


中国研究生创新实践系列大赛

“华为杯”第二十届中国研究生

数学建模竞赛

学　　校	
参赛队号	
队员姓名	1.
	2.
	3.

中国研究生创新实践系列大赛/

“华为杯”第二十届中国研究生

数学建模竞赛

题 目： 出血性脑卒中临床智能诊疗建模与分析

摘 要：

本文针对 100 名患者（sub001--sub100）的病例，对出血性脑卒中于题目给定定义的血肿扩张的发生进行了概率分析，基于血肿扩张的时间和概率进而对治疗干预和血肿周围水肿的发生和进展之间的关系展开探究，又根据前 100 名病例的数据分析对患者（sub101--sub160）的预后和影响预后的关键性因素进行了预测，建立了**基于随机森林的血肿扩张预测模型、基于高斯拟合的水肿体积随时间变化模型、基于 XGBoost 的预后恢复水平预测模型**。

首先，本文根据患者入院时的首次影像及 **48 小时内**随访检查的影像结果,判断 48 小时内是否发生血肿扩张事件（后续检查比首次检查绝对体积增加≥6ml 或相对体积增加≥33%），并记录下发生血肿扩张的患者编号及对应的检测时间。然后利用患者的个人史、疾病史等多源特征，建立逻辑回归和随机森林模型对所有患者发生血肿扩张的概率进行预测，并比较两者的性能，最终选择使用随机森林模型，F1 分数达到 **0.78**，Accuracy 达到 **0.83**。

本文通过高斯函数拟合模型分析了水肿体积随时间的变化趋势，并基于无监督聚类算法对患者进行分组，获得了中年、中高龄及高龄三组患者的水肿体积子模型，并比对有无聚类算法下，预估的值和实际值的残差的差距，得到了经过聚类后拟合的子模型性能比原始模型优秀这一结论。为评价不同治疗方式对水肿体积的影响,采用方差分析法进行假设检验，并建立**多层感知器（MLP）模型**对方差分析法的结果进行验证，结果表明降颅压治

疗和降压治疗对水肿进展有显著影响。

最后，建立基于 XGBoost 预测患者（sub001--sub100）90 天的神经功能（mRS）评分的模型，得到模型的性能 MSE 为 3.7，MAE 为 1.6，RMSE 为 1.92。在建立模型的过程中发现患者的神经功能恢复与脑部血肿的体积密切相关，患者的其它疾病也会对神经功能恢复造成影响，其中一些还有较强的关联性，并通过相关性分析，发现血肿体积、舒张压与饮酒史等多种因素均与神经功能恢复相关。该分析结果可为出血性脑卒中后继发性损伤的预测及个体化治疗方案的制定提供依据。

关键词： XGBoost 随机森林 智能诊疗 出血性脑卒中 方差分析法 相关性分析 多层感知器 高斯拟合

1 引 言

出血性脑卒中指非外伤性脑实质内血管破裂引起的脑出血，占全部脑卒中发病率的10-15%。其病因复杂。出血性脑卒中起病急、进展快，预后较差，急性期内病死率高达45-50%，约 80%的患者会遗留较严重的神经功能障碍，为社会及患者家庭带来沉重的健康和经济负担。因此，发掘出血性脑卒中的发病风险，整合影像学特征、患者临床信息及临床诊疗方案，精准预测患者预后，并据此优化临床决策具有重要的临床意义。

出血性脑卒中后，血肿范围扩大是预后不良的重要危险因素之一。此外，血肿周围的水肿作为脑出血后继发性损伤的标志，在近年来引起了临床广泛关注。针对出血性脑卒中后的两个重要关键事件，即血肿扩张和血肿周围水肿的发生及发展，进行早期识别和预测对于改善患者预后、提升其生活质量具有重要意义。

医学影像技术的飞速进步，为无创动态监测出血性脑卒中后脑组织损伤和演变提供了有力手段。近年来，迅速发展并广泛应用于医学领域的人工智能技术，为海量影像数据的深度挖掘和智能分析带来了全新机遇。

本文根据患者的影像信息，联合患者个人信息、治疗方案和预后等数据，构建智能诊疗模型，明确导致出血性脑卒中预后不良的危险因素，实现精准个性化的疗效评估和预后预测。

2 问题分析

经过一系列神经影像学所观察到，血肿扩张（HE：hematoma expansion）是指血肿随着时间的推移而增长。[1]曾经认为脑出血发生几分钟后就止血了，但近期随着科学和医学的发展发现血肿扩张是脑出血的一种常见现象，据报道，脑出血症状出现 6 小时内 HE 的发生率从13%到38%不等，这在一定程度上是由于不同研究对HE的定义以及血肿检测间隔时间不同导致的。[2]本文通过建模分析，对于已知治疗方法以及多源因素对血性脑产中的治疗效果和预后的关联性进行分析，对患者(sub101--sub160) 的病情走向进行预测，为临床相关决策提出建议。

2.1 问题 1 分析

(a) 已知数据：入院首次影像检查流水号(SN)，发病到首次影像检查时间间隔(T_0)，各时间点流水号及对应的HM_volume。

解：判断患者 sub001 至 sub100 发病后 48 小时内是否发生血肿扩张事件。

根据题目信息分析出随访随访时间点与随访距离发病时间的推导公式，因为各流水号有对应的随访时间点，所以对于同一个病患，随访 i 距发病所差小时数T_i有以下公式：

$$T_i = T_0 + (P_i - P_0) \qquad (i \geq 0) \qquad (1)$$

式中P_i是第 i 个随访时间点，P_0是入院首次检查时间点。

根据该公式，可以计算每个病患每次随访距离发病的时间。

(b) 经过上面的分析，我们已将每个病患每次随访距离发病的时间与HM_volume联系了起来，根据题目的定义，当HM_volume发生以下任一一种情况时算作发生血肿扩张。

$$（V_i - V_0）\times 10^{-3} \geq 6（ml）\qquad (2)$$

$$（V_i - V_0）\div V_0 \times 100\% \geq 33\% \qquad (3)$$

其中V_i是第 i 次随访检查的 HM_volume 的值，即第 i 次随访检查的血肿的体积；V_0是入院首次检查 HM_volume 的值，即首次检查血肿的体积。

已有报道中[2]，HE 的发生率由于没有统一的定义以及患者随访时间点的不同，测得数据差异很大。从一项用了常用 HE 定义的分析中可以看出，从症状出现开始，前 3 小时内出现 HE 的比例高达 73%，临床上显著的 HE 发生率为 35%。另一项研究报道，随着发病至诊断 CT 时间的延长，HE 的发生率逐渐下降，即在发病前 3 小时内 HE 发生率为 36%，3-6 小时为 16%，6-12 小时仅为 6%，24-48 小时内无患者出现 HE。本文根据题目提供的 sub001-sub100 的 100 名患者信息分析了他们在 48 小时内是否发生 HE 以及发生了 HE 的时间。并且对 sub101-sub160 的患者 48 小时内是否发生 HE 进行了预测。该问题是一个多自变量，单一因变量的问题，且因变量是概率，即是一个 0~1 的数。

2.2 问题 2 分析

血肿周围水肿(perihematomal edema，PHE)在脑出血超早期阶段迅速增长，有动物实验表明，血肿周围水肿量起初增长很轻微，2h 后开始增多，3～4d 达到高峰，随后水肿缓慢下降，直到出血后 7d 仍然存在[3]。在发病 24h 内，PHE 体积较基线可增长至 75%~100%[4]。经过快速增长期（1~3 d），PHE 的生长速度逐渐减慢，直到发病后 1~2 周达到峰值，并进入吸收阶段[5]。

(a)自变量：发病至影像检查时间，因变量：水肿体积。该问题是一个单一自变量，单一因变量的问题。对于这种问题，通常情况下可以建立函数进行求解。

(b)自变量和因变量与 a 小问相同，只不过要先聚类分析。

(c)对出血性卒中引起的脑部水肿，有脑室引流，止血治疗，降颅压治疗，降压治疗，镇静、镇痛治疗，止吐护胃，营养神经这几种治疗方式，这是比较一个多组之间差异问题。

(d)施加各种治疗手段后，对血肿体积及水肿体积的首次变化（即第一次随访与入院检查的差值）进行特征重要性分析。多自变量，单一因变量问题。

2.3 问题 3 分析

在 0.5 至 3 小时之间，下降幅度最大，表明脑出血在前 3 小时内发生的可能性更大。因此，脑出血是一种动态疾病(图 1)。根据问题 1 题目定义出血总体积大于 6ml 时为发生血肿扩张，由图 2.1 可知患者接受治疗后发生 HE 的概率降低。因此，一种有临床意义的解释可能是，患者越早接受药物治疗，治疗策略越有可能有效降低脑出血的风险。

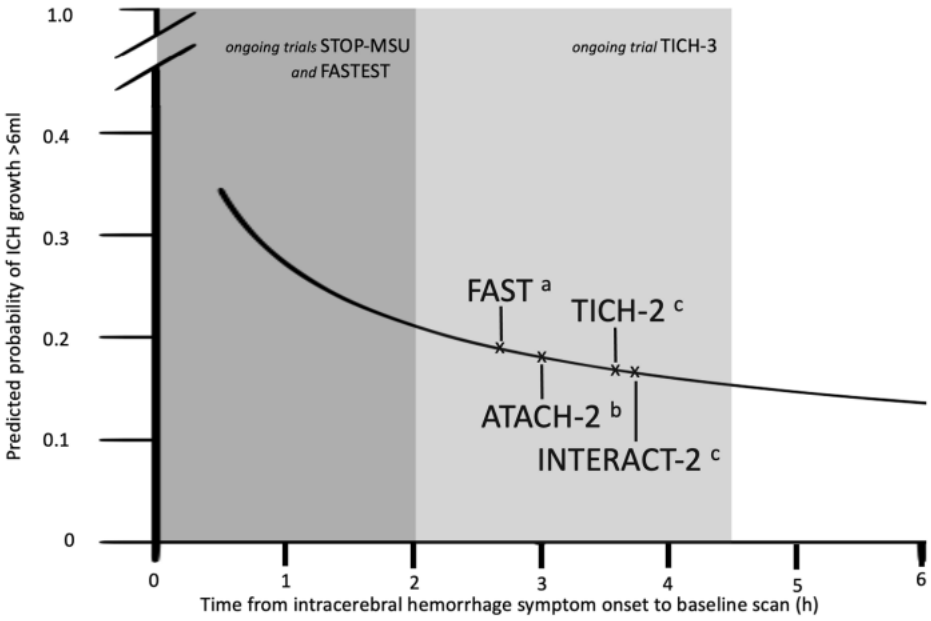

图 2.1 从症状出现到基线成像时间预测脑出血生长>6ml 的概率根据 al - shahi Salman 等重建的主要试验及其各自的(a)从发病到治疗的平均时间(b)从发病到随机化的平均时间和(c)从发病到随机化的中位时间。此外，正在进行的试验的时间窗(阴影部分)[1]

(a)自变量：1、前 100 个患者的个人史、疾病史、发病相关，2、首次影像结果。因变量：预测患者 90 天 mRS 评分。

同样也是多自变量单一因变量的问题，而因变量也是非 0~1。

(b)对上一小问建立的模型的数据集进行了增强，应对先前的模型进行优化。

(c)分析血肿体积、水肿体积和治疗方法三者之间的关系，相关性分析。

3 模型假设

1. 患者在初次入院时没有伴随其他疾病发生；
2. 数据独立性假设，每个患者的数据点不受其他患者的数据点影响；
3. 本研究在选择特征时，假设选择的特征与最终的目标存在相关性，即这些特征包含了相关的信息；
4. 本研究的机器学习模型的建立基于训练数据，并假设模型能够泛化到未见过的数据；
5. 本研究假设不同类别之间的样本是平衡的。

4 符号说明

符号	意义	单位
T_0	发病到首次影像检查时间间隔	小时/h
P_0	入院首次检测时间点	
P_i	第 i 个随访时间点	
V_i	第 i 次随访检查的 HM_volume	10^{-3}mL
V_0	入院首次检查 HM_volume	10^{-3}mL
ε	模型预测的误差或噪声	
$\lvert C_j \rvert$	簇C_j中的数据点数量	
SSb	组间差异	
SSw	组内差异	
$z_j^{(l)}$	第 l 层的神经元 j	
$w_{ji}^{(l)}$	第 l-1 层的神经元 i 到第 l 层的神经元的权重	
$a_i^{(l-1)}$	第 l-1 层的神经元 i 的输出	
$b_j^{(l)}$	第 l 层的神经元 j 的偏差项	
$n^{(L)}$	第 l 层的神经元数量	
ρ	斯皮尔曼相关系数	
d	排名差	
r	皮尔逊相关系数	

5 模型建立与求解

5.1 问题 1 模型建立与求解

5.1.1 判断患者 48h 是否发生血肿扩张的计算结果

据分析评判标准，首先计算 i=1 的情况下，发生血肿扩张的患者，总计 19 位患者，这 19 位患者的第一次随访距离发病时间均小于 48h。然后，计算 i=2 的情况下，发生血肿扩张的患者，并排除掉 i=1 时就发生 HE 的患者。此时对第二次随访距离发病时间小于 48h 的进行标注，得到 sub003，sub039，sub061，sub099 号病患。对于所有发生 HE 的患者，记录下对应的随访距离发病时间，填写表格。

