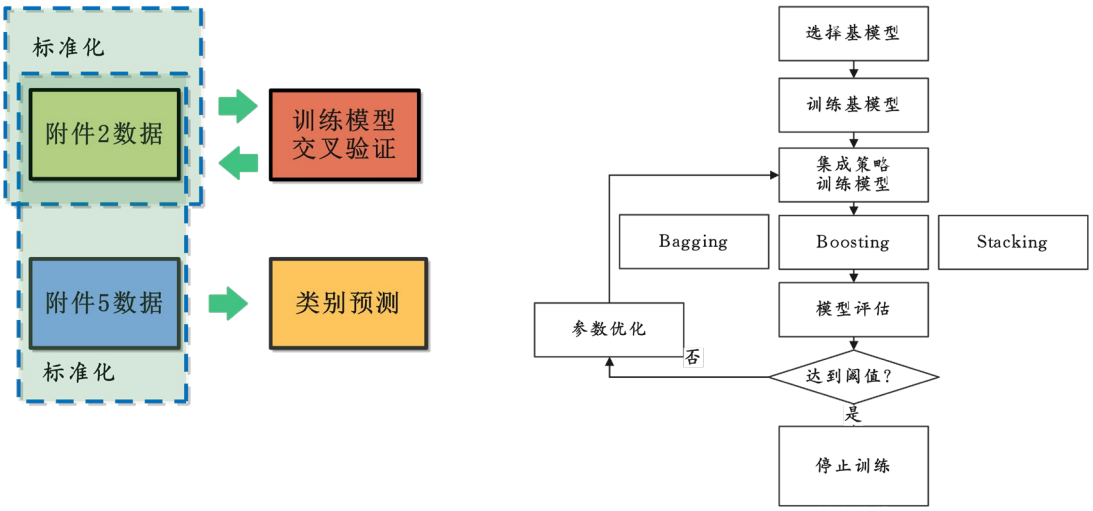


图21 基于多结构 CNN 的预测问题框架

6.3.1 基于集成学习的人员 ID 分类预测

1）数据处理与算法构建。集成学习将多个模型（通常称为“弱学习器”）结合起来，以提高整体的预测或分类性能，解决单一模型可能遇到的过拟合、欠拟合或准确率不足的问题，其核心为通过组合多个模型的预测来获得比单个模型更好的结果。在建立集成学习的输入数据时，首先对数据进行标准化，采用与第二问相同的正态分布标准化方法。

同时，附件 5 作为测试集，同样需要进行归一化，并采用与附件二总体相同的均值与方差作为标准化参数，如图 22 所示。集成优化的算法流程图如图 22 .(b)所示。



（a）数据归一化 **（b）集成优化算法流程图**

图22 集成学习数据处理与算法流程

2）模型训练与结果预测

在训练过程中，模型利用附件 2 中的 600 条特征数据，构建 600×168 维的输入数据，训练标签为活动来源人员的 ID，即一个活动特征向量对应一个标签。对模型进行训练，通过多轮的集成策略选择与 K 折验证，得到最终的训练模型，模型参数如表 10 所示。

表10 集成优化模型参数

参数	集成器			学习率		权重		观测量	代次		预测类
值	AdaBoostM2			0.8914		0.0017		600	11		10
参数	W1	W2	W3	W4	W5	W6	W7	W8	W9	W10	W11
值	0.47	0.43	0.38	0.41	0.37	0.39	0.34	0.38	0.43	0.36	0.38

利用该模型对附件 5 中的五名人员的每一个活动进行预测，共 60 条测试数据，其中每名未知人员均含有 12 条对应不同活动的状态数据，每条数据的活动类别、来源人员与预测 ID 如所示，通过多次训练与测试，得到结果稳定，因此该集成优化模型预测的结果是，**五位未知人员分别来自附件 2 中的人员 10，7，6，9，13**。

3）模型评估与结果分析

进一步，在测试结果稳定性的基础上，验证模型测试准确率的稳定性，进而验证其鲁棒性与泛化性。具体测试方法采用 K 折验证，通过随机选取 N 个维度的数据，遍历参数 $K(K=2,4,...,20)$，进行 K 折验证，得到不同维度与训练样本比例下的模型测试准确率，如表 11 所示，最终得到随着维度的升高，模型测试准确率可以稳定在 80%以上。

表11 改变维度与参数 K 条件下的模型测试准确率

平均准确率	10	15	20	…	155	160	165
K=2	0.7967	0.8750	0.9150	…	0.9700	0.9650	0.9683
K=4	0.8450	0.9083	0.9400	…	0.9850	0.9800	0.9850
K=6	0.9200	0.9067	0.9517				
…	…	…		…		…	…
K=16	0.7833	0.9133	0.9483		0.9900	0.9900	0.9883
K=18	0.8800	0.9583	0.9650	…	0.9867	0.9867	0.9883
K=20	0.8817	0.9467	0.9567	…	0.9900	0.9867	0.9883

6.3.2 基于时序矩阵与静态特征矩阵的 CNN 分类预测

在集成学习预测人员 ID 的基础上，通过构建静态特征矩阵与时序数据作为 CNN 的输入，实现单通道的标签分类预测。

1）数据输入与网络结构。当选择静态特征矩阵与时序数据作为输入时，考虑到时序数据矩阵对应一个预测标签，因此训练样本对应的标签数据为一维向量，因此时序 CNN 的输入矩阵尺寸为（6，样本点数量），由于不同传感器样本的长度不同，且差异较大，样本数量从 1300 到 5000 不等，同时考虑到统一运动状态下便于比较不同人员的活动数据差异，因此将同一状态下同一人员的 5 条运动状态拼接，并以长度为 1000 进行裁剪，即可得到标签为该人员 ID 的若干时间序列矩阵，因此时序 CNN 的输入矩阵尺度为（6，1000）。同样的，考虑到测试数据中每个人对应于 12 个已知标签的活动状态，因此在构建训练样本时参考该数据规模，将静态特征矩阵的输入尺寸设置为（12，48），其中矩阵的每一行代表一个活动状态数据集，同时也对应一个标签。因此训练样本的标签数据尺寸为（样本量，4），4 代表网络的通道数，也即待预测的标签数量，分别为 ID，年龄，身高，体重。

