中国研究生创新实践系列大赛
“华为杯”第十八届中国研究生数学建模竞赛

学　　校	天津大学
参赛队号	**21100560019**
队员姓名	**1.孙明阳**
	2.闫杰
	3.赵博

中国研究生创新实践系列大赛
“华为杯”第十八届中国研究生数学建模竞赛

题 目 抗乳腺癌候选药物的优化建模

摘 要：

乳腺癌药物研究更趋数据化和智能化，这对国家稳定发展、生物科技创新、人民健康安全等诸多方面有着巨大的影响。如何在乳腺癌药物研制过程中保证化合物安全性的同时，获得更高的生物活性，已经成为了乳腺癌药物研发领域的一大挑战。本文采用皮尔森相关分析、随机森林、支持向量回归、公共空间模式和 MLP 等算法，实现了对数据的深度挖掘，分别建立了化合物的分子描述符同生物活性和 ADMET 性质间的数量关系，具体做法如下：

针对问题 1，首先，分别计算 729 组分子描述符数据中含有 0 值的比率，并设置 0 值比率阈值，对 0 值比率低于阈值的分子描述符予以剔除；其次，分别计算剩余数据中各分子描述符与生物活性值之间的皮尔森相关系数 PLCC，剔除 PLCC 值低于 0.3 的分子描述符；最后，在经过前两步处理的剩余数据的基础上，构建随机森林模型，计算剩余分子描述符重要程度并进行排序，取其中重要性排名前 20 的分子描述符作为最终筛选结果，以供后续研究。

针对问题 2，根据问题要求，基于问题 1 筛选的特征构建了一个基于支持向量回归的组合 K 近邻、决策树和梯度提升的回归模型，利用组合的回归模型，充分获取多种有效信息指导最终的预测，首先特征输入到 K 近邻、决策树和梯度提升三个回归模型中得到三个中间预测分数，将三个分数融合后送入最终的支持向量回归模型中得到最终的分数，将给定的训练集重新按照 8:2 划分为新的训练集和测试集后，模型的 RMSE 能够达到 0.525，验证了模型的有效性。

针对问题 3，首先，对原始数据进行数据筛选，将 729 组分子描述符数据中值全 0 的分子描述符剔除；然后，分别设计了基于机器学习和深度学习的两种分类模型。对于机器学习的分类模型，采用 CSP 算法对分子描述符数据进行特征提取，该算法可以将不同类别的特征差异最大化，而后将特征送入 RBF 核函数的 SVM 进行分类预测。为了探索更优性能，本文基于深度学习设计了第二种分类模型，该模型利用自注意力机制对数据特征进行全局加权，而后基于多尺度一维卷积对特征进一步融合，最后利用 MLP 网络进行分类预测。在 ADMET 测试集上的结果表明，本文设计的两个分类模型均取得较好的性能。然而，通过分析模型发现，原始的数据中正负样本比例失衡，该情况可能对模型的分类性能造成影响。针对这一问题，本文提出了对应的改进手段备后续研究。

针对问题 4，首先分析满足 ADMET 性质的最优组合为[1,0,0,1,0]，接着分两步进行分析，（1）在问题 2 回归模型的基础上采用 20 个分子描述符作为输入数据建立初始回归模型，经过回归预测后记录生物活性最好的前三种化合物。（2）在问题 3 分类模型的基础上对 20 个分子描述符的测试集进行分类预测，得到上述三种化合物对应的 ADMET 性质，并判断是否满足有三个及以上性质较好，如果满足，则记录该化合物的分子描述符取值情况。如果不

满足，则根据问题 1 得到的前 20 个分子描述符的重要程度得分高低，剔除得分最低的一个，此时剩下 19 个分子描述符，重复上述过程。考虑到分子描述符剔除到一定程度后会严重影响回归和分类模型的性能，因此规定剔除的数量到 10 个为止，此时得到的所有化合物的分子描述符的取值即为最终结果。

关键词：随机森林；支持向量回归；CSP 算法；神经网络；自注意力

目录

1. 问题重述

1.1 问题背景

乳腺癌是危害女性生命健康的常见疾病之一[1]，也是目前世界上致死率最高的癌症之一。2018 年全球癌症统计结果表明，全球共计约 1810 万新发癌症病例，其中乳腺癌占比 11.6%；新发癌症死亡病例 960 万例，其中乳腺癌死亡病例占 6.6%，在女性癌症死亡病例和新发病例中均高居首位。近年来，我国乳腺癌的发病率和死亡率均位居女性恶性肿瘤患者的首位，且发病人群呈现年轻化趋势，形势严峻。