5.1.2 所有患者（sub001 至 sub160）发生血肿扩张的概率建模与分析

1 回归任务分析

回归任务是一种监督式机器学习问题，其目标是建立一个函数或模型，将输入特征x映射到连续型目标变量y的预测值。回归任务可以用函数关系式表达为：

$$y = f(x) + \varepsilon \quad (4)$$

其中：

x：输入特征向量。包含多个特征$x^1, x^2, \ldots, xn$，表示待预测的样本的属性；

y：连续型目标变量。表示我们希望预测的数值；

f：未知的真实函数。表示特征与目标变量之间的关系；

ε：表示模型预测的误差或噪声。即模型无法完全准确地预测真实目标变量的部分。

回归任务的目标是通过观察和学习一组已知的训练样本(xi, yi)，其中$i = 1, 2, \ldots, m$，来估计或逼近真实函数f。

本次目标是想找到一个函数或模型$h(x)$，既能高度拟合已知训练样本的 100 例患者（sub001-sub100），又能利用首次影像检测数据对未知样本（sub101-sub160）的 HE 概率进行准确的预测。

通常处理回归任务的算法有：线性回归，随机森林，逻辑回归。对于多自变量的回归问题，线性回归的拟合能力较差。又因为最终的预测结果是一个概率，首先使用逻辑回归模型进行拟合。

逻辑回归使用 Sigmoid 函数将线性组合的结果映射到概率空间（0 到 1 之间）。Sigmoid 函数的“S”型曲线能够平滑地将任意实数值转化为 0 到 1 之间的概率值。逻辑回归模型的参数（权重）具有直观的解释性，能够反映出每个特征对于预测的影响程度。这方便我们更直观的观察到在模型中起到了关键作用的因素。

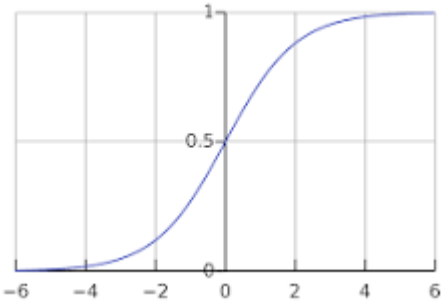

图 5.1　Sigmoid 函数

2 随机森林算法分析

随机森林是一种基于集成学习（Ensemble Learning）的方法，它通过组合多个决策树来进行预测。随机森林算法[6]基本原理是采用 Bootstrap 子自采样的方法获得不同的样本集用于构建模型，从而增加了模型之间的差异度并提高了外推预测的能力。对于属性选择:首先,从基决策树结点的属性集合中随机选择一个包含 k 属性的子集(k 通常为 log2d，其中 d 为属性集合的大小）；其次，从这个子集中选择一个最佳属性用于划分；最后，采用多数结果法、平均法、投票法等综合考虑多个决策树的预测结果确定最终随机森林算法的预测结果[7]。

3 回归任务分析结果

经过分析原始数据，发现表 1 中，性别和血压栏可以分别通过赋值及拆分高低压的方法来保持与表一中其他数据的连续性。所以我们对表 1 中，性别男赋 1，女赋 0，血压的高低压拆分为两列以用于后续处理。

根据题目的要求，我们决定分三次提取自变量——特征数据，第一次针对表 1 将 1 到 100 号人员,年龄到营养神经的值提取出来,总计 20 列作为 df1 这个数据框;第二次表 2，将 1 到 100 号人员，'HM_volume'到'ED_Cerebellum_L_Ratio'的值提取出来，作为 df2 数据框;第三次将表 2 的前 100 行提取出来检查首次检查流水号,与表 3 的流水号进行匹配，对匹配结果单独保存，得到的即包含对应患者首次影像检查结果。

而对于因变量——血肿扩张发生与否，我们在上一问已经将 1 到 100 号人员做好了标记，0 为未发生，1 为发生。问题简化为简单的回归问题。

使用逻辑回归模型，划分训练集与测试集 7:3，进行训练，为了便于后续建模，也画出了特征重要性柱状图如下：

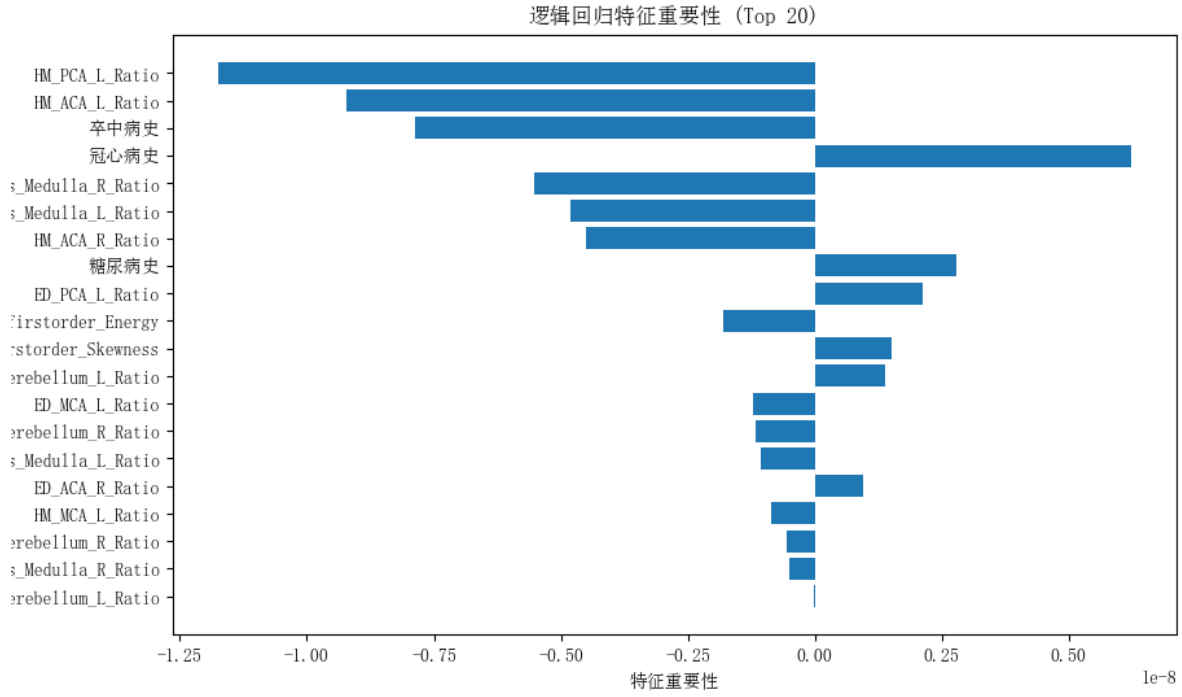

图 5.2 逻辑回归特征重要性柱状图

读图可知逻辑回归模型将左侧大脑后动脉（HM_PCA_L_Ration）和左侧大脑前动脉（HM_ACA_L_Ration）上血肿的体积占比归结为负面影响，说明在当前模型下左侧大脑后

动脉与左侧大脑前动脉上的血肿体积越大，血肿扩张发生的几率越小。

4 随机森林算法分析结果

我们发现逻辑回归只能找出负面影响，以此，促使转向另一个机器学习方法：随机森林。采用与逻辑回归相同的训练方法与评估准则，并画出特征重要性图。

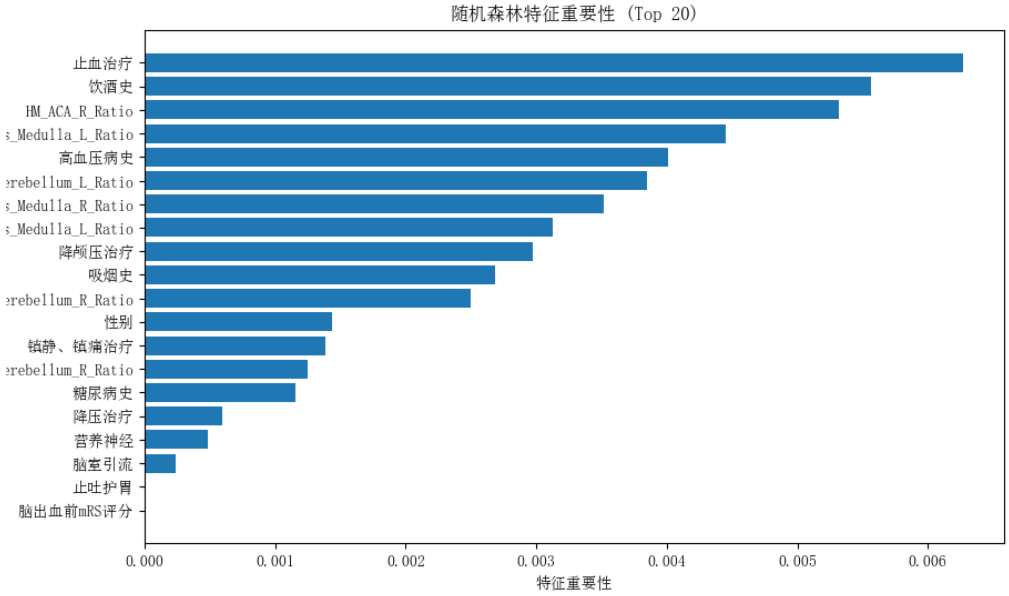

图 5.3 随机森林特征重要性柱状图

上图显示，此时重要性最高的为止血治疗这一治疗方法，而疾病史中的饮酒史为第二，根据现有研究结论，约 8%的缺血性卒中和 16%的脑出血与饮酒相关[8]，印证了饮酒对血肿体积增大的影响。

图 5.4 反映了两个模型 ROC 曲线，可以看到两者的 AUC 差距较小，只有不到 0.2。

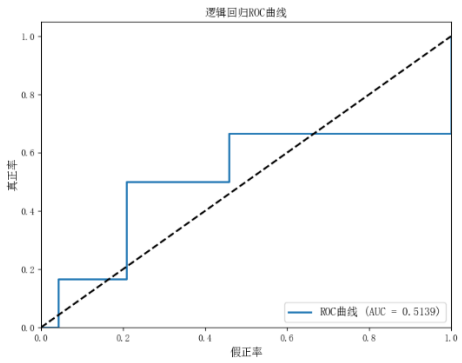

图 5.4.1 逻辑回归 ROC 曲线

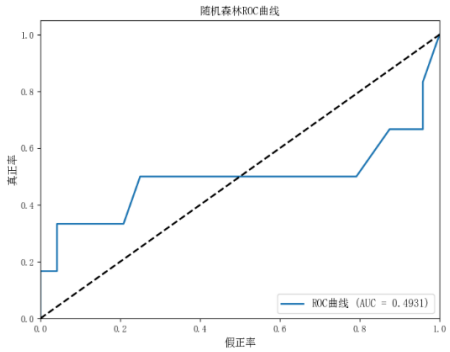

图 5.4.2 随机森林 ROC 曲线

图 5.4 两种机器学习算法 ROC 曲线

表 5-1 不同算法性能比较

Model	Accuracy	Precision	Recall	F1 score
逻辑回归	0.8	0.5	0.17	0.76
随机森林	**0.83**	**1**	**0.17**	**0.78**

根据上表，可以看出随机森林算法在准确率和 F1 分数上均高于逻辑回归，虽然逻辑回归的 AUC 较高，但是精确率太低，所以本文使用随机森林做预测模型。

5 模型预测

基于以上预测模型，我们同时计算得到了相关指标的数值，并对样本的 160 组数据进行预测，得到 160 个患者的血肿扩张预测概率。

5.2 问题 2 模型的建立与求解

5.2.1 水肿体积随时间变化模型

1 高斯模型

高斯模型通过利用正态分布曲线精确描述事物，将其分解成基于高斯概率密度函数形成的模型。正态分布是概率论中一个重要的连续概率分布，其密度函数具有钟形曲线，以均值（μ）为中心，标准差（σ）为宽度，其概率密度函数为：

$$f(x;\mu,\sigma)=\frac{1}{\sqrt{2\pi}\sigma}e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{5}$$

其中：

x：随机变量，μ：是均值，σ：是标准差。

高斯拟合通过调整参数（均值、标准差）来使得拟合曲线最符合实际数据分布。通常采用最小二乘法或最大似然估计等优化算法来寻找最优参数。

2 高斯模型拟合结果

首先算出随访 i 距发病所差小时数，该计算过程已在第 1 个问题的第一部分进行展示。然后，将患者 ID 与之对应的每次随访的时间（距离发病时间），和对应时刻的水肿体积（ED_Volume）提取出来，因为原始数据中有些病人并不是完成了所有的 12 次随访，且每个病人的最后一次随访（第 t 次随访）检查出的水肿体积相较于第（t-1）次随访都有下降，没有出现增长的趋势，故可以认为，所有患者最终都可痊愈。继续推理，我们对表格中缺失的 ED_Volume 都进行了赋 0 的处理，而对于缺失的随访 i 距发病所差小时数，进行了填充无穷大的做法，这样可以排除缺失值的干扰。针对题目指定一条全体患者水肿体积随时间进展曲线，使用高斯模型进行拟合。