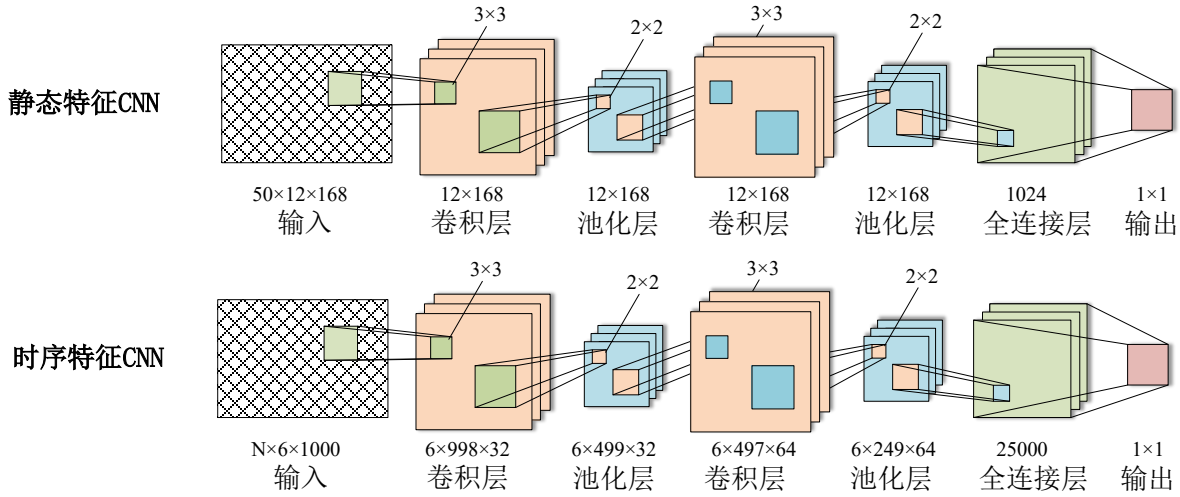


图23 时序 CNN 与静态 CNN 结构示意图

2）模型训练与预测结果。在训练上述模型时，通过构建三维矩阵，实现输入数据的批次化，设定批数 batch_size=5，模型迭代数为 epoch=10。通过 100 次预测并进行统计，通过选取最相近标签的原则，确定 CNN 对于附件 5 中**五名未知编号人员的预测结果为 10，7，6，9，5**。

6.3.3 基于四通道特征矩阵的 CNN 回归预测

在单通道静态 CNN 模型的基础上，通过设计四个输入数据通道，将相同划分结构的训练数据（50×12×168×1）输入到具有共享卷积层以及一个分类输出层（用于输出分类预测的 ID）和三个回归输出层（用于预测身高、年龄、体重）的四通道 CNN 中，其中 CNN 的网络结构如所示，参数如所示，

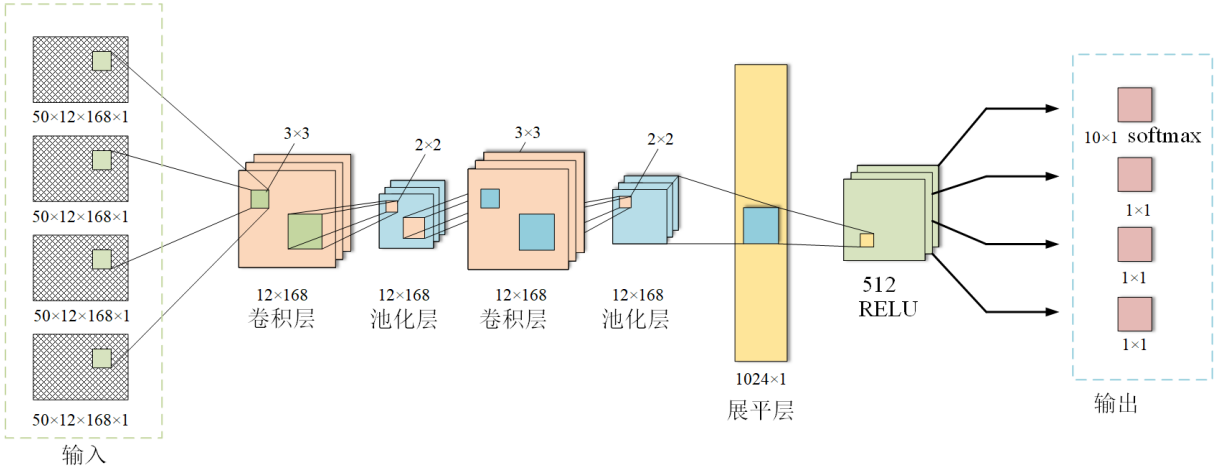


图24 四通道 CNN 结构示意图

利用上述结构网路进行训练，并在每次训练后对附件 5 的 5 个特征矩阵进行多任务预测。通过设置 100 次训练预测，并将预测结果均值化输出，得到模型对于 5 个未知人员的 ID、年龄、身高、体重预测结果，如所示

人员代号	ID	年龄	身高	体重
Unknown1	10.2928	47.3018	173.2534	67.7791
Unknown2	7.1920	37.5540	185.1168	61.7355
Unknown3	6.3460	26.9672	168.3410	53.4088
Unknown4	9.1474	22.5749	174.1392	61.6477
Unknown5	5.1681	23.2317	179.0506	58.3730

6.3.4 附件 5 人物特征与 ID 预测结果

基于以上模型建立与结果预测，最终得到附件 5 中五名未知人员的 ID，身高，年龄，体重预测结果如表 6.3.4 表 12 所示。

表12 附件 5 的预测结果

人员代号	ID	年龄	身高	体重
Unknown1	10	47	173	67
Unknown2	7	37	185	62
Unknown3	6	27	168	53
Unknown4	9	23	174	62
Unknown5	13	23	179	58

对应的正确 ID 与其属性如下表所示，

表13 真实 ID 对应属性

人员代号	ID	年龄	身高	体重
Unknown1	10	49	166	68
Unknown2	7	35	170	63
Unknown3	6	27	164	50
Unknown4	9	22	180	76
Unknown5	13	36	170	80

通过对比，发现除身高外，其余属性差异较小。

7 模型评价与推广

7.1 模型的优点

(1) 模型一通过特征工程，综合考虑数据的统计特征、时序特征、频域特征、时-域特征，对每个维度的数据构建了 21 个特征指标，能够很好的将时序信号数据转化为一个向量，为模型训练提供方便；
(2) 问题一采用的基于简单对比学习框架的集成聚类模型能够很好的发掘高维数据样本之间的差异，在无标签聚类问题上显著优于传统聚类算法；
(3) 模型二采用多次 K 折交叉验证并测试不同的 K 值，充分利用已有数据进行训练，同时通过比较不同 K 值训练下判别模型的准确度，得出最佳的 K 值；
(4) 模型二通过遗传算法筛选关键特征集合，同时比较不同分类器的性能，在保证判别准确性的同时，提高了模型泛化性；
(5) 问题三中，通过分类器的分类预测与 CNN 分类预测相互验证，并作为 CNN 回归预测的验证对象，有利于提高模型预测的准确性与可信性。

7.2 模型的不足

(1) 对于关键特征的提取，遗传算法并不一定能找到每个数量限制下的最佳特征集合；
(2) SimCLR 模型是一个黑盒，样本特征数据转换后的实际意义未知，可解释性较差；
(3) 在问题三中，没有充分考虑身高，体重，年龄等特征之间的相关性，将回归预测人物简化为单任务模式，忽略了多任务模式下的预测效果评测。
(4) 问题三对于身高数据的预测效果较差，且基于特征矩阵的训练模型容易忽略时序信息的特征。

7.3 模型的推广

对于问题三中的 CNN 网络，通过加入长期记忆因子，使得网络进化为 RNN，能够更好地适应题目所给的时序数据，相对于静态结构网络，能够提取出其中的时序数据。

参考文献

[1] 徐冉, 许有熊, 郑煊宇, 等. 基于加速度传感器的人体运动状态识别监测研究[J/OL]. 电脑与信息技术, 2022, 30(2): 46-48.