目前乳腺癌的临床治疗方法主要包括手术切除、内分泌综合治疗、放射和化学疗法以及光动力学疗法等。然而，乳腺癌具有较高的转移和复发特性，而放化疗可能引起多药耐药性和诸多不良反应，其治疗效果仍无法满足临床需求。近年来，药物疗法在乳腺癌的治疗方面取得一定突破，成为了乳腺癌术后辅助疗法之一，在减少放化疗副作用、提高人体免疫力和降低术后复发风险等方面具有独特优势。

近些年，在药物研发过程中，通常会针对乳腺癌相关的某个靶标设计生成化合物。为了评估化合物对标靶的拮抗作用，需要监测一些重要的指标供制药师参考，在这些指标中，生物活性被广泛用于评估化合物对标靶的拮抗作用，反映化合物的治疗效果。化合物的自身特性通常用分子结构描述符表示，在实际药物生产中，构建化合物的定量结构-活性关系模型，对预测更好生物活性的新化合物，或指导已有化合物的优化均具有重要意义。

一种化合物若想成为候选药物，在拥有良好生物活性的同时，还需确保自身的安全性。临床药物研究中，通常使用 ADMET 性质作为评估化合物安全性的指标。ADMET 性质对于乳腺癌药物研发具有重要意义，一个化合物的生物活性指标再好，若其不具备较好的 ADMET 性质，则依然难以成为临床用药。若能改善相关技术，探索出符合安全性要求的 ADMET 指标与化合物自身结构特性之间的关系，则可以导向性地对相关化合物合成操作进行调整，从而进一步提升药物安全性。

综上所述，利用数据挖掘技术，分析化合物的分子描述符特性，设计即时学习框架，进而建立模型预测，对有效提升化合物生物活性，保障化合物临床应用安全性具有重要意义。以上研究内容将为乳腺癌临床药物研发的高效运行提供理论支持。

1.2 需要解决的问题

问题 1. 数据预处理：根据题目中提供的分子描述符数据和生物活性数据，从 1974 个化合物的 729 个分子描述符中筛选出前 20 个对生物活性影响程度最高的分子描述符，供下面研究使用。

问题 2. 建立生物活性预测模型：在问题 1 筛选出的 20 个分子描述符的基础上，构建化合物对ER_α生物活性的定量预测模型，即为回归预测问题。在问题 1 筛选特征的基础上，构建回归预测模型实现对测试集中给定化合物的生物活性预测，同时可利用 RMSE 和 SROCC 等指标对构建的模型进行有效性验证

问题 3. 建立 ADMET 性质分类预测模型：利用题目中提供的分子描述符数

据和 1974 个化合物的 ADMET 数据，分别构建化合物的 Caco-2、CYP3A4、hERG、HOB、MN 的分类预测模型。并利用 5 个分类预测模型，预测文件“ADMET.xlsx” test 表中 50 个化合物的 ADMET 数据，将结果填入 ADMET.xlsx”的 test 表中对应的 Caco-2、CYP3A4、hERG、HOB、MN 列。

问题 4. 探究化合物的哪些分子描述符在满足什么条件或什么取值范围内能够使得化合物对抑制 Erα 的生物活性具有更好的效果，同时具有三个及以上较好的 ADMET 性质。

2. 模型的假设

1. 1974 个化合物的 729 个分子描述符、对 ERα 的生物活性值和 ADMET 性质的数据均是在同一条件下得到，互相之间不存在其他影响。

2. 上述数据真实可靠，不需再行验证可直接应用到数学模型中。

3. 符号说明

符号	意义
ρ	皮尔森相关系数
zero_per	分子描述符 0 值比率
FIM	特征重要性指数
GI	基尼指数
ζ	支持向量回归距离残差
RBF	径向基函数
SVR	支持向量回归
RMSE	均方根误差
SROCC	斯皮尔曼秩相关系数
SVM	支持向量机
CSP	公共空间模式
MLP	多层感知机

4. 问题一分析与求解

4.1 问题一分析

根据问题一的要求，对于某种化合物，其生物活性的高低同时受到 729 个分子描述符的影响。然而，不同的分子描述符对生物活性的影响各不相同，本题需从“Molecular_Descriptor.xlsx”中的原始数据筛选出 20 个对生物活性影响程度最高的分子描述符。总体的分子描述符筛选方法如下：