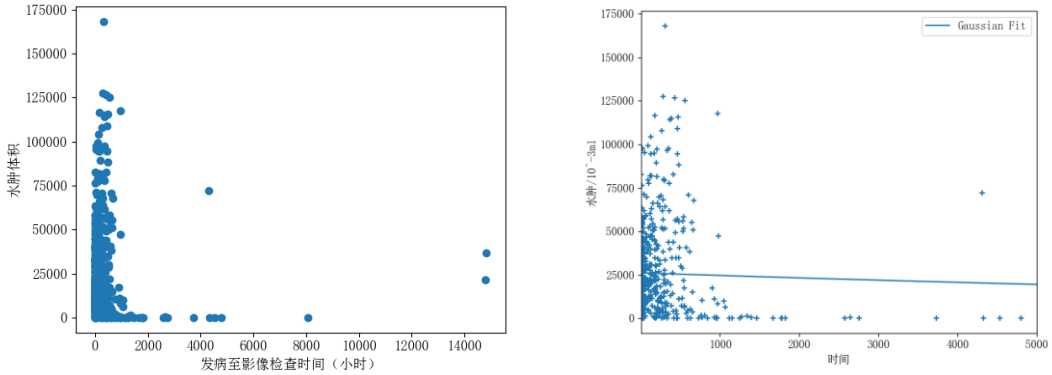

图 5.5 发病至影像检查时间——水肿体积散点及高斯拟合曲线

经过拟合，此时的 x 轴为发病至影像检查时间，y 轴为水肿体积，以拟合出的高斯曲线为预测函数，可以计算出前 100 个患者真实值和拟合曲线之间的残差。

5.2.2 水肿体积随时间变化（亚组）模型

1 K-means 聚类算法

K-means 聚类算法是一种常用的无监督学习算法，用于将一组数据点分成具有相似特征的不同群集（簇）。它通过迭代的方式将数据点分配给最近的簇，并更新簇的中心点，直到达到收敛状态。

数学上，K-means 聚类算法可以表示为以下步骤：

（1）初始化：选择要分成的簇的数量 K，并随机选择 K 个数据点作为初始的簇中心。令簇中心为 $C_1, C_2, \ldots, C_K$

（2）分配数据点：对于每个数据点 X_i，计算它与各个簇中心的距离，并将其分配给距离最近的簇。令$d(X_i, C_j)$表示数据点X_i到簇中心C_j的距离。则数据点X_i属于第j个簇的概率为：

$$P(C_j|X_i) = \begin{cases} 1 & \text{if } j = \arg\min_k d(X_i, C_k) \\ 0 & \text{otherwise} \end{cases} \quad (6)$$

（3）更新簇中心：对于每个簇C_j，计算其中所有数据点的平均值，将该平均值作为新的簇中心：

$$C_j = \frac{\sum_{X_i \in C_j} X_i}{|C_j|} \quad (7)$$

其中：

$|C_j|$：簇C_j中的数据点数量。

（4）重复分配数据点和更新簇中心，直到簇中心不再改变或达到预定的迭代次数。K-means 聚类算法通过迭代的方式不断优化簇中心和数据点的分配，使得同一簇内的数据点更加相似，不同簇之间的数据点更加不相似。最终得到的簇划分可以帮助我们发现数据的内在结构和模式。

在聚类后，将聚类后每簇患者的数据都提取出来，重复 2(a)的做法分别做 3 次多项式拟合，结果如下，Cluster_0 为高龄，Cluster_1 为中年，Cluster_2 为中高龄。拟合后的三个函数再代入对应的每个患者的随访记录，进行残差计算。

2 K-means 聚类算法计算及结果

针对该问题本文首先进行聚类分析。在分析问题 3 的(a)小问时得知年龄是患者 90 天 mRS 评分模型中的最重要的特征。所以本问题对表 1 中，患者年龄这一特征进行 K-means 聚类算法，筛选出 3 个簇，分别对应中年人、中高龄和高龄。

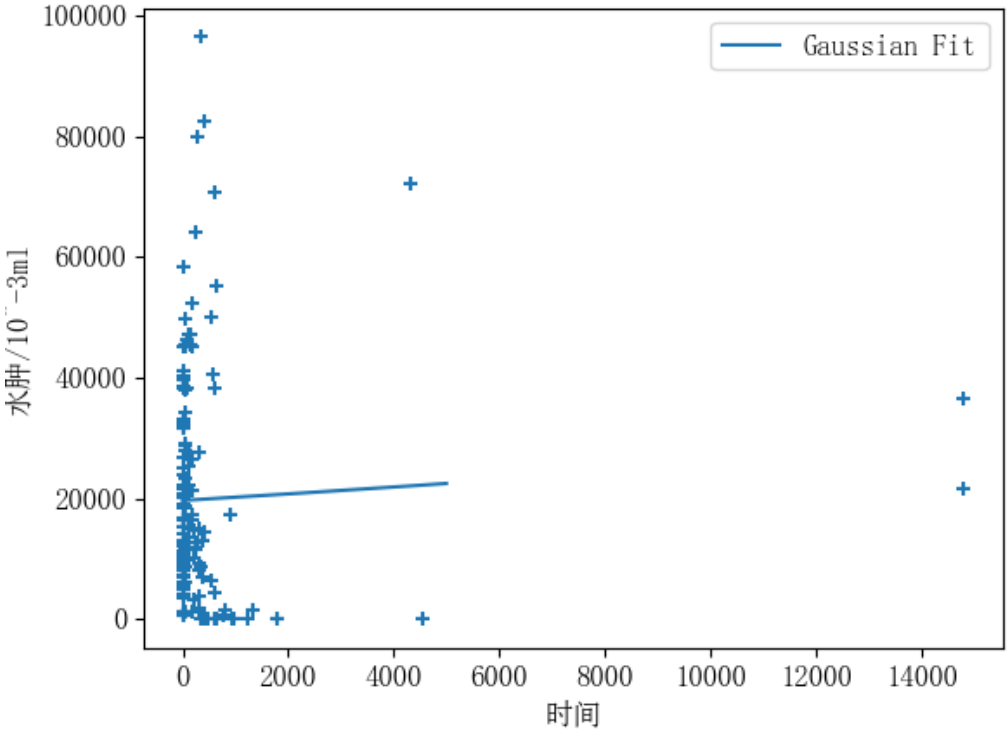

图 5.6 高龄人群发病至影像检查时间——水肿体积高斯拟合曲线

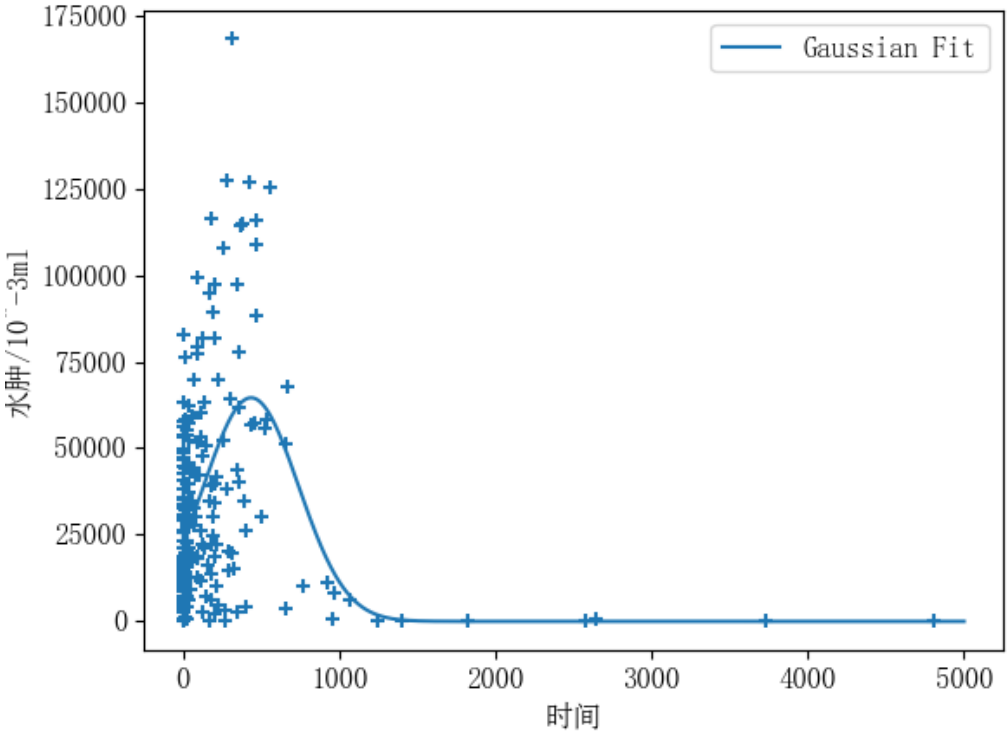

图 5.7 中年人群发病至影像检查时间——水肿体积高斯拟合曲线

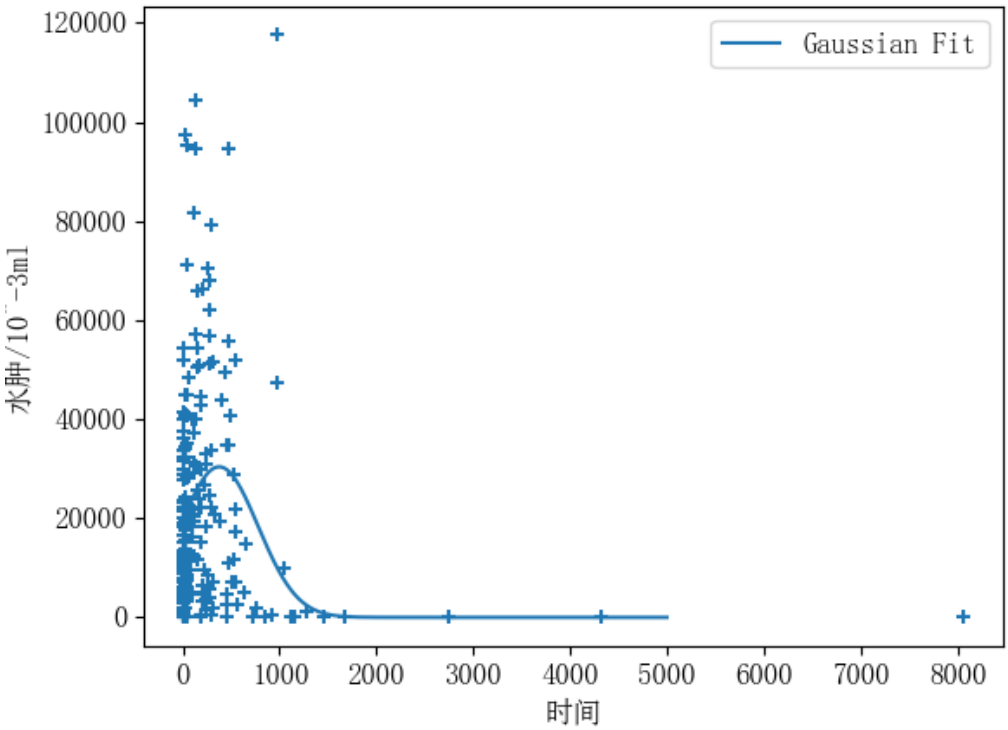

图 5.8 中高龄人群发病至影像检查时间——水肿体积高斯拟合曲线

除此以外，为了解对水肿体积与随访时间点的联系，我们还对随访时间点——每次随访的平均水肿体积用 9 次多项式做了多项式拟合。比对后可以看到，高斯拟合的结果除了高龄人群的拟合曲线与随访时间点-平均水肿体积拟合曲线趋势不一致，对于中年，中高龄人群，曲线的趋势大体一致。