[2] 李丹, 陈焱焱, 姚志明, 等. 基于三轴加速度传感器的人体日常体力活动识别系统设计[J/OL]. 仪表技术, 2013(9): 1-5.

[3] 李锋, 潘敬奎. 基于三轴加速度传感器的人体运动识别[J]. 计算机研究与发展, 2016, 53(3): 621-631.

[4] 张乾勇. 基于多传感器的人体运动模式识别方法研究[D/OL]. 西南交通大学, 2018.

[5] 张文婷. 基于智能手机传感器的人体运动状态识别研究[D/OL]. 吉林大学, 2020.

[6] 殷晓玲, 陈晓江, 夏启寿, 等. 基于智能手机内置传感器的人体运动状态识别[J]. 通信学报, 2019, 40(3): 157-169.

[7] JAIN A, KANHANGAD V. Human Activity Classification in Smartphones Using Accelerometer and Gyroscope Sensors[J/OL]. IEEE Sensors Journal, 2018, 18(3): 1169-1177.

附　录

附录 A: 问题一分类模型对附件 2 数据分类具体结果

表14 分类模型对 Person4 判别结果

数据	类别	数据	类别	数据	类别	数据	类别
a1t1	1	a4t1	4	a7t1	7	a10t1	10
a1t2	1	a4t2	4	a7t2	7	a10t2	10
a1t3	1	a4t3	4	a7t3	7	a10t3	10
a1t4	1	a4t4	4	a7t4	7	a10t4	10
a1t5	1	a4t5	4	a7t5	7	a10t5	10
a2t1	2	a5t1	5	a8t1	8	a11t1	12
a2t2	2	a5t2	5	a8t2	8	a11t2	11
a2t3	2	a5t3	5	a8t3	8	a11t3	11
a2t4	2	a5t4	5	a8t4	8	a11t4	11
a2t5	2	a5t5	5	a8t5	8	a11t5	11
a3t1	3	a6t1	6	a9t1	9	a12t1	12
a3t2	3	a6t2	6	a9t2	9	a12t2	11
a3t3	3	a6t3	6	a9t3	9	a12t3	12
a3t4	3	a6t4	6	a9t4	9	a12t4	12
a3t5	3	a6t5	6	a9t5	9	a12t5	12

表15 分类模型对 Person5 判别结果

数据	类别	数据	类别	数据	类别	数据	类别
a1t1	1	a4t1	5	a7t1	11	a10t1	10
a1t2	1	a4t2	5	a7t2	7	a10t2	10
a1t3	1	a4t3	4	a7t3	7	a10t3	10
a1t4	1	a4t4	4	a7t4	7	a10t4	10
a1t5	1	a4t5	4	a7t5	7	a10t5	10
a2t1	2	a5t1	11	a8t1	8	a11t1	11
a2t2	2	a5t2	11	a8t2	8	a11t2	4
a2t3	2	a5t3	5	a8t3	8	a11t3	6
a2t4	2	a5t4	5	a8t4	8	a11t4	12
a2t5	2	a5t5	5	a8t5	8	a11t5	12
a3t1	3	a6t1	11	a9t1	9	a12t1	10
a3t2	3	a6t2	6	a9t2	7	a12t2	12
a3t3	3	a6t3	6	a9t3	9	a12t3	12
a3t4	3	a6t4	6	a9t4	9	a12t4	12
a3t5	3	a6t5	6	a9t5	9	a12t5	4

表16 分类模型对 Person6 判别结果

数据	类别	数据	类别	数据	类别	数据	类别
a1t1	4	a4t1	1	a7t1	7	a10t1	10
a1t2	1	a4t2	4	a7t2	7	a10t2	10
a1t3	1	a4t3	4	a7t3	7	a10t3	10
a1t4	1	a4t4	4	a7t4	7	a10t4	10
a1t5	1	a4t5	4	a7t5	7	a10t5	10
a2t1	2	a5t1	5	a8t1	8	a11t1	11
a2t2	2	a5t2	5	a8t2	8	a11t2	11
a2t3	2	a5t3	5	a8t3	8	a11t3	11
a2t4	2	a5t4	5	a8t4	8	a11t4	12
a2t5	2	a5t5	5	a8t5	8	a11t5	12
a3t1	3	a6t1	6	a9t1	9	a12t1	12
a3t2	3	a6t2	6	a9t2	9	a12t2	12
a3t3	3	a6t3	6	a9t3	9	a12t3	11
a3t4	3	a6t4	6	a9t4	9	a12t4	11
a3t5	3	a6t5	6	a9t5	9	a12t5	12

表17 分类模型对 Person7 判别结果

数据	类别	数据	类别	数据	类别	数据	类别
a1t1	3	a4t1	4	a7t1	7	a10t1	12
a1t2	1	a4t2	4	a7t2	7	a10t2	10
a1t3	1	a4t3	4	a7t3	7	a10t3	10
a1t4	1	a4t4	4	a7t4	7	a10t4	10
a1t5	1	a4t5	4	a7t5	7	a10t5	10
a2t1	2	a5t1	5	a8t1	8	a11t1	11
a2t2	2	a5t2	5	a8t2	8	a11t2	12
a2t3	2	a5t3	5	a8t3	8	a11t3	9
a2t4	2	a5t4	5	a8t4	8	a11t4	11
a2t5	2	a5t5	5	a8t5	8	a11t5	11
a3t1	1	a6t1	6	a9t1	10	a12t1	11
a3t2	3	a6t2	6	a9t2	9	a12t2	12
a3t3	3	a6t3	6	a9t3	9	a12t3	12
a3t4	3	a6t4	6	a9t4	9	a12t4	12
a3t5	3	a6t5	6	a9t5	9	a12t5	11

表18 分类模型对 Person8 判别结果

数据	类别	数据	类别	数据	类别	数据	类别
a1t1	1	a4t1	4	a7t1	7	a10t1	11
a1t2	1	a4t2	4	a7t2	7	a10t2	11
a1t3	1	a4t3	4	a7t3	7	a10t3	10
a1t4	1	a4t4	4	a7t4	7	a10t4	10
a1t5	1	a4t5	4	a7t5	7	a10t5	10
a2t1	2	a5t1	5	a8t1	8	a11t1	11
a2t2	2	a5t2	5	a8t2	8	a11t2	8
a2t3	2	a5t3	5	a8t3	8	a11t3	12
a2t4	2	a5t4	5	a8t4	8	a11t4	10
a2t5	2	a5t5	5	a8t5	11	a11t5	11
a3t1	3	a6t1	6	a9t1	9	a12t1	10
a3t2	3	a6t2	6	a9t2	9	a12t2	12
a3t3	3	a6t3	6	a9t3	9	a12t3	12
a3t4	3	a6t4	6	a9t4	9	a12t4	12
a3t5	3	a6t5	6	a9t5	9	a12t5	12