（1）对于题目给定的 729 种分子描述符，分别计算其数据中取值为 0 的概率，将含 0 率较高的分子描述符剔除；

（2）剔除剩余分子描述符中与生物活性相关性较低的分子描述符；

（3）在经过（1），（2）两步筛选后剩余数据的基础上，建立随机森林回归模型，对不同分子描述符的重要性进行判断，并将重要性最高的 20 个分子描述符作为最终结果。

问题一数据处理的流程图如图 4.1 所示。首先计算各个分子描述符的数据中含 0 值数据的比例，将含 0 率高于某个阈值的分子描述符数据视为不良数据，并将不良数据剔除，得到数据集 1。而后分别计算数据集 1 中各分子描述符数据与生物活性值 pIC50 之间的相关性，并从中筛选出相关性较高的分子描述符数据，得到数据集 2。最后在数据集 2 的基础上，建立随机森林回归模型并计算出不同分子描述符的重要性，从数据集 2 的所有分子描述符中筛选出重要性排序前 20 的分子描述符，整理得到最终的数据处理结果。

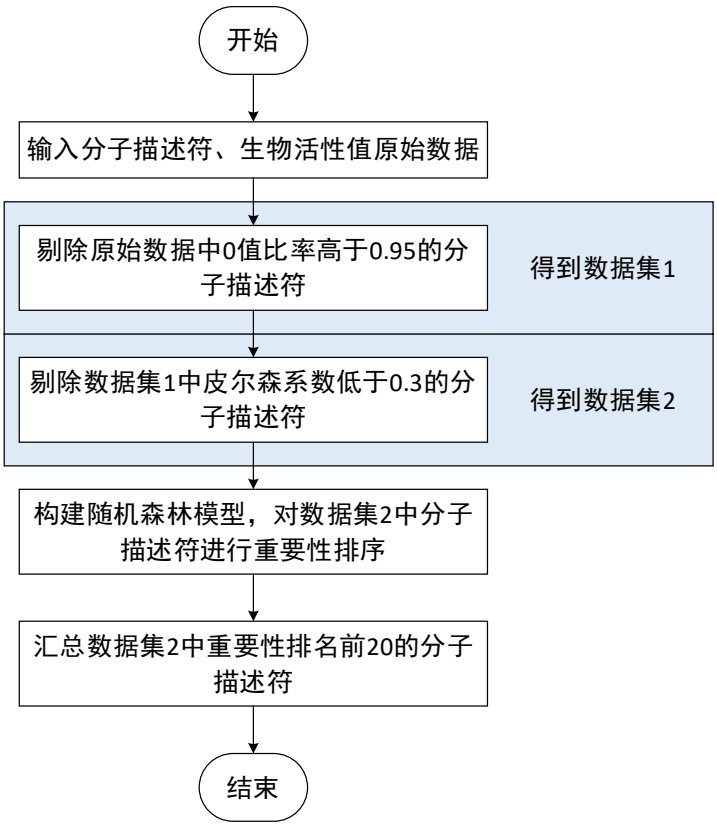

图 4.1 问题一思路流程图

4.2 数据处理

4.2.1 高 0 值比率分子描述符的剔除

通过对文件“Molecular_Descriptor.xlsx”中的数据进行观察，可以看出一部分分子描述符包含大量取值为 0 的数据。而通过对不同化合物的生物活性值进行观察分析可知，不同化合物的生物活性值呈现多样性分布。例如，对于分子描述符 nI，不同化合物的 nI 取值几乎均为 0，然而不同化合物的生物活性值呈现着多样性分布，这表明，不同化合物的生物活性值变化与 nI 取值之间关联性较小。基于上述分析，可将原始数据中含 0 元素较多的分子描述符视作异常数据予以剔除。具体做法如下，定义第 i 个分子描述符的 0 值比率 *zero_per* 为：