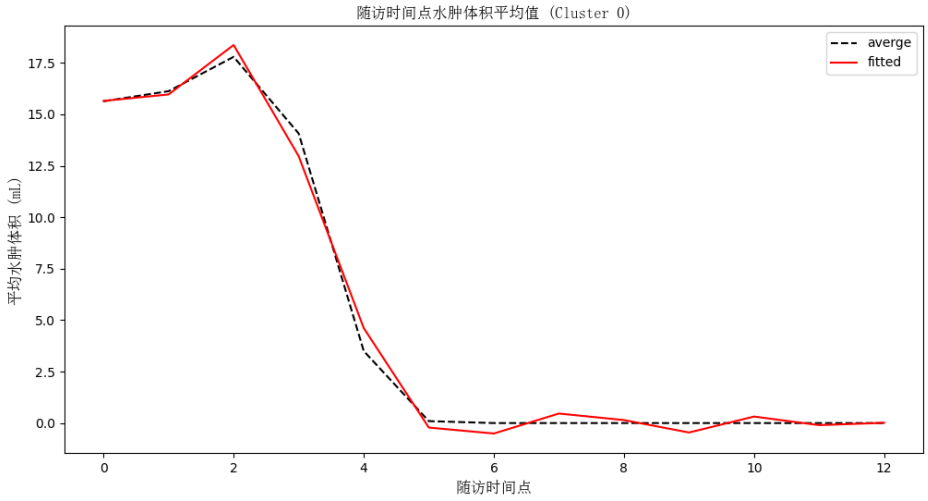

图 5.9 高龄人群随访时间点——平均水肿体积拟合曲线

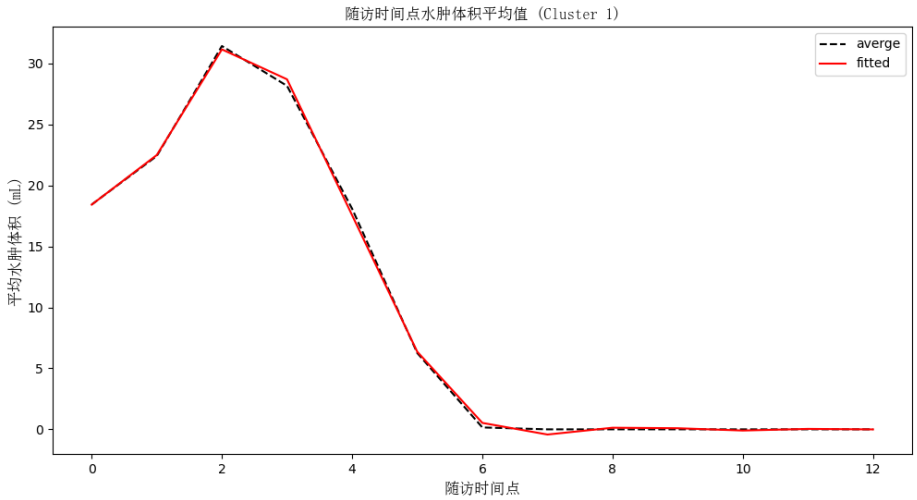

图 5.10 中年人群随访时间点——平均水肿体积拟合曲线

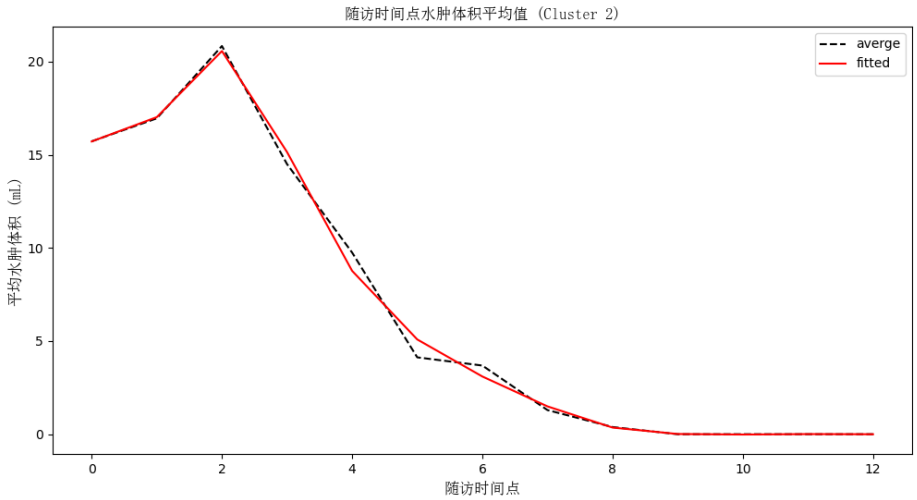

图 5.11 中高龄人群随访时间点——平均水肿体积拟合曲线

读图可知，中高龄人群随着随访时间点的推进平均水肿体积的下降最缓慢，说明中高龄人群接受治疗后水肿消除最慢。所有的年龄段下，均会在第二次随访达到平均水肿体积的最高点，然后下降；其中，高龄老人在遭受水肿扩张后恢复的最快，因为曲线梯度在 2 到 4 次随访中最大。中年和中高龄年龄段的人最终都会随着随访时间点的推进而逐渐消除水肿。

5.2.3 不同治疗方法对水肿体积进展模式的影响分析

1 方差分析

ANOVA（方差分析，Analysis of Variance）是一种对多个水平或多组样本之间期望的差异进行显著性检验的方法。方差分析的基本原理是认为不同处理组的均数间的差别基本来源有两个：

(1) 实验条件，即不同的处理造成的差异，称为组间差异。用变量在各组的均值与总均值之偏差平方和的总和表示，记作SSb，组间自由度dfb。

(2) 随机误差，如测量误差造成的差异或个体间的差异，称为组内差异，用变量在各组的均值与该组内变量值之偏差平方和的总和表示， 记作SSw，组内自由度dfw。

总偏差平方和

$$SSt = SSb + SSw \quad (8)$$

组内SSw、组间SSb除以各自的自由度(组内$dfw = n - m$，组间$dfb = m - 1$，其中 n 为样本总数，m 为组数)，得到其均方MSw和MSb，一种情况是处理没有作用，即各组样本均来自同一总体，$MSb/MSw \approx 1$。另一种情况是处理确实有作用，组间均方是由于误差与不同处理共同导致的结果，即各样本来自不同总体。那么，$MSb \gg MSw$。

MSb/MSw比值构成 F 分布。用 F 值与其临界值比较，推断各样本是否来自相同的总体。

2 方差分析结果

我们在显著性 0.05 的水平下，判断组间均值差异存在与否，以判断不同治疗手段对水肿体积进展的影响。分析结果如下表所示,

表 5-2 不同治疗手段 ANOVA 分析

Method	F-statistic	p-value	Y/N
脑室引流	0.71	0.40	N
止血治疗	2.55	0.11	N
降颅压治疗	**6.38**	**0.01**	**Y**
降压治疗	**4.67**	**0.03**	**Y**
镇静、镇痛治疗	3.53	0.06	N
止吐护胃	0.01	0.94	N
营养神经	0.35	0.55	N

可以看到，降颅压治疗和降压治疗在显著性 0.05 的水平下可以认定为是有效果的。为了验证 ANOVA 分析的结果，我们还建立了一个多层感知器（MLP）模型，并对每个治疗方法依次进行训练，并用评估指标加以评估。

3 多层感知器模型概念

多层感知器（Multilayer Perceptron，MLP）是一种基本的人工神经网络模型，通常用于解决分类和回归问题。MLP 是由多个神经元组成的神经网络，其中神经元分布在多个层中，包括输入层、隐藏层和输出层。

输入层（Input Layer）：输入层接受外部输入数据。每个输入神经元对应输入特征的一个维度，输入层的神经元数量由特征的维度决定。

隐藏层（Hidden Layer）：MLP 至少有一个或多个隐藏层，每个隐藏层包含多个神经元。隐藏层的主要作用是通过权重调整和激活函数对输入信号进行非线性变换。

前向传播（Forward Propagation）：在前向传播阶段，输入信号经过一系列线性和非线性变换，从输入层传递到输出层。每层的输出将作为下一层的输入。

输入层不进行任何计算，直接将输入特征传递给下一层：

$$a^{(1)} = x \tag{9}$$

对于第 l 层的神经元 j，其输入$z_j^{(l)}$可以表示为：

$$z_j^{(l)} = \sum_{i=1}^{n^{(l-1)}} w_{ji}^{(l)} \cdot a_i^{(l-1)} + b_j^{(l)} \tag{10}$$

其中：$w_{ji}^{(l)}$表示第 l-1 层的神经元 i 到第 l 层的神经元的权重，$a_i^{(l-1)}$表示第 l-1 层的神经元 i 的输出，$b_j^{(l)}$表示第 l 层的神经元 j 的偏差项。

之后将$z_j^{(l)}$通过激活函数 f 进行非线性变换，得到第 l 层的输出$a_j^{(l)}$：

$$a_j^{(l)} = f\left(z_j^{(l)}\right) \tag{11}$$

对于输出层，可以类似的计算其输入和输出：

$$z_j^{(L)} = \sum_{i=1}^{n^{(L-1)}} w_{ji}^{(L)} \cdot a_i^{(L-1)} + b_j^{(L)} \tag{12}$$

$$a_j^{(L)} = f\left(z_j^{(L)}\right) \tag{13}$$

其中 L 表示网络的总层数，$n^{(L)}$表示第 l 层的神经元数量。最后，通过这个过程，我们可以得到 MLP 对于给定输入的输出。

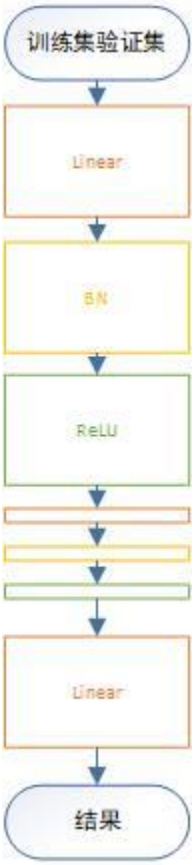

图 5.12 本实验所建立的 MLP 模型结构

主要评估指标有 MAE，R2 分数，解释分数，最大误差。虽然性能指标按照要求预测准确值的水平来看较差，但是可以判定降压与降颅压治疗是所有治疗中最有效的。

表 5-3 不同治疗手段 MLP 分析

Method	MAE	R2 Score	Explained variance	Max error
脑室引流	6708.19	0.03	0.04	31092
止血治疗	7080.78	-0.02	-0.015	32350.03
降颅压治疗	**5799.92**	**0.11**	**0.11**	**32970.06**
降压治疗	**6511.78**	**0.23**	**0.23**	**24269.81**
镇静、镇痛治疗	**6525.81**	**0.13**	**0.13**	**27722.67**
止吐护胃	6836.71	-4.90	0.0	30704.58
营养神经	6814.65	0.00	0.01	31446.02

5.2.4 血肿体积、水肿体积和治疗方法三者之间的关系

PHE 是脑出血后继发的最严重的脑损伤之一，是二次脑损伤发生的关键因素，与脑出血后神经功能的下降密切相关。[3]在施加各种治疗手段后，本文对血肿体积及水肿体积的首次变化（即第一次随访与入院检查的差值）进行特征重要性分析。

1 XGBoost

针对多自变量，单一因变量，且因变量并不是 0~1 的概率问题，我们使用 XGBoost[9]。XGBoost（eXtreme Gradient Boosting）是一种基于梯度提升树的机器学习算法，它在许多机器学习竞赛和实际应用中表现出色。XGBoost 结合了梯度提升框架和正则化技术，通过迭代训练一系列弱学习器（通常是决策树），以最小化损失函数并提高预测性能。

梯度提升树是一种集成学习算法，通过迭代训练多个弱学习器来构建一个强大的预测模型。梯度提升树的核心思想是通过逐步优化损失函数的梯度来改进模型的预测能力。每个弱学习器都尝试纠正前一个学习器的错误，并在残差之间构建关联，以逐步减少模型的预测误差。弱学习器是指预测性能相对较弱的学习算法或模型，如决策树。而 XGBoost 使用决策树作为基本的弱学习器，通过组合多个决策树来构建更强大的模型。

图 5.13 是血肿变化特征重要性图，图 5.14 是水肿变化特征性图，结果表明，降颅压治疗、止血治疗与镇静、镇痛治疗都对血肿和水肿的快速缩小起到较好的效果。所以针对出血性卒中患者，应当优先施加这三种治疗手段，能有效降低血肿和水肿的体积。