表19 分类模型对 Person9 判别结果

数据	类别	数据	类别	数据	类别	数据	类别
a1t1	1	a4t1	4	a7t1	7	a10t1	11
a1t2	1	a4t2	4	a7t2	7	a10t2	11
a1t3	1	a4t3	4	a7t3	7	a10t3	10
a1t4	1	a4t4	4	a7t4	7	a10t4	10
a1t5	1	a4t5	4	a7t5	7	a10t5	10
a2t1	2	a5t1	5	a8t1	8	a11t1	11
a2t2	2	a5t2	5	a8t2	8	a11t2	8
a2t3	2	a5t3	5	a8t3	8	a11t3	12
a2t4	2	a5t4	5	a8t4	8	a11t4	10
a2t5	2	a5t5	5	a8t5	11	a11t5	11
a3t1	3	a6t1	6	a9t1	9	a12t1	10
a3t2	3	a6t2	6	a9t2	9	a12t2	12
a3t3	3	a6t3	6	a9t3	9	a12t3	12
a3t4	3	a6t4	6	a9t4	9	a12t4	12
a3t5	3	a6t5	6	a9t5	9	a12t5	12

表20 分类模型对 Person10 判别结果

数据	类别	数据	类别	数据	类别	数据	类别
a1t1	1	a4t1	4	a7t1	7	a10t1	10
a1t2	1	a4t2	4	a7t2	7	a10t2	10
a1t3	1	a4t3	4	a7t3	7	a10t3	10
a1t4	1	a4t4	4	a7t4	7	a10t4	10
a1t5	1	a4t5	4	a7t5	7	a10t5	10
a2t1	2	a5t1	5	a8t1	12	a11t1	11
a2t2	2	a5t2	5	a8t2	8	a11t2	11
a2t3	2	a5t3	5	a8t3	8	a11t3	8
a2t4	2	a5t4	5	a8t4	11	a11t4	8
a2t5	2	a5t5	5	a8t5	12	a11t5	11
a3t1	3	a6t1	6	a9t1	9	a12t1	8
a3t2	3	a6t2	6	a9t2	9	a12t2	11
a3t3	3	a6t3	6	a9t3	9	a12t3	12
a3t4	3	a6t4	6	a9t4	9	a12t4	12
a3t5	3	a6t5	6	a9t5	9	a12t5	12

表21 分类模型对 Person11 判别结果

数据	类别	数据	类别	数据	类别	数据	类别
a1t1	1	a4t1	4	a7t1	7	a10t1	10
a1t2	1	a4t2	4	a7t2	7	a10t2	10
a1t3	1	a4t3	4	a7t3	7	a10t3	10
a1t4	1	a4t4	4	a7t4	7	a10t4	10
a1t5	1	a4t5	4	a7t5	7	a10t5	10
a2t1	2	a5t1	5	a8t1	8	a11t1	9
a2t2	2	a5t2	5	a8t2	8	a11t2	12
a2t3	2	a5t3	5	a8t3	8	a11t3	11
a2t4	2	a5t4	5	a8t4	8	a11t4	11
a2t5	2	a5t5	5	a8t5	8	a11t5	11
a3t1	3	a6t1	6	a9t1	12	a12t1	11
a3t2	3	a6t2	6	a9t2	12	a12t2	12
a3t3	3	a6t3	6	a9t3	9	a12t3	9
a3t4	3	a6t4	6	a9t4	9	a12t4	12
a3t5	3	a6t5	6	a9t5	9	a12t5	11

表22 分类模型对 Person12 判别结果

数据	类别	数据	类别	数据	类别	数据	类别
a1t1	1	a4t1	4	a7t1	7	a10t1	10
a1t2	1	a4t2	4	a7t2	7	a10t2	10
a1t3	1	a4t3	4	a7t3	7	a10t3	10
a1t4	1	a4t4	4	a7t4	7	a10t4	10
a1t5	1	a4t5	4	a7t5	7	a10t5	10
a2t1	2	a5t1	5	a8t1	8	a11t1	11
a2t2	2	a5t2	5	a8t2	8	a11t2	11
a2t3	2	a5t3	5	a8t3	8	a11t3	11
a2t4	2	a5t4	5	a8t4	8	a11t4	11
a2t5	2	a5t5	5	a8t5	8	a11t5	11
a3t1	3	a6t1	6	a9t1	9	a12t1	12
a3t2	3	a6t2	6	a9t2	9	a12t2	12
a3t3	3	a6t3	6	a9t3	9	a12t3	12
a3t4	3	a6t4	6	a9t4	9	a12t4	12
a3t5	3	a6t5	6	a9t5	9	a12t5	12

表23 分类模型对 Person13 判别结果

数据	类别	数据	类别	数据	类别	数据	类别
a1t1	1	a4t1	4	a7t1	7	a10t1	10
a1t2	1	a4t2	4	a7t2	7	a10t2	10
a1t3	1	a4t3	4	a7t3	7	a10t3	10
a1t4	1	a4t4	4	a7t4	7	a10t4	10
a1t5	1	a4t5	4	a7t5	7	a10t5	10
a2t1	2	a5t1	5	a8t1	8	a11t1	12
a2t2	2	a5t2	5	a8t2	8	a11t2	11
a2t3	2	a5t3	5	a8t3	8	a11t3	11
a2t4	2	a5t4	5	a8t4	8	a11t4	11
a2t5	2	a5t5	5	a8t5	8	a11t5	12
a3t1	3	a6t1	6	a9t1	9	a12t1	11
a3t2	3	a6t2	6	a9t2	9	a12t2	12
a3t3	3	a6t3	6	a9t3	9	a12t3	11
a3t4	3	a6t4	6	a9t4	9	a12t4	12
a3t5	3	a6t5	6	a9t5	9	a12t5	12

附录 B：支撑材料列表

1、支撑材料文件夹数据目录中的所有.xlsx 文件均为将初始数据扩充到 8 维后提取的 168 维特征的数据，并已经标准化；

2、“问题一”中包含了用于问题一聚类分析的代码与运行所需数据；

3、“问题二”中包含不同分类器测试不同维度的代码、结果、论文中出现的图片，以及分类模型对附件二数据的聚类结果；

4、“问题三”中包含了依次将 168 维作为响应变量的双因素分析的结果统计，T_score 对应图 15 168 维特征作为响应变量的活动状态主效应显著性数量图，score_col_names 对应图 16 168 维特征作为响应变量体现显著性差异的运动状态数量。