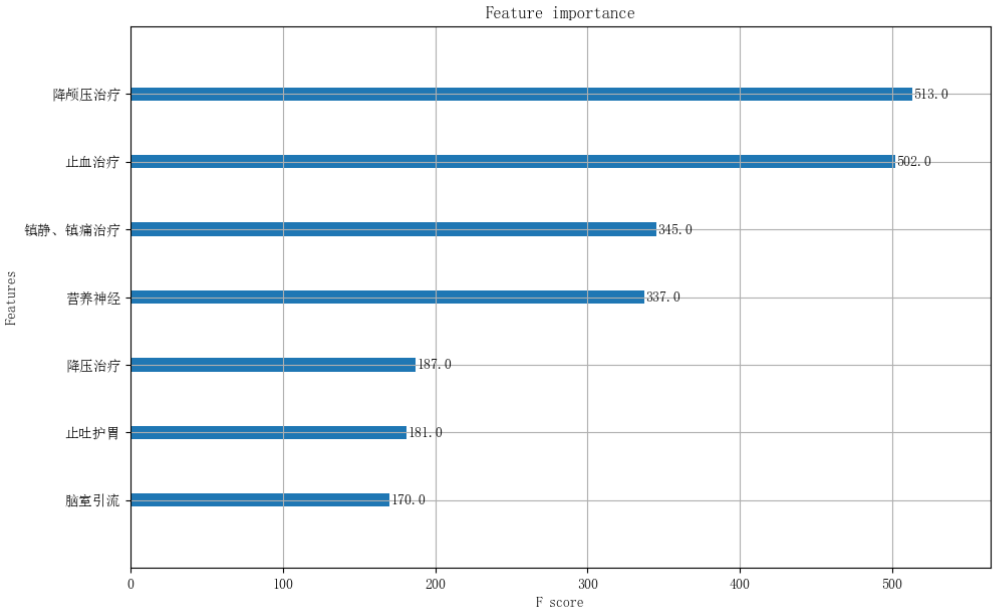

图 5.13 血肿变化特征重要性图

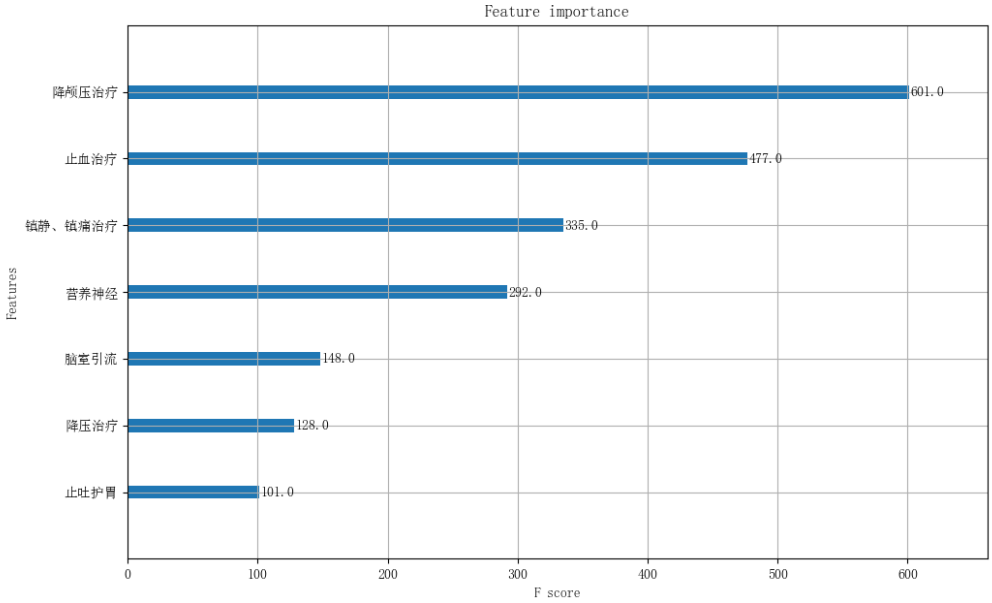

图 5.14 水肿变化特征重要性图

2 斯皮尔曼相关系数

斯皮尔曼相关系数（Spearman's rank correlation coefficient）是一种用于衡量两个变量之间的单调关系的统计量。单调关系指的是，随着一个变量的增加，另一个变量也会以某种趋势增加或减少，尽管可能不是线性的关系。

$$\rho = 1 - \frac{6\sum d^2}{n(n^2-1)} \quad (14)$$

其中：

ρ：斯皮尔曼相关系数；d：排名差；n：数据点的数量。

具体而言，斯皮尔曼相关系数基于将原始数据转换为排序（或秩）的形式进行计算。首先，将每个变量的观察值按照大小进行排序，然后用相应的秩值替代原始观察值。接着，计算排序后的变量之间的皮尔逊相关系数。斯皮尔曼相关系数的取值范围在 -1 到 1 之间：当两个变量的排列顺序完全一致时，斯皮尔曼相关系数为 1，表示存在完全的单调正相关关系。当两个变量的排列顺序完全相反时，斯皮尔曼相关系数为-1，表示存在完全的单调负相关关系。当两个变量之间没有明显的单调关系时，斯皮尔曼相关系数接近于 0。斯皮尔曼相关系数不要求变量之间的关系是线性的，因此它对于非线性关系的发现非常有用。它也相对不受异常值的影响，因为它基于秩而不是原始数值。斯皮尔曼相关系数不要求变量之间的关系是线性的，因此它对于非线性关系的发现非常有用。它也相对不受异常值的影响，因为它基于秩而不是原始数值。

3 斯皮尔曼相关系数计算

针对血肿指标与水肿指标的相关性分析，使用斯皮尔曼相关性分析。对 HM_volume 和 ED_volume，进行相关系数计算，得到结果 0.5274。

5.3 问题 3 模型的建立与求解

5.3.1 构建 mRS 预测模型

1 XGBoost

回归问题，使用 XGBoost 作为模型。对模型进行训练，并画出特征重要性图。

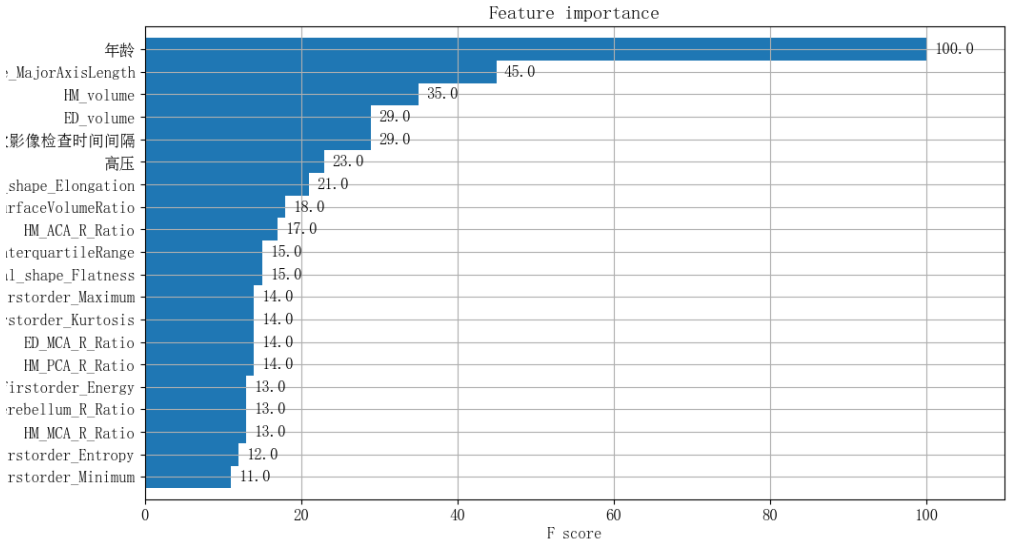

图 5.15 mRS 特征重要性图

可以看到该模型下影响 mRS 的最重要的特征是患者的年龄，图 5.16 是 XGBoost 回归预测结果图。表 5-5 是最终 XGBoost 的性能指标。

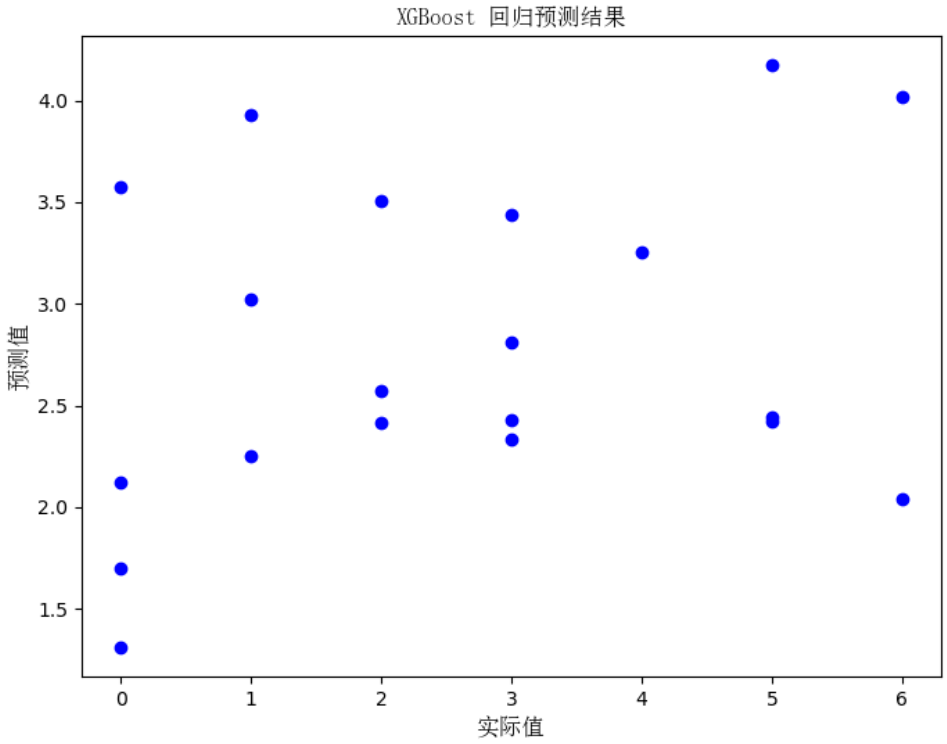

图 5.16 XGBoost 回归预测结果图

表 5-5 XGBoost 性能指标

Model	MSE	MAE	RMSE

XGBoost	3.70	1.60	1.92

2 对建立好的模型进行 mRS 预测并填表。

5.3.2 构建完备 mRS 预测模型

对(a)的训练集和验证集进行扩充，比例不变，加入自变量：第 i 次影像结果，重新训练模型，并对所有含影像检查的患者进行 90 天 mRS 评分，并填表。

5.3.3 患者的预后（90 天 mRS）和个人史、疾病史、治疗方法及影像特征相关性分析

1 皮尔逊相关系数

皮尔逊相关系数（Pearson correlation coefficient）是一种用于度量两个变量之间线性相关程度的统计量。它衡量了两个变量之间的线性关系的强度和方向。这个系数的值介于-1 和 1 之间，其中-1 表示完全的负相关，1 表示完全的正相关，0 表示没有线性相关。

数学上，皮尔逊相关系数可以通过以下公式计算：

$$r = \frac{\sum_{i=1}^{n}(X_i-\bar{X})(Y_i-\bar{Y})}{\sqrt{\sum_{i=1}^{n}(X_i-\bar{X})^2}\sqrt{\sum_{i=1}^{n}(Y_i-\bar{Y})^2}} \quad (14)$$

其中：

r：皮尔逊相关系数；X_i和Y_i：分别表示第i个数据点的两个变量的值；$\bar{X}$和$\bar{Y}$：分别表示X和Y的平均值。

通过计算皮尔逊相关系数，我们可以了解两个变量之间的线性关系。如果r的值接近 1 或-1，表示两个变量之间存在强烈的线性相关性，正值表示正相关，负值表示负相关。如果r的值接近 0，则表示两个变量之间几乎没有线性关系。

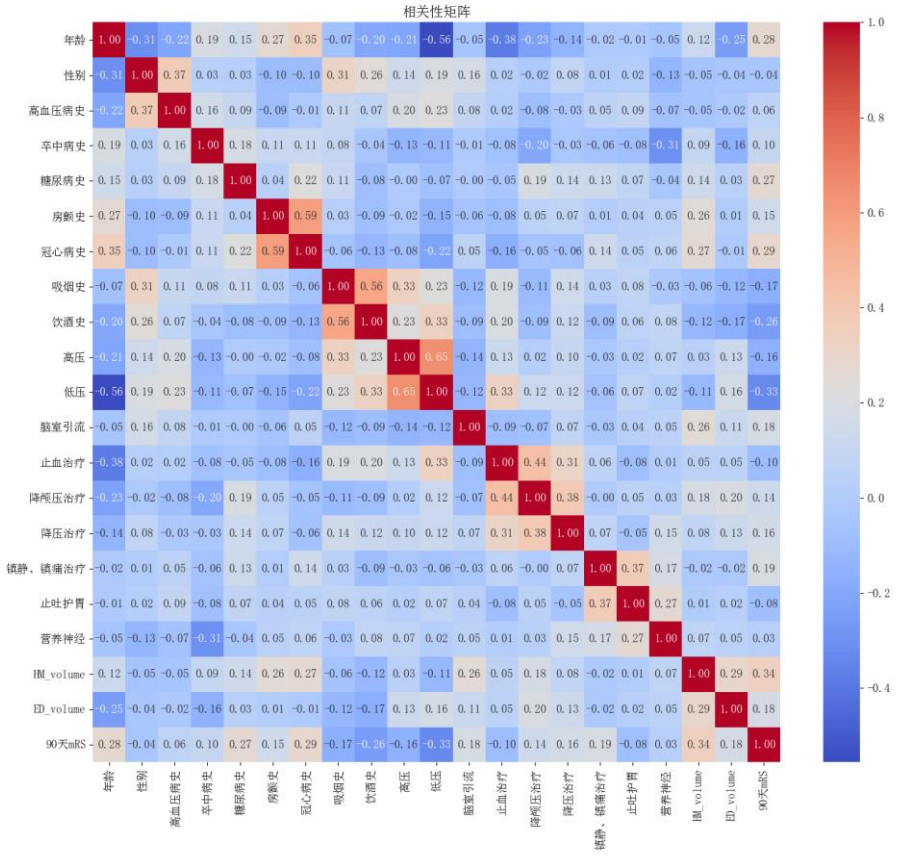

图 5.17 病史、治疗、血水肿体积和 90 天 mRS 相关热力图

这里只读取前 100 位患者的数据，对'年龄'，'性别'，'高血压病史'，'卒中病史'，'糖尿病史'，'房颤史'，'冠心病史'，'吸烟史'，'饮酒史'，'高压'，'低压'，'脑室引流'，'止血治疗'，'降颅压治疗'，'降压治疗'，'镇静、镇痛治疗'，'止吐护胃'，'营养神经'，'HM_volume'，'ED_volume'，'90 天 mRS'这些自变量绘制热力图。通过相关性分析，我们观察到了一些重要的趋势。

首先，90 天 mRS 与 HM_volume 之间呈现出了较高的正相关性，这表明在 90 天的随访期间，患者的神经功能恢复与脑部血肿的体积密切相关。其次，冠心病史与神经功能恢复之间也呈现出一定程度的相关性，这可能反映了心血管健康状况与脑部康复之间的一些关联。此外，年龄和糖尿病史也在一定程度上影响了患者的神经功能恢复，尽管与前两者相比，它们的相关性较弱。这些观察结果为进一步的研究提供了重要线索，也提示了在康复过程中需要关注脑部血肿的体积、心血管状况以及患者的年龄和糖尿病史等因素。特别是，我们观察到，90 天 mRS 与舒张压（低压）之间呈现出了显著的负相关性，表明在康复的 90 天内，较低的舒张压水平可能与患者的神经功能恢复呈现出一定程度的负向关联。这一发现强调了在康复过程中对血压进行有效的管理可能对于促进神经功能康复具有积极的影响。此外，我们还观察到了与饮酒史之间的负相关性，这也提示了在康复阶段避免过量饮酒可能对于神经系统康复至关重要。这些结果强调了在康复过程中综合考虑血压和饮酒史等因素的重要性。

6 模型分析检验

6.1 预测血肿概率模型拟合度分析

我们使用了随机森林模型来预测血肿发生的概率。为了评估模型的拟合度，我们采用了 AUC（曲线下面积）作为指标。AUC 可以在[0, 1]范围内取值，值越接近 1 表示模型的性能越好。我们进行了 5 折交叉验证来稳定地评估模型的性能，得到了平均 AUC 为 0.50。

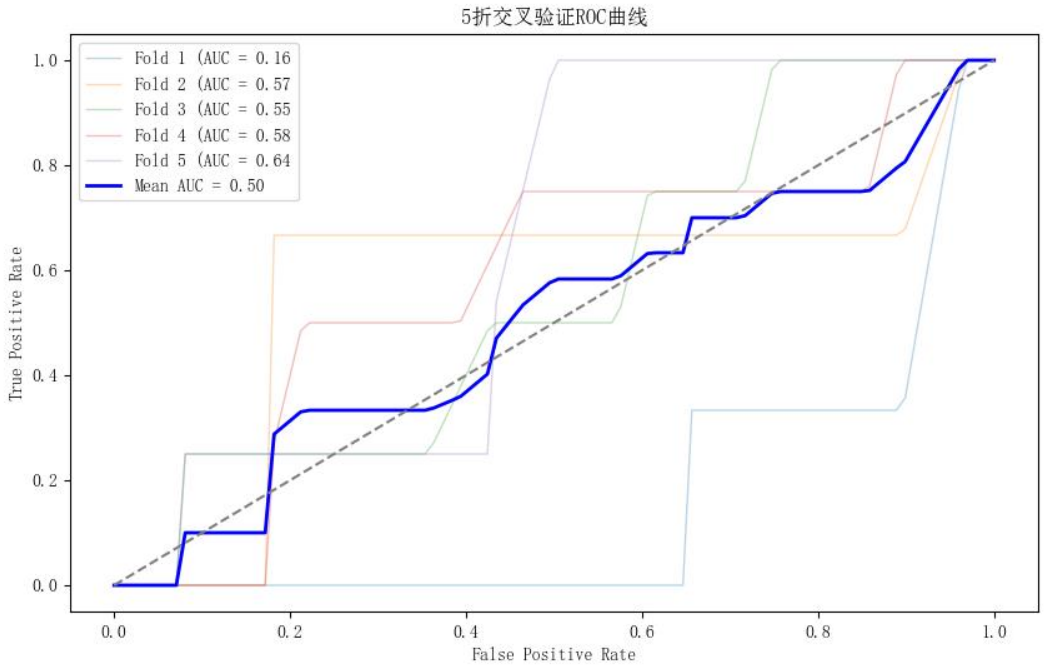

图 6.1 随机森林预测血肿概率模型 ROC 曲线

6.2 预测血肿概率模型拟合度分析

残差是实际观测值与模型预测值之间的差异，其分布特征对模型的稳健性和准确性具有重要影响。我们对模型的残差进行了分布分析，结果显示残差呈现接近正态分布的特征，这表明了模型在训练集上的良好拟合度。而且进行聚类后子模型预估的残差均值较低，方差也低，说明聚类后子模型的性能比原始模型有提升。

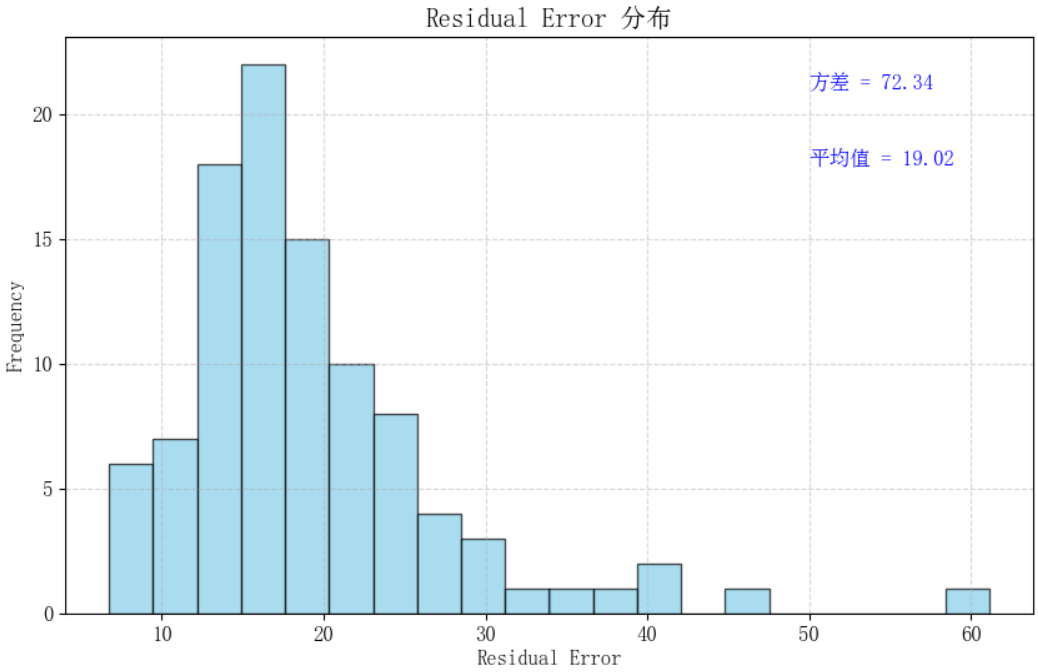

图 6.2 水肿体积随时间进展（全体）残差分布图

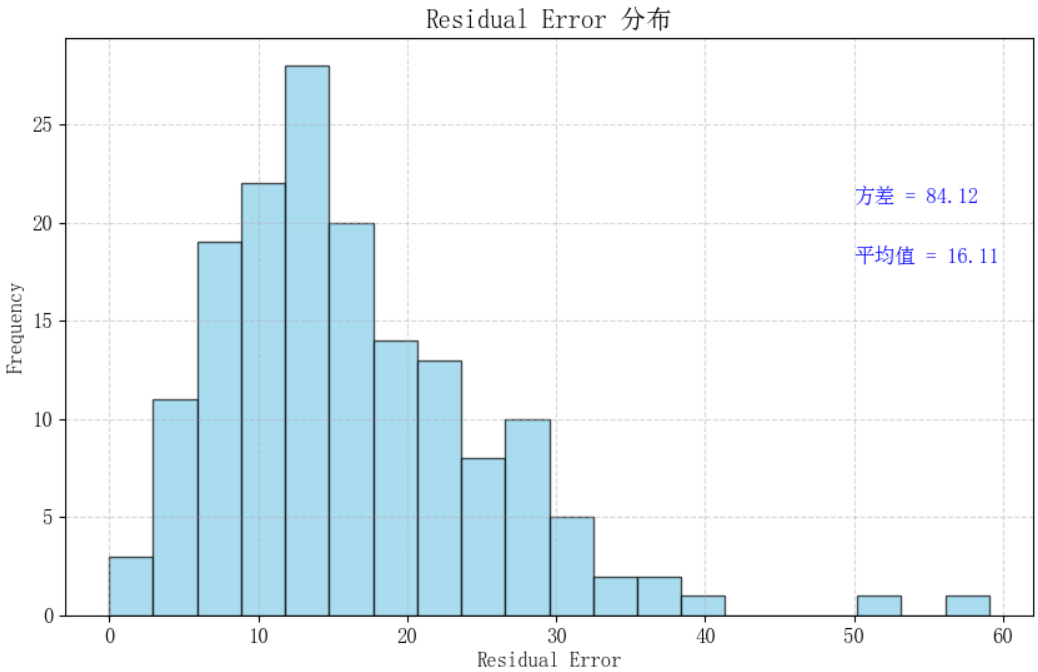

图 6.3 水肿体积随时间进展（亚组）残差分布图

7 模型评价与推广

7.1 模型的优点

(1)模型对于竞赛所给数据尽可能多的都进行了使用，简化了人工筛选数据的过程，使用机器学习这一工具提升模型的性能。

(2)效率高，可以较快的训练和进行快速预测。对于对病人的血肿、水肿及预后恢复情况能进行一个快速的预估，为医护人员可以提前划分出高危人群加以重点监测，增加随访次数。

7.2 模型的缺点

(1)实际医疗中，出血性脑卒中患者往往伴随着其他并存的急性全身性疾病（如感染、脱水、缺氧、高血糖和高血压等）[10]，而本模型在建立时就已经假设没有发生其他疾病。

(2)本文提出的模型对于有准确数据的使用效果较好，由于时间问题没有对其他情况进行检验。但是实际医疗中，想要得知患者准确的发病时间是比较难的，在这种情况下，本模型可能无法达到较好效果。

(3)实际上患者的预后(90 天 mRS)与患者的年龄、疾病史和治疗方法并不一定是线性相关的，本文将其视为线性相关，忽略了边际效应的影响。

7.3 模型的推广

在预测血肿概率的模型上，可以考虑使用 BP 神经网络，来获得较好的性能。

齐欣[11]使用了机器学习加诊断影像用于预测脑出血预后情况，如果想要获得更合理的模型，可以加入计算机视觉的部分来完善。

参考文献

[1] HAUPENTHAL D, SCHWAB S, KURAMATSU J B， Hematoma expansion in intracerebral hemorrhage - the right target? ， Neurological Research and Practice，5(1): 36，2023。

[2] LI Z, YOU M, LONG C, et al.， Hematoma Expansion in Intracerebral Hemorrhage: An Update on Prediction and Treatment ， Frontiers in Neurology， 11，2020。

[3] 范敬争，姜玉艳， 脑出血后血肿周围水肿形成机制的研究进展 ， 山东医药，61(2): 92-4，2021。

[4] CARCEL C, SATO S, ZHENG D, et al.， Prognostic Significance of Hyponatremia in Acute Intracerebral Hemorrhage: Pooled Analysis of the Intensive Blood Pressure Reduction in Acute Cerebral Hemorrhage Trial Studies* ， Critical Care Medicine， 44(7): 1388-94，2016。

[5] SPRüGEL M I, KURAMATSU J B, VOLBERS B, et al.， Perihemorrhagic edema ，Revisiting hematoma volume, location, and surface， 93(12): e1159-e70，2019。

[6] 董红瑶，王弈丹，李丽红， 随机森林优化算法综述 ， 信息与电脑(理论版)，33(17): 34-7，2021。

[7] TIN KAM H， Random decision forests ， Proceedings of 3rd International Conference on Document Analysis and Recognition， 1: 278-82 vol.1，1995。

[8] MILLWOOD I Y, WALTERS R G, MEI X W, et al.， Conventional and genetic evidence on alcohol and vascular disease aetiology: a prospective study of 500 000 men and women in China ， Lancet， 393(10183): 1831-42，2019。

[9] CHEN T, GUESTRIN C， XGBoost: A Scalable Tree Boosting System ， Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining，: 785 - 94，2016。

[10] KERNAN W N, VISCOLI C M, FURIE K L, et al.， Pioglitazone after Ischemic Stroke or Transient Ischemic Attack ， New England Journal of Medicine，374(14): 1321-31，2016。

[11] 齐欣. 基于机器学习的血肿周围组织模型预测脑出血预后的研究 [D]; 大连医科大学，2022.