5、“数据”文件夹

支撑材料列表

序号	文件名	材料说明
01	数据/Person1.xlsx	问题 1：对附件 1-Person1 的运动数据提取特征
02	数据/Person2.xlsx	问题 1：对附件 1-Person2 的运动数据提取特征
03	数据/Person3.xlsx	问题 1：对附件 1-Person3 的运动数据提取特征
04	数据/附件 2.xlsx	问题 2：对附件 2 的运动数据提取特征
05	数据/附件 3.xlsx	问题 2：对附件 3 的运动数据提取特征
06	数据/Score_col_name.xlsx	问题 3：168 维指标逐一做方差分析的显著性状态列表
07	数据/T_score.xlsx	问题 3：168 维指标逐一做方差分析的显著性状态列表
08	数据/集成学习测试准确率.xlsx	问题 3：保存了不同维度不同 K 折下的模型测试准确率
09	数据/人员信息与活动状态的关联数据.xlsx	问题 3：用于 CNN 训练的多层标签输入数据矩阵
10	问题一（文件夹）	问题 1 代码、数据结果、聚类结果
11	问题二（文件夹）	问题 2 代码、框图、数据结果、分类结果
12	问题三（文件夹）	问题 3 代码、框图、数据结果

附录 C：主要代码环境

代码环境	操作系统: Windows 10 家庭中文版(Version 19044.2486) 编程语言: Python 3.7.1 (Anaconda Navigator 1.9.2)、Matlab2021b 编辑器: PyCharm 2018.3.2 (Professional Edition) 代码详见: Code/Combine_Pyecharts_with_igraph.py

附录 D：问题一： SimCLRFeatureMatrix：python 代码

```
# -- coding:utf-8 --
from collections import Counter

import pandas as pd
import torch
import torch.nn.functional as F
from matplotlib import pyplot as plt
from sklearn.preprocessing import MinMaxScaler
from torch.utils.data import DataLoader, Dataset
from sklearn.cluster import KMeans
import numpy as np


# 创建数据集类
class SimpleDataset(Dataset):
    def __init__(self, data, transform=None):
        self.data = data
        self.transform = transform

    def __len__(self):
        return len(self.data)

    def __getitem__(self, idx):
        sample = self.data[idx]
        if self.transform:
            sample = self.transform(sample)
        return sample


# 数据增强（对于非图像数据，这里简单采用数据复制）
class SimCLRTransform:
```

```
    def __init__(self):
        pass

    def __call__(self, x):
        return x, x  # 返回两个相同的数据来模拟增强后的两视图
```

```
class SimpleEncoder(torch.nn.Module):
    def __init__(self, input_dim=96, output_dim=128):
        super(SimpleEncoder, self).__init__()
        self.layer1 = torch.nn.Linear(input_dim, 512)
        self.layer2 = torch.nn.Linear(512, output_dim)

    def forward(self, x):
        x = F.relu(self.layer1(x))
        x = self.layer2(x)
        return F.normalize(x, dim=-1)  # L2 归一化


# 自监督对比学习训练
class SimCLR:
    def __init__(self, encoder, tau=0.5):
        self.encoder = encoder
        self.tau = tau
        self.criterion = torch.nn.CrossEntropyLoss()

    def info_nce_loss(self, features):
        labels = torch.cat([torch.arange(features.size(0) // 2) for _ in
range(2)], dim=0)
        labels = (labels.unsqueeze(0) == labels.unsqueeze(1)).float()
        features = F.normalize(features, dim=1)

        similarity_matrix = torch.matmul(features, features.T)
        mask = torch.eye(labels.shape[0], dtype=torch.bool)
        labels = labels[~mask].view(labels.shape[0], -1)
        similarity_matrix = similarity_matrix[~mask].view(similarity_ma-
trix.shape[0], -1)

        positives = similarity_matrix[labels.bool()].view(labels.shape[0], -1)
        negatives = similarity_matrix[~labels.bool()].view(similarity_ma-
trix.shape[0], -1)

        logits = torch.cat([positives, negatives], dim=1)
        labels = torch.zeros(logits.shape[0], dtype=torch.long)

        logits = logits / self.tau
        return self.criterion(logits, labels)

    def train(self, dataloader, optimizer, epochs=500):
        lossData = []
        self.encoder.train()
        for epoch in range(epochs):
            for (x1, x2) in dataloader:
                optimizer.zero_grad()
```

```
                x1, x2 = x1.float(), x2.float()  # 保证输入是浮点数类型
                h1 = self.encoder(x1)
                h2 = self.encoder(x2)
                loss = self.info_nce_loss(torch.cat([h1, h2], dim=0))
                loss.backward()
                optimizer.step()
            lossData.append(loss.item())
            print(f'Epoch [{epoch + 1}/{epochs}], Loss: {loss.item():.4f}')
        lossData = pd.DataFrame(lossData)
        lossData.to_excel("损失函数.xlsx")

def updateRelationship(relationship, set, counter):
    for i in range(12):
        if counter[i] == 5:
            reward = 1
        elif counter[i] > 5:
            reward = 1 / counter[i]
        else:
            reward = 1 / (10 - counter[i])
    for subset in set:
        for j in range(len(subset) -1):
            for k in range(j+1, len(subset)):
                relationship[subset[j]][subset[k]] += reward
                relationship[subset[k]][subset[j]] += reward


indexx = []
for i in range(1,13):
    for j in range(5):
        indexx.append(i)


for name0 in range(1, 2):
    personIndex = "Person" + str(name0)
    personPath = "E:/2024 年湖南省研究生数学建模竞赛赛题及附件/附件 1/" + personIn-
dex + "/feature.xlsx"
    # 读取数据
    data = pd.read_excel(personPath)
    # 删除最后一列 如果 data 的列有 97 个特征，则删除最后一列
    if data.shape[1] == 97:
        data = data.iloc[:, :-1]
    # 初始化 MinMaxScaler
    scaler = MinMaxScaler()
    # 对 DataFrame 中的数据进行缩放（归一化）
    data = scaler.fit_transform(data)
    # 建立一个 60*60 的矩阵
    relationship = np.zeros((60, 60))
    for i in range(1):
        print("*******第" + str(i+1) + "次分类*******")
```

```
        tempLabel = [[] for m in range(12)]
        transform = SimCLRTransform()
        dataset = SimpleDataset(data, transform=transform)
        dataloader = DataLoader(dataset, batch_size=10, shuffle=True)
        # 初始化模型并训练
        encoder = SimpleEncoder(input_dim=96, output_dim=128)
        simclr = SimCLR(encoder)
        optimizer = torch.optim.Adam(encoder.parameters(), lr=0.001)
        simclr.train(dataloader, optimizer, epochs=500)
        # 提取特征并进行聚类
        feature_vectors = []
        for x in DataLoader(SimpleDataset(data), batch_size=1):
            x = x.float()  # 保证输入是浮点数类型
            with torch.no_grad():
                feature = encoder(x)
            feature_vectors.append(feature.squeeze(0).numpy())
        feature_vectors = np.array(feature_vectors)
        df = pd.DataFrame(feature_vectors)
        df["index"] = indexx
        # df.to_excel("E:/2024年湖南省研究生数学建模竞赛赛题及附件/" + personIn-
dex + "NewFeature.xlsx")
        kmeans = KMeans(n_clusters=12, random_state=0).fit(feature_vectors)
        labels = kmeans.labels_
        for j in range(60):
            la = labels[j]
            tempLabel[la].append(j)
        counter = Counter(labels)
        updateRelationship(relationship, tempLabel, counter)
    df = pd.DataFrame(relationship)

    # df.to_excel("E:/2024年湖南省研究生数学建模竞赛赛题及附件/" + personIndex +
"Relationship0.xlsx")
    print(relationship)
```