附录 A 血肿扩张模型代码

1.1 逻辑回归代码

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
import joblib
from sklearn.metrics import precision_score, recall_score, accuracy_score, f1_score, roc_curve, roc_auc_score
import matplotlib.pyplot as plt
from matplotlib.font_manager import FontProperties

plt.rcParams['font.sans-serif'] = ['SimSun']
plt.rcParams['axes.unicode_minus'] = False
font = FontProperties(fname=r"C:\Windows\Fonts\simsun.ttc", size=12)
df1 = pd.read_excel('竞赛发布数据\\表 1-患者列表及临床信息.xlsx')
df1 = df1.loc[:99, '年龄': '营养神经']

df2 = pd.read_excel('匹配结果.xlsx')
df2 = df2.loc[:99, 'HM_volume': 'ED_Cerebellum_L_Ratio']

df3 = pd.read_excel('竞赛发布数据\\表 3-患者影像信息血肿及水肿的形状及灰度分布.xlsx')
df3 = df3.loc[:99, 'original_shape_Elongation': 'NCCT_original_firstorder_Variance']

X = pd.concat([df1, df2, df3], axis=1)

target_data = pd.read_excel('竞赛发布数据\\血肿扩张.xlsx')
y = target_data['血肿扩张']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=888)

logistic_regression = LogisticRegression(random_state=56666)

logistic_regression.fit(X_train, y_train)
print("Logistic Regression Model training completed.")
importances = logistic_regression.coef_[0]
features = X.columns

feature_importance = list(zip(features, importances))
```

```
feature_importance.sort(key=lambda x: abs(x[1]), reverse=False)

features, importances = zip(*feature_importance)


top_features = features[:20]
top_importances = importances[:20]

plt.figure(figsize=(10, 6))
plt.barh(range(len(top_features)), top_importances, align='center')
plt.yticks(range(len(top_features)), top_features)
plt.xlabel('特征重要性')
plt.title('逻辑回归特征重要性 (Top 20)')
plt.show()

model_filename = 'logistic_regression_model.pkl'
joblib.dump(logistic_regression, model_filename)

y_pred = logistic_regression.predict(X_test)

# 计算准确度
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)

# 计算 Precision
precision = precision_score(y_test, y_pred)
print("Precision:", precision)

# 计算 Recall
recall = recall_score(y_test, y_pred)
print("Recall:", recall)

# 计算 F1 分数
f1 = f1_score(y_test, y_pred, average='weighted')
print("F1 Score:", f1)

# 计算 ROC 曲线和 AUC
probabilities = logistic_regression.predict_proba(X_test)[:, 1]

fpr, tpr, _ = roc_curve(y_test, probabilities)
auc = roc_auc_score(y_test, probabilities)

# 绘制 ROC 曲线
```

```
plt.figure(figsize=(8, 6))
plt.plot(fpr, tpr, label=f'ROC 曲线 (AUC = {auc:.4f})', linewidth=2)
plt.plot([0, 1], [0, 1], 'k--', linewidth=2)
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('假正率', fontproperties=font)
plt.ylabel('真正率', fontproperties=font)
plt.title('逻辑回归 ROC 曲线', fontproperties=font)
plt.legend(loc="lower right", prop=font)
plt.show()
```

1.2 随机森林代码

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
import joblib
from sklearn.metrics import precision_score, recall_score, accuracy_score, f1_score, roc_curve, roc_auc_score
import matplotlib.pyplot as plt
from matplotlib.font_manager import FontProperties
from sklearn.metrics import confusion_matrix, classification_report
plt.rcParams['font.sans-serif'] = ['SimSun']
plt.rcParams['axes.unicode_minus'] = False
font = FontProperties(fname=r"C:\Windows\Fonts\simsun.ttc", size=12)
df1 = pd.read_excel('竞赛发布数据\\表 1-患者列表及临床信息.xlsx')
df1 = df1.loc[:99, '年龄': '营养神经']

df2 = pd.read_excel('匹配结果.xlsx')
df2 = df2.loc[:99, 'HM_volume': 'ED_Cerebellum_L_Ratio']

df3 = pd.read_excel('竞赛发布数据\\表 3-患者影像信息血肿及水肿的形状及灰度分布.xlsx')
df3 = df3.loc[:99, 'original_shape_Elongation': 'NCCT_original_firstorder_Variance']

X = pd.concat([df1, df2, df3], axis=1)
target_data = pd.read_excel('竞赛发布数据\\血肿扩张.xlsx')
y = target_data['血肿扩张']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=888)
```

```
26.
27. rf_classifier = RandomForestClassifier(random_state=56666)
28.
29. rf_classifier.fit(X_train, y_train)
30. print("Random Forest Model training completed.")
31. feature_importances = rf_classifier.feature_importances_
32.
33. y_pred = rf_classifier.predict(X_test)
34. importances = rf_classifier.feature_importances_
35. features = X.columns
36. feature_importance = list(zip(features, importances))
37. feature_importance.sort(key=lambda x: x[1], reverse=False)
38. features, importances = zip(*feature_importance)
39.
40. top_features = features[:20]
41. top_importances = importances[:20]
42.
43. plt.figure(figsize=(10, 6))
44. plt.barh(range(len(top_features)), top_importances, align='cente
    r')
45. plt.yticks(range(len(top_features)), top_features)
46. plt.xlabel('特征重要性')
47. plt.title('随机森林特征重要性 (Top 20)')
48. plt.show()
49. # 计算准确度
50. accuracy = accuracy_score(y_test, y_pred)
51. print("Accuracy:", accuracy)
52.
53. # 计算 Precision
54. precision = precision_score(y_test, y_pred)
55. print("Precision:", precision)
56.
57. # 计算 Recall
58. recall = recall_score(y_test, y_pred)
59. print("Recall:", recall)
60.
61. # 计算 F1 分数
62. f1 = f1_score(y_test, y_pred,average='weighted')
63. print("F1 Score:", f1)
64. # 计算 ROC 曲线和 AUC
65. probabilities = rf_classifier.predict_proba(X_test)[:, 1]
66. fpr, tpr, _ = roc_curve(y_test, probabilities)
67. auc = roc_auc_score(y_test, probabilities)
68.
```

```
# 绘制 ROC 曲线
plt.figure(figsize=(8, 6))
plt.plot(fpr, tpr, label=f'ROC 曲线 (AUC = {auc:.4f})', linewidth=2)
plt.plot([0, 1], [0, 1], 'k--', linewidth=2)
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('假正率', fontproperties=font)
plt.ylabel('真正率', fontproperties=font)
plt.title('随机森林 ROC 曲线', fontproperties=font)
plt.legend(loc="lower right", prop=font)
plt.show()
conf_matrix = confusion_matrix(y_test, y_pred)
print("Confusion Matrix:")
print(conf_matrix)

class_report = classification_report(y_test, y_pred)
print("Classification Report:")
print(class_report)
import seaborn as sns

plt.figure(figsize=(8, 6))
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues',
            xticklabels=['Predicted 0', 'Predicted 1'],
            yticklabels=['Actual 0', 'Actual 1'])
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix')
plt.show()

model_filename = 'random_forest_model.pkl'
joblib.dump(rf_classifier, model_filename)
```

1.3 随机森林五折验证代码

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
import joblib
from sklearn.metrics import precision_score, recall_score, accuracy_score, f1_score, roc_curve, roc_auc_score
import matplotlib.pyplot as plt
from matplotlib.font_manager import FontProperties
```

```
from sklearn.metrics import confusion_matrix, classification_report
plt.rcParams['font.sans-serif'] = ['SimSun']
plt.rcParams['axes.unicode_minus'] = False
font = FontProperties(fname=r"C:\Windows\Fonts\simsun.ttc", size=12)
df1 = pd.read_excel('竞赛发布数据\\表 1-患者列表及临床信息.xlsx')
df1 = df1.loc[:99, '年龄': '营养神经']

df2 = pd.read_excel('匹配结果.xlsx')
df2 = df2.loc[:99, 'HM_volume': 'ED_Cerebellum_L_Ratio']

df3 = pd.read_excel('竞赛发布数据\\表 3-患者影像信息血肿及水肿的形状及灰度分布.xlsx')
df3 = df3.loc[:99, 'original_shape_Elongation': 'NCCT_original_firstorder_Variance']

X = pd.concat([df1, df2, df3], axis=1)

target_data = pd.read_excel('竞赛发布数据\\血肿扩张.xlsx')
y = target_data['血肿扩张']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=888)

rf_classifier = RandomForestClassifier(random_state=56666)

rf_classifier.fit(X_train, y_train)
print("Random Forest Model training completed.")
feature_importances = rf_classifier.feature_importances_

y_pred = rf_classifier.predict(X_test)
importances = rf_classifier.feature_importances_
features = X.columns

feature_importance = list(zip(features, importances))
feature_importance.sort(key=lambda x: x[1], reverse=False)
features, importances = zip(*feature_importance)

top_features = features[:20]
top_importances = importances[:20]

plt.figure(figsize=(10, 6))
```

```
plt.barh(range(len(top_features)), top_importances, align='center')
plt.yticks(range(len(top_features)), top_features)
plt.xlabel('特征重要性')
plt.title('随机森林特征重要性 (Top 20)')
plt.show()
# 计算准确度
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)

# 计算 Precision
precision = precision_score(y_test, y_pred)
print("Precision:", precision)

# 计算 Recall
recall = recall_score(y_test, y_pred)
print("Recall:", recall)

# 计算 F1 分数
f1 = f1_score(y_test, y_pred,average='weighted')
print("F1 Score:", f1)
# 计算 ROC 曲线和 AUC
probabilities = rf_classifier.predict_proba(X_test)[:, 1]
fpr, tpr, _ = roc_curve(y_test, probabilities)
auc = roc_auc_score(y_test, probabilities)

# 绘制 ROC 曲线
plt.figure(figsize=(8, 6))
plt.plot(fpr, tpr, label=f'ROC 曲线 (AUC = {auc:.4f})', linewidth=2)
plt.plot([0, 1], [0, 1], 'k--', linewidth=2)
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('假正率', fontproperties=font)
plt.ylabel('真正率', fontproperties=font)
plt.title('随机森林 ROC 曲线', fontproperties=font)
plt.legend(loc="lower right", prop=font)
plt.show()
conf_matrix = confusion_matrix(y_test, y_pred)
print("Confusion Matrix:")
print(conf_matrix)

class_report = classification_report(y_test, y_pred)
print("Classification Report:")
```

```
print(class_report)
import seaborn as sns

plt.figure(figsize=(8, 6))
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues',
                        xticklabels=['Predicted 0', 'Predicted 1'],
                        yticklabels=['Actual 0', 'Actual 1'])
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix')
plt.show()

model_filename = 'random_forest_model.pkl'
joblib.dump(rf_classifier, model_filename)
```

附录 B 水肿体积随时间变化模型代码

2.1 水肿体积高斯拟合代码

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from scipy.optimize import curve_fit
plt.rcParams['font.sans-serif'] = ['SimSun']
plt.rcParams['axes.unicode_minus'] = False
plt.rcParams['font.size'] = 12
df = pd.read_excel('竞赛发布数据\\数据.xlsx')
UniqueNums = df.iloc[1:, 0].unique()

RecheckTime = []
EdemaData = []
DiagnosisInfo = []

for i in range(len(UniqueNums)):
        a = np.where(df.iloc[:, 0] == UniqueNums[i])
        DiagnosisInfo.append(df.iloc[a[0][0], 3:].values.astype(float))
        RecheckTime.extend(df.iloc[a[0], 2].values.astype(float))
        EdemaData.extend(df.iloc[a[0], 33].values.astype(float))


```

```
def gaussian_func(x, a, b, c):
        return a * np.exp(-((x - b) / c)**2)

params, covariance = curve_fit(gaussian_func, RecheckTime, EdemaData, p0=[1, 1000, 100], maxfev=5000)

x_fit = np.linspace(1, 5000, 1000)
y_fit = gaussian_func(x_fit, *params)
#  计算残差
EdemaFit = gaussian_func(np.array(RecheckTime), *params)
ErrorA = []

for i in range(len(UniqueNums)):
        a = np.where(df.iloc[:, 0] == UniqueNums[i])[0] - 1

        error = np.mean(np.abs(np.array(EdemaFit)[a] - np.array(EdemaData)[a])) * 0.001
        ErrorA.append(error)

#  创建包含残差的 DataFrame
residual_df = pd.DataFrame({'UniqueNums': UniqueNums, 'ResidualError': ErrorA})

#  保存 DataFrame 到 Excel 文件
residual_df.to_excel('残差数据.xlsx', index=False)

#  绘制散点图
plt.scatter(RecheckTime, EdemaData, marker='+')
plt.plot(x_fit, y_fit, label='Gaussian Fit')
plt.xlabel('时间')
plt.ylabel('水肿/10^-3ml')
plt.legend()
plt.show()
```

2.2 年龄聚类代码

```
import pandas as pd
from sklearn.cluster import KMeans

data = pd.read_excel('竞赛发布数据\\表 1-患者列表及临床信息.xlsx')

X = data[['年龄']]

kmeans = KMeans(n_clusters=3)

```

```
kmeans.fit(X)

data['聚类结果'] = kmeans.labels_

for cluster_label in range(3):
        cluster_data = data[data['聚类结果'] == cluster_label]
        cluster_data.to_excel(f'cluster_{cluster_label}.xlsx', index=False)
```

2.3 聚类分配数据代码

```
import pandas as pd

cluster_0 = pd.read_excel('cluster_0.xlsx')
cluster_1 = pd.read_excel('cluster_1.xlsx')
cluster_2 = pd.read_excel('cluster_2.xlsx')


original_data = pd.read_excel('竞赛发布数据\\数据.xlsx')

cluster_0_data = original_data[original_data['ID'].isin(cluster_0['ID'])]
cluster_1_data = original_data[original_data['ID'].isin(cluster_1['ID'])]
cluster_2_data = original_data[original_data['ID'].isin(cluster_2['ID'])]

cluster_0_data.to_excel('cluster_0_data.xlsx', index=False)
cluster_1_data.to_excel('cluster_1_data.xlsx', index=False)
cluster_2_data.to_excel('cluster_2_data.xlsx', index=False)
```