附录 E：问题二：KNN 分类器 python 代码

```
    KNN分类器
function [trainedClassifier, validationAccuracy] = KNN(trainingData,length_te-
zheng)
%  输入:
%      trainingData: 一个与导入 App 中的矩阵具有相同列数和数据类型的矩阵。
%
%  输出:
%      trainedClassifier: 一个包含训练的分类器的结构体。该结构体中具有各种关于所训练分
%       类器的信息的字段。
%
```

```
%      trainedClassifier.predictFcn: 一个对新数据进行预测的函数。
%
%      validationAccuracy: 包含准确度百分比的双精度值。在 App 中，"模型" 窗格显示每
%       个模型的总体准确度分数。
% 提取预测变量和响应
% 以下代码将数据处理为合适的形状以训练模型。
%
% 将输入转换为表
af=[];
for i=1:length_tezheng
    as{i}=['column_',num2str(i)];
    ap{i}=['column_',num2str(i)];
    af=[af,'false'];
end

as{length_tezheng+1}=['column_',num2str(length_tezheng+1)];
inputTable = array2table(trainingData, 'VariableNames', as);
% aa=['column_',num2str(21)]
predictorNames=ap;
predictors = inputTable(:, predictorNames);
eval(['response = inputTable.column_',num2str(length_tezheng+1),';']);
isCategoricalPredictor=af;

% 训练分类器
% 以下代码指定所有分类器选项并训练分类器。
classificationKNN = fitcknn(...
    predictors, ...
    response, ...
    'Distance', 'Correlation', ...
    'Exponent', [], ...
    'NumNeighbors', 1, ...
    'DistanceWeight', 'SquaredInverse', ...
    'Standardize', true, ...
    'ClassNames', [1; 2; 3; 4; 5; 6; 7; 8; 9; 10; 11; 12]);
%     'ClassNames', [1; 2; 3; 4; 5; 6; 7; 8; 9; 10; 11; 12; 13]);

% 使用预测函数创建结果结构体
predictorExtractionFcn = @(x) array2table(x, 'VariableNames', predictorNames);
knnPredictFcn = @(x) predict(classificationKNN, x);
trainedClassifier.predictFcn = @(x) knnPredictFcn(predictorExtractionFcn(x));

% 向结果结构体中添加字段
trainedClassifier.ClassificationKNN = classificationKNN;
trainedClassifier.About = '此结构体是从分类学习器 R2021b 导出的训练模型。';
trainedClassifier.HowToPredict = sprintf('要对新预测变量列矩阵 X 进行预测，请使用: \n
yfit = c.predictFcn(X) \n 将 ''c'' 替换为作为此结构体的变量的名称，例如 ''trainedMod-
el''。\n \nX 必须包含正好 20 个列，因为此模型是使用 20 个预测变量进行训练的。\nX 必须仅
包含与训练数据具有完全相同的顺序和格式的\n 预测变量列。不要包含响应列或未导入 App 的任何
列。\n \n 有关详细信息，请参阅 <a href="matlab:helpview(fullfile(docroot, ''stats'',
''stats.map''), ''appclassification_exportmodeltoworkspace'')">How to predict us-
ing an exported model</a>。');

% 提取预测变量和响应
% 以下代码将数据处理为合适的形状以训练模型。
%
% 将输入转换为表
inputTable = array2table(trainingData, 'VariableNames', as);
```

```
predictors = inputTable(:, predictorNames);
eval(['response = inputTable.column_',num2str(length_tezheng+1),';']);
isCategoricalPredictor=af;

% 执行交叉验证
partitionedModel = crossval(trainedClassifier.ClassificationKNN, 'KFold', 5);

% 计算验证预测
[validationPredictions, validationScores] = kfoldPredict(partitionedModel);

% 计算验证准确度
validationAccuracy = 1 - kfoldLoss(partitionedModel, 'LossFun', 'ClassifError');


end
```

附录 F：问题二：判别分析分类器 python 代码

```
    判别分析

    function [trainedClassifier, validationAccuracy] = PBFX(training-
Data,length_tezheng)

    % 提取预测变量和响应
    % 以下代码将数据处理为合适的形状以训练模型。
    %
    % 将输入转换为表

    af=[];
    for i=1:length_tezheng
        as{i}=['column_',num2str(i)];
        ap{i}=['column_',num2str(i)];
        af=[af,'false'];
    end

    as{length_tezheng+1}=['column_',num2str(length_tezheng+1)];
    inputTable = array2table(trainingData, 'VariableNames', as);
    predictorNames=ap;
    predictors = inputTable(:, predictorNames);

    eval(['response = inputTable.column_',num2str(length_tezheng+1),';']);
    isCategoricalPredictor=af;
```

```
    % 训练分类器
    % 以下代码指定所有分类器选项并训练分类器。
```

```
    classificationDiscriminant = fitcdiscr(...
        predictors, ...
        response, ...
        'DiscrimType', 'linear', ...
        'Gamma', 0, ...
        'FillCoeffs', 'off', ...
        'ClassNames', [1; 2; 3; 4; 5; 6; 7; 8; 9; 10; 11; 12]);

    % 使用预测函数创建结果结构体
    predictorExtractionFcn = @(x) array2table(x, 'VariableNames', predictorNames);
    discriminantPredictFcn = @(x) predict(classificationDiscriminant, x);
    trainedClassifier.predictFcn = @(x) discriminantPredictFcn(predictorExtrac-
tionFcn(x));

    % 向结果结构体中添加字段
    trainedClassifier.ClassificationDiscriminant = classificationDiscriminant;
    trainedClassifier.About = '此结构体是从分类学习器 R2021b 导出的训练模型。';
    trainedClassifier.HowToPredict = sprintf('要对新预测变量列矩阵 X 进行预测，请使用:
\n yfit = c.predictFcn(X) \n 将 ''c'' 替换为作为此结构体的变量的名称，例如 ''trained-
Model''。\n \nX 必须包含正好 168 个列，因为此模型是使用 168 个预测变量进行训练的。\nX 必
须仅包含与训练数据具有完全相同的顺序和格式的\n 预测变量列。不要包含响应列或未导入 App 的任
何列。\n \n 有关详细信息，请参阅 <a href="matlab:helpview(fullfile(docroot,
''stats'', ''stats.map''), ''appclassification_exportmodeltoworkspace'')">How to
predict using an exported model</a>。');

    % 提取预测变量和响应
    % 以下代码将数据处理为合适的形状以训练模型。
    %
    % 将输入转换为表



    inputTable = array2table(trainingData, 'VariableNames', as);
    predictors = inputTable(:, predictorNames);
    eval(['response = inputTable.column_',num2str(length_tezheng+1),';']);
    isCategoricalPredictor=af;

    % 执行交叉验证
    partitionedModel = crossval(trainedClassifier.ClassificationDiscriminant,
'KFold', 5);

    % 计算验证预测
    [validationPredictions, validationScores] = kfoldPredict(partitionedModel);

    % 计算验证准确度
    validationAccuracy = 1 - kfoldLoss(partitionedModel, 'LossFun', 'ClassifEr-
ror');
    end
```