2.4 聚类后拟合代码

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from scipy.optimize import curve_fit
plt.rcParams['font.sans-serif'] = ['SimSun']
plt.rcParams['axes.unicode_minus'] = False
plt.rcParams['font.size'] = 12
df = pd.read_excel('cluster_0_data.xlsx')

UniqueNums = df.iloc[1:, 0].unique()

RecheckTime = []
EdemaData = []
DiagnosisInfo = []
```

```
15.
16. for i in range(len(UniqueNums)):
17.         a = np.where(df.iloc[:, 0] == UniqueNums[i])
18.         DiagnosisInfo.append(df.iloc[a[0][0], 3:].values.astype(f
    loat))
19.         RecheckTime.extend(df.iloc[a[0], 2].values.astype(float))

20.         EdemaData.extend(df.iloc[a[0], 33].values.astype(float))

21.
22.
23. def gaussian_func(x, a, b, c):
24.         return a * np.exp(-((x - b) / c)**2)
25.
26. params, covariance = curve_fit(gaussian_func, RecheckTime, Ed
    emaData, p0=[1, 1000, 100], maxfev=5000)
27.
28. x_fit = np.linspace(1, 8000, 1000)
29. y_fit = gaussian_func(x_fit, *params)
30. EdemaFit = gaussian_func(np.array(RecheckTime), *params)
31. ErrorA = []
32.
33. for i in range(len(UniqueNums)):
34.         a = np.where(df.iloc[:, 0] == UniqueNums[i])[0] - 1

35.         error = np.mean(np.abs(np.array(EdemaFit)[a] - np.arra
    y(EdemaData)[a])) * 0.001
36.         ErrorA.append(error)
37.
38. residual_df = pd.DataFrame({'UniqueNums': UniqueNums, 'Residua
    lError': ErrorA})
39.
40. residual_df.to_excel('残差数据聚类0.xlsx', index=False)
41.
42. plt.scatter(RecheckTime, EdemaData, marker='+')
43. plt.plot(x_fit, y_fit, label='Gaussian Fit')
44. plt.xlabel('时间')
45. plt.ylabel('水肿/10^-3ml')
46. plt.legend()
47. plt.show()
```

2.5 判断显著性代码

```
1. import pandas as pd
2. from scipy.stats import f_oneway
3.
```

```
data = pd.read_excel('竞赛发布数据\\2a.xlsx')
data = data.loc[:99]
independent = pd.read_excel('竞赛发布数据\\表 1-患者列表及临床信息.xlsx')
independent = independent.loc[:99, '脑室引流': '营养神经']

X = independent[['营养神经']]
y = data['ED_volume1'] - data['ED_volume0']    # 计算随访 1 与随访 0 之间的差值

# 执行方差分析
grouped_data = [y[X['营养神经'] == 0], y[X['营养神经'] == 1]]
f_statistic, p_value = f_oneway(*grouped_data)

print(f'F-statistic: {f_statistic}')
print(f'p-value: {p_value}')

alpha = 0.05
if p_value < alpha:
        print('在显著性水平 0.05 下，存在组间均值差异。')
else:
        print('在显著性水平 0.05 下，不存在组间均值差异。')
```

2.6 多层感知器代码

```
import torch
import torch.nn as nn
import torch.optim as optim
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import mean_absolute_error, r2_score, explained_variance_score, max_error

data = pd.read_excel('竞赛发布数据\\2a.xlsx')
data = data.loc[:99]
independent = pd.read_excel('竞赛发布数据\\表 1-患者列表及临床信息.xlsx')
independent = independent.loc[:99, '脑室引流': '营养神经']

X = independent[['止血治疗']]
y = data['ED_volume4'] - data['ED_volume2']    # 计算随访 1 与随访 0 之间的差值

```

```
scaler = StandardScaler()
X = scaler.fit_transform(X)

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

X_train = torch.FloatTensor(X_train)
y_train = torch.FloatTensor(y_train.values.reshape(-1, 1))
X_test = torch.FloatTensor(X_test)
y_test = torch.FloatTensor(y_test.values.reshape(-1, 1))

class RegressionModel(nn.Module):
        def __init__(self):
                super(RegressionModel, self).__init__()
                self.fc = nn.Sequential(
                        nn.Linear(1, 64),
                        nn.BatchNorm1d(64),
                        nn.ReLU(),
                        nn.Linear(64, 32),
                        nn.BatchNorm1d(32),
                        nn.ReLU(),
                        nn.Linear(32, 1)
                )

        def forward(self, x):
                return self.fc(x)


model = RegressionModel()
criterion = nn.MSELoss()
optimizer = optim.Adam(model.parameters(), lr=0.001)

epochs = 50000
for epoch in range(epochs):
        optimizer.zero_grad()
        output = model(X_train)
        loss = criterion(output, y_train)
        loss.backward()
        optimizer.step()

with torch.no_grad():
        y_pred = model(X_test)
        y_pred = y_pred.squeeze().numpy()
        y_true = y_test.squeeze().numpy()
```

```
61.
62. mae = mean_absolute_error(y_true, y_pred)
63. r2 = r2_score(y_true, y_pred)
64. explained_var = explained_variance_score(y_true, y_pred)
65. max_err = max_error(y_true, y_pred)
66.
67. print(f'平均绝对误差 (MAE): {mae}')
68. print(f'R-squared (R2): {r2}')
69. print(f'解释方差得分: {explained_var}')
70. print(f'最大误差: {max_err}')
```

2.7 水肿变化特征重要性代码

```
1. import pandas as pd
2. import xgboost as xgb
3. from xgboost import plot_importance
4. import matplotlib.pyplot as plt
5. plt.rcParams['font.sans-serif'] = ['SimSun']
6. plt.rcParams['axes.unicode_minus'] = False
7. merged_data = pd.read_excel('merged_treatments.xlsx')
8.
9. ed_volume_data = pd.read_excel('竞赛发布数据\\2a.xlsx')
10.
11. merged_data = pd.merge(merged_data, ed_volume_data[['ID', 'ED_volume0', 'ED_volume1']], on='ID', how='inner')
12. merged_data.fillna(0, inplace=True)
13. merged_data['Delta_ED_volume'] = merged_data['ED_volume1'] - merged_data['ED_volume0']
14. X = merged_data[['脑室引流', '止血治疗', '降颅压治疗', '降压治疗', '镇静、镇痛治疗', '止吐护胃', '营养神经']]
15. y = merged_data['Delta_ED_volume']
16. model = xgb.XGBRegressor()
17. model.fit(X, y)
18.
19. plot_importance(model)
20. plt.show()
```

2.8 斯皮尔曼相关性代码

```
1. import pandas as pd
2.
3. data = pd.read_excel('竞赛发布数据\\数据.xlsx')

4. data.fillna(0, inplace=True)
5.
6. hm_volume = data['HM_volume']
7. ed_volume = data['ED_volume']
8.
```

```
correlation = hm_volume.corr(ed_volume, method='spearman')

print(f'HM_volume 和 ED_volume 的相关系数为: {correlation:.4f}')
```

附录 C mRS 预测模型代码

3.1 XGBoost mRS 预测模型

```
import pandas as pd
from sklearn.model_selection import train_test_split
import xgboost as xgb
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error, mean_absolute_error
import matplotlib.pyplot as plt
import numpy as np
import joblib
from matplotlib.font_manager import FontProperties
plt.rcParams['font.sans-serif'] = ['SimSun']
plt.rcParams['axes.unicode_minus'] = False
plt.rcParams['font.size'] = 12
font = FontProperties(fname=r"C:\Windows\Fonts\simsun.ttc", size=12)
df1 = pd.read_excel('竞赛发布数据\\表 1-患者列表及临床信息.xlsx')
df1 = df1.loc[:99, '年龄':'营养神经']

df2 = pd.read_excel('竞赛发布数据\\表 2-患者影像信息血肿及水肿的体积及位置.xlsx')
df2 = df2.loc[:99, 'HM_volume':'ED_Cerebellum_L_Ratio']

df3 = pd.read_excel('竞赛发布数据\\表 3-患者影像信息血肿及水肿的形状及灰度分布.xlsx')
df3 = df3.loc[:99, 'original_shape_Elongation':'NCCT_original_firstorder_Variance']

df = pd.concat([df1, df2, df3], axis=1)

target_data = pd.read_excel('竞赛发布数据\\表 1-患者列表及临床信息.xlsx')
df['90 天 mRS'] = target_data['90 天 mRS']

y = df['90 天 mRS']
X = df.drop(columns=['90 天 mRS'])

```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=66)

xgb_regressor = xgb.XGBRegressor(n_estimators=50,random_state=42)
xgb_regressor.fit(X_train, y_train)
print("XGBoost Regressor training completed.")
xgb.plot_importance(xgb_regressor,height=1,max_num_features=20)

model_filename = 'xgboost3a_model.pkl'
joblib.dump(xgb_regressor, model_filename)
y_pred_xgb = xgb_regressor.predict(X_test)

mse_xgb = mean_squared_error(y_test, y_pred_xgb)
print("Mean Squared Error (XGBoost):", mse_xgb)


mae_xgb = mean_absolute_error(y_test, y_pred_xgb)
print("Mean Absolute Error (XGBoost):", mae_xgb)


rmse_xgb = np.sqrt(mse_xgb)
print("Root Mean Squared Error (XGBoost):", rmse_xgb)



plt.figure(figsize=(8, 6))
plt.scatter(y_test, y_pred_xgb, color='blue')
plt.xlabel('实际值', fontproperties=font)
plt.ylabel('预测值', fontproperties=font)
plt.title('XGBoost 回归预测结果', fontproperties=font)
plt.show()
```

3.2 XGBoost mRS 预测模型（全）

```
mport pandas as pd
from sklearn.model_selection import train_test_split
import xgboost as xgb
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error, mean_absolute_error
import matplotlib.pyplot as plt
import numpy as np
import joblib
from matplotlib.font_manager import FontProperties
plt.rcParams['font.sans-serif'] = ['SimSun']
```

```
plt.rcParams['axes.unicode_minus'] = False
plt.rcParams['font.size'] = 12
font = FontProperties(fname=r"C:\Windows\Fonts\simsun.ttc", size=12)
df1 = pd.read_excel('竞赛发布数据\\表 1-患者列表及临床信息.xlsx')
df1 = df1.loc[:159, '年龄': '营养神经']

df2 = pd.read_excel('竞赛发布数据\\表 2-患者影像信息血肿及水肿的体积及位置.xlsx')
df2 = df2.loc[:159, 'HM_volume': 'ED_Cerebellum_L_Ratio']

df3 = pd.read_excel('竞赛发布数据\\表 3-患者影像信息血肿及水肿的形状及灰度分布.xlsx')
df3 = df3.loc[:576, 'original_shape_Elongation': 'NCCT_original_firstorder_Variance']

df = pd.concat([df1, df2, df3], axis=1)

target_data = pd.read_excel('竞赛发布数据\\表 1-患者列表及临床信息.xlsx')
df['90 天 mRS'] = target_data['90 天 mRS']

y = df['90 天 mRS']
X = df.drop(columns=['90 天 mRS'])

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=66)

xgb_regressor = xgb.XGBRegressor(n_estimators=50,random_state=42)

xgb_regressor.fit(X_train, y_train)
print("XGBoost Regressor training completed.")
xgb.plot_importance(xgb_regressor,height=1,max_num_features=20)

model_filename = 'xgboost3a_model.pkl'
joblib.dump(xgb_regressor, model_filename)
y_pred_xgb = xgb_regressor.predict(X_test)

# 计算均方误差（MSE）
mse_xgb = mean_squared_error(y_test, y_pred_xgb)
print("Mean Squared Error (XGBoost):", mse_xgb)

```

```
# 计算平均绝对误差（MAE）
mae_xgb = mean_absolute_error(y_test, y_pred_xgb)
print("Mean Absolute Error (XGBoost):", mae_xgb)

# 计算均方根误差（RMSE）
rmse_xgb = np.sqrt(mse_xgb)
print("Root Mean Squared Error (XGBoost):", rmse_xgb)
# 可视化预测结果
plt.figure(figsize=(8, 6))
plt.scatter(y_test, y_pred_xgb, color='blue')
plt.xlabel('实际值', fontproperties=font)
plt.ylabel('预测值', fontproperties=font)
plt.title('XGBoost 回归预测结果', fontproperties=font)
plt.show()
```

3.3 皮尔逊相关性分析

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
plt.rcParams['font.sans-serif'] = ['SimSun']
plt.rcParams['axes.unicode_minus'] = False
plt.rcParams['font.size'] = 12
data_df = pd.read_excel('竞赛发布数据\\数据.xlsx', nrows=448)

columns_of_interest = [
        '年龄', '性别', '高血压病史', '卒中病史', '糖尿病史', '房颤史', '冠心病史', '吸烟史', '饮酒史',
        '高压', '低压', '脑室引流', '止血治疗', '降颅压治疗', '降压治疗', '镇静、镇痛治疗', '止吐护胃', '营养神经',
        'HM_volume', 'ED_volume', '90 天 mRS'
]

correlation_matrix = data_df[columns_of_interest].corr(method='spearman')

plt.figure(figsize=(16, 14))
sns.heatmap(correlation_matrix, annot=True, fmt=".2f", cmap="coolwarm")
plt.title('相关性矩阵')
```

目录

```
20.plt.show()
```