附录 G: 问题二：决策树分类器 python 代码

```
    function [trainedClassifier, validationAccuracy] = JCS(trainingData,length_te-
zheng)

    % 提取预测变量和响应
    % 以下代码将数据处理为合适的形状以训练模型。
    %
    % 将输入转换为表
    af=[];
    for i=1:length_tezheng
        as{i}=['column_',num2str(i)];
        ap{i}=['column_',num2str(i)];
        af=[af,'false'];
    end

    as{length_tezheng+1}=['column_',num2str(length_tezheng+1)];
    inputTable = array2table(trainingData, 'VariableNames', as);
    predictorNames=ap;
    predictors = inputTable(:, predictorNames);
    eval(['response = inputTable.column_',num2str(length_tezheng+1),';']);
    isCategoricalPredictor=af;

    % 训练分类器
    % 以下代码指定所有分类器选项并训练分类器。
    classificationTree = fitctree(...
        predictors, ...
        response, ...
        'SplitCriterion', 'deviance', ...
        'MaxNumSplits', 67, ...
        'Surrogate', 'off', ...
        'ClassNames', [1; 2; 3; 4; 5; 6; 7; 8; 9; 10; 11; 12]);

    % 使用预测函数创建结果结构体
    predictorExtractionFcn = @(x) array2table(x, 'VariableNames', predictorNames);
    treePredictFcn = @(x) predict(classificationTree, x);
    trainedClassifier.predictFcn = @(x) treePredictFcn(predictorExtractionFcn(x));

    % 向结果结构体中添加字段
    trainedClassifier.ClassificationTree = classificationTree;
    trainedClassifier.About = '此结构体是从分类学习器 R2021b 导出的训练模型。';
    trainedClassifier.HowToPredict = sprintf('要对新预测变量列矩阵 X 进行预测，请使用:
\n yfit = c.predictFcn(X) \n 将 ''c'' 替换为作为此结构体的变量的名称，例如 ''trained-
Model''。\n \nX 必须包含正好 168 个列，因为此模型是使用 168 个预测变量进行训练的。\nX 必
须仅包含与训练数据具有完全相同的顺序和格式的\n 预测变量列。不要包含响应列或未导入 App 的任
何列。\n \n 有关详细信息，请参阅 <a href="matlab:helpview(fullfile(docroot,
''stats'', ''stats.map''), ''appclassification_exportmodeltoworkspace'')">How to
predict using an exported model</a>。');
```

```
    % 提取预测变量和响应
    % 以下代码将数据处理为合适的形状以训练模型。
    %
    % 将输入转换为表
    inputTable = array2table(trainingData, 'VariableNames', as);

    % predictorNames = {'column_1', 'column_2', 'column_3', 'column_4', 'col-
umn_5', 'column_6', 'column_7', 'column_8', 'column_9', 'column_10', 'column_11',
'column_12', 'column_13', 'column_14', 'column_15', 'column_16', 'column_17',
'column_18', 'column_19', 'column_20'};
    predictors = inputTable(:, predictorNames);
    eval(['response = inputTable.column_',num2str(length_tezheng+1),';']);
    isCategoricalPredictor=af;

    % 执行交叉验证
    partitionedModel = crossval(trainedClassifier.ClassificationTree, 'KFold', 5);

    % 计算验证预测
    [validationPredictions, validationScores] = kfoldPredict(partitionedModel);

    % 计算验证准确度
    validationAccuracy = 1 - kfoldLoss(partitionedModel, 'LossFun', 'ClassifEr-
ror');
    end
```

附录 H: 问题三：主成分分析 python 代码

```
def PCA2(data):

    df = data
    # indices =
[37,28,130,71,86,168,155,127,165,103,117,131,156,107,58,5,99,83]
    # indices = [i-1 for i in indices]
    # df= df.iloc[:,indices]
    # Bartlett's 球状检验
    chi_square_value, p_value = calculate_bartlett_sphericity(df)
    print("Bartlett's 球状检验:",chi_square_value, p_value,"\n")

    # KMO 检验
    # 检查变量间的相关性和偏相关性，取值在 0-1 之间；KOM 统计量越接近 1，变量间的相关性
越强，偏相关性越弱，因子分析的效果越好。
    # 通常取值从 0.6 开始进行因子分析
    kmo_all, kmo_model = calculate_kmo(df)
    print("KMO 检验:", kmo_all, "\n")

    # 使用列表推导找出超过 0.6 的元素的索引
```

```
    indices = [index for index, value in enumerate(kmo_all) if value > 0.6]

    # #标准化
    df = preprocessing.scale(df.iloc[:,indices])
    print("维度",np.shape(df))

    kmo_all, kmo_model = calculate_kmo(df)
    print("KMO 检验:", kmo_all, "\n")

    # #求解系数相关矩阵
    # covX = np.around(np.corrcoef(df.T),decimals=3)
    # print(covX)

    # #求解特征值和特征向量
    # featValue, featVec=  np.linalg.eig(covX.T)  #求解系数相关矩阵的特征值和特征
向量
    # print(featValue, featVec)

    average = meanX(df)
    # print(average)

    #查看列数和行数
    m, n = np.shape(df)
    # print(m,n)

    #均值矩阵
    data_adjust = []
    avgs = np.tile(average, (m, 1))
    # print(avgs)

    #去中心化
    data_adjust = df - avgs
    # print(data_adjust)

    #协方差阵
    covX = np.cov(data_adjust.T)   #计算协方差矩阵
    # print(covX)

    #计算协方差阵的特征值和特征向量
    featValue, featVec=  np.linalg.eig(covX)  #求解协方差矩阵的特征值和特征向量
    # print(featValue, featVec)
```

```
#对特征值进行排序并输出 降序
    featValue = sorted(featValue)[::-1]
    # print(featValue)

    #绘制散点图和折线图
```

```
    # 同样的数据绘制散点图和折线图
    plt.scatter(range(1, 20), featValue[0:19],s=60, marker="^",
color="#3E5D70")
    plt.plot(range(1, 20), featValue[0:19],".-",color="#FAAF78",lin-
ewidth=2,markersize=0.5)
    # 显示图的标题和 xy 轴的名字
    # 最好使用英文，中文可能乱码
    plt.xlabel("主成分")
    plt.ylabel("特征值")
    plt.grid()  # 显示网格
    plt.savefig("主成分方差贡献率.png", dpi=500)
    plt.show()  # 显示图形

    #求特征值的贡献度
    gx = featValue/np.sum(featValue)
    # print("特征值的贡献度:", np.real(gx))

    #求特征值的累计贡献度
    lg = np.cumsum(gx)
    # print(lg)

    #选出主成分
    k=[i for i in range(len(lg)) if lg[i]<0.85]
    k = list(k)
    print("累计贡献达到 85%的方差： ", k)

    #选出主成分对应的特征向量矩阵
    selectVec = np.matrix(featVec.T[k]).T
    selectVe=selectVec*(-1)
    # print("主成分对应的特征向量矩阵:", np.real(selectVec))

    #主成分得分
    finalData = np.dot(data_adjust,selectVec)
    # print("主成分得分:", np.real(finalData))

    #绘制热力图
    plt.figure(figsize = (14,14))
    ax = sns.heatmap(np.real(selectVec), annot=False, cmap="BuPu")
    # 设置 y 轴字体大小
    ax.yaxis.set_tick_params(labelsize=15)
    plt.title("主成分-因子热力图", fontsize="xx-large")
    # 设置 y 轴标签
```

```
    plt.ylabel("主成分", fontsize="xx-large")
    # 显示图片
    plt.savefig("因子分析.png", dpi=500)
    plt.show()
```

```
    return 0


# In[]
def first_pca(data):
    # 标准化数据
    scaler = StandardScaler()
    df_scaled = scaler.fit_transform(data)

    # 执行 PCA，只保留第一主成分
    pca = PCA(n_components=1)
    pca.fit(df_scaled)

    # 计算第一主成分的值
    pc1_values = pca.transform(df_scaled)

    # 创建一个新的 DataFrame，只包含第一主成分
    df_pc1 = pd.DataFrame(data=pc1_values, columns=['First_Principal_Compo-
nent'])

    # 输出结果
    print(df_pc1)

    # 如果需要，保存到 CSV 文件
    df_pc1.to_excel('附件 2 第一主成分.xlsx', index=False)
        return df_pc1
```

附录 I：问题三：方差分析 python 代码

```
# In[] 构建用于双因素方差分析的数据
PCA2(data2)
A1_data2 = first_pca(data2)


#将 data2 数据按照 ID 进行划分,按照运动状态进行循环
A1_data2 = pd.concat([T_feature2, ID_feature2, A1_data2],axis=1)

# In[]
#对 T，ID 双维度下的主成分进行双因素方差分析

# 使用 ols 方法拟合双因素方差分析模型，包括交互项
```

```
model = ols('First_Principal_Component ~ T + ID + T * ID',
data=A1_data2).fit()

# 输出模型摘要
print(model.summary())
```

```
# In[] 距离度量差异性
index_main_feature =
[91,80,114,51,76,26,8,103,122,70,164,14,119,43,6,25,136,35,148,64,156,86,90,67
,123,144,127,163,60,71,65,124]
# index_main_feature = [i-1 for i in index_main_feature]
# main_feature = feature2.iloc[:,index_main_feature]
main_feature = pd.concat([data2, A1_data2.iloc[:,0:2]],axis=1)

#将 main_feature 进行排序
main_feature_sorted = main_feature.sort_values(by="T")
#绘制热力图，计算距离矩阵
col_names = main_feature_sorted.columns
score_col_names = [[0 for i in range(len(col_names)-2)],[[] for i in
range(len(col_names)-2)]]
T_score = [[0 for j in range(12)], [[] for j in range(12)]]
# In[]
for i in range(1,13):
    for j in range(len(col_names)-2):
        #进行方差分析
        # 使用 ols 方法拟合双因素方差分析模型，包括交互项
        y = col_names[j]
        model = ols(y + ' ~ ID', data=main_feature_sorted[main_fea-
ture_sorted["T"]==i]).fit()
        # 输出模型摘要
        anova_table = sm.stats.anova_lm(model, typ=2)
        p_values = anova_table['PR(>F)']  # 或者使用列名，如果列名不同
        if p_values["ID"]<0.05:
            print("T=",i,":","指标为",y)
            print(p_values["ID"],"\n")
            score_col_names[0][j] = score_col_names[0][j]+1
            score_col_names[1][j].append(i)
            T_score[0][i-1] = T_score[0][i-1] + 1
            T_score[1][i-1].append(j)
        # print(model.summary())

# In[]
for i in range(len(score_col_names)):
    if score_col_names[i]>10:
            print(col_names[i],score_col_names[i])
```

附录 J：问题三：CNN 回归预测 python 代码

```
# In[] CNN 预测
from keras.models import Sequential
from keras.layers import Conv1D, GlobalMaxPooling1D, Dense
```

```
# 假设 X_new 是你的新输入数据，形状为(50, 12, 168)
# 假设 y_new 是你的新标签数据，形状为(50,)
X_new = ...  # 50 个 12 维×168 维的矩阵
y_new = ...  # 对应的 50 个标签

# In[]
#ID 样本重构
time = pd.DataFrame([i%5+1 for i in range(600)],columns=["times"])
train_data = pd.DataFrame()
train_data = feature2
train_data = pd.concat([feature2,time, ID_feature2,body_information_2],axis=1)
X_new = np.random.rand(50,12,168)
Y_new = np.random.rand(50,4)

num=0
for i in range(4,14):
    for j in range(1,6):
        temp =
train_data[train_data["ID"]==i][train_data["times"]==j].iloc[:,0:168]
        temp_label =
train_data[train_data["ID"]==i][train_data["times"]==j].iloc[:,-4:]
        X_new[num]=temp
        Y_new[num][:]=temp_label.iloc[0,:]
        num = num + 1

test_data = pd.DataFrame()
test_data = feature5
temp_ID = [i//12+1 for i in range(60)]
temp_ID = pd.DataFrame(temp_ID, columns=["ID"])
test_data  = pd.concat([test_data, temp_ID],axis=1)
X_pre = np.random.rand(5,12,168)
Y_pre = np.random.rand(5,4)

for i in range(5):
    X_pre[i]=test_data.iloc[i*12:(i+1)*12, 0:168]

# 模型训练——ID 预测
#定义 CNN 模型，注意 input_shape 的修改
model = Sequential()
model.add(Conv1D(filters=64, kernel_size=3, activation='relu', in-
put_shape=(12, 168)))
```

```
model.add(GlobalMaxPooling1D())  # 使用全局最大池化简化特征
model.add(Dense(64, activation='relu'))
# 根据任务类型调整最后的 Dense 层
model.add(Dense(1,activation="relu"))  # 假设是回归任务，输出一个连续值
```

```
# 编译模型，选择合适的损失函数和优化器
model.compile(optimizer='adam', loss='mean_squared_error')

for i in range(4):

    model.fit(X_new, Y_new[:,i], epochs=10, batch_size=5,valida-
tion_split=0.2)  # 调整 batch_size 以适应新数据集大小

    # 评估模型
    loss = model.evaluate(X_new, Y_new[:,i])
    print(f'Test loss: {loss}')

    #预测模型
    #利用训练好的模型进行预测
    Y_pre[:,i] = np.reshape(model.predict(X_pre),(5,)) # 使用模型进行预测
        print(Y_pre)
```