$$zero\_per = \frac{第i个分子描述符中数据为0的个数}{第i个分子描述符的数据总量} \tag{4.1}$$

为了剔除所有分子描述符中 0 值比率高的分子描述符，需要设定阈值进行筛选。若存在：

$$zero\_per > 0.95 \tag{4.2}$$

则认为该组分子描述符数据的重要性较低，需对其进行剔除。

按照上述原则编写 Matlab 程序对原始数据进行遍历检验，剔除各分子描述符中的 0 值比率较高的分子描述符，处理结果如表 4.1 所示。

表 4.1 高 0 值比率分子描述符剔除结果

操作类别	分子描述符
剔除部分数据	nP nI … nT12Ring
总计	涉及 326 个分子描述符

原始数据中共包含 729 个分子描述符，因此，经过高 0 值比率分子描述符剔除后，剩余 403 个分子描述符构成新的数据集 1，供进一步筛选使用。为了阶段性展示分子描述符与生物活性值之间的关系，我们从各个阶段的数据集中随机选择 36 种分子描述符，通过绘制每种分子描述符数据与生物活性值数据之间的散点图，反映二者间的关系。图 4.2 展示了数据集 1 中 36 种分子描述符与生物活性值之间的关系。分析图 4.2 中的可视化结果可知，数据集 1 中各分子描述符的数据多数分布在非 0 范围内，证明数据集 1 相比于原始数据集，其数据的有效性得到增强。

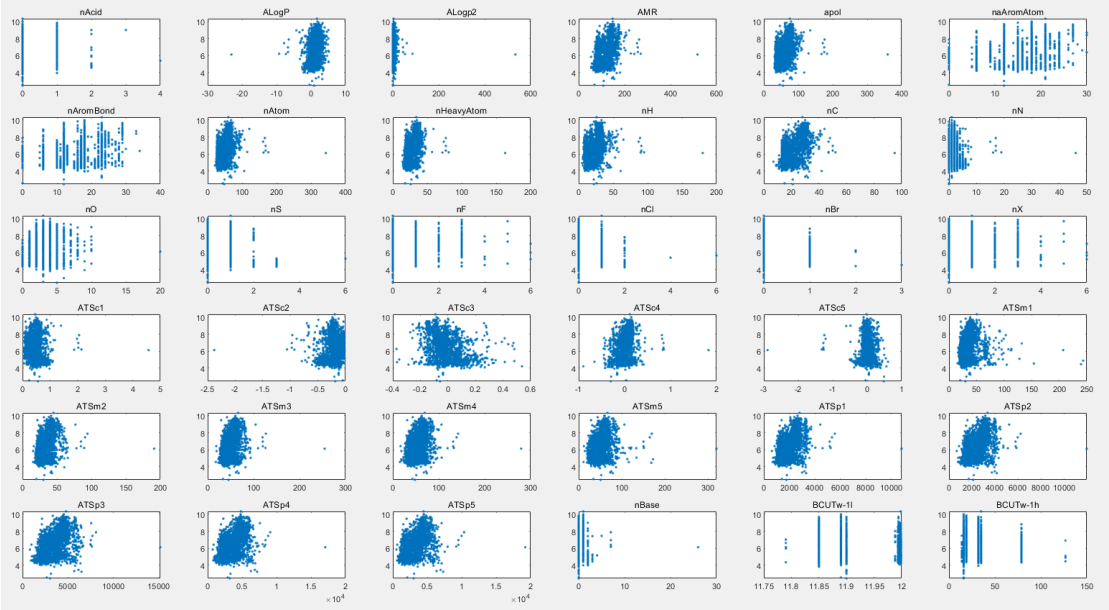

图 4.2 数据集 1 中分子描述符与生物活性值关系可视化

4.2.2 低相关性分子描述符的剔除

剔除高 0 值比率数据可以增强数据的有效性，有效地避免 0 值数据对实验结果的干扰。但仅剔除高 0 值比率数据仍不能满足题目需求，需在此基础上完成进一步筛选。通过分析图 4.2 可知，新数据集 1 中各分子描述符与生物活性值之间的数据相关性呈现多样性，例如，分子描述符 ATSc3 与生物活性值的数据分布之间大致呈现负相关的关系，而 ALogp2 与生物活性值之间并未呈现明显的相关关系。本题要求筛选出对生物活性值影响程度高的分子描述符，就与生物活性值相关性高的分子描述符而言，当这些分子描述符的数据在某个范围内出现变化时，生物活性值也会随之产生较为明显的变化。反观与生物活性值相关性较低的分子描述符，分子描述符的变化对生物活性值的影响并不显著。因此，分子描述符与生物活性值之间的相关性高低可以作为筛选高影响度分子描述符的一个有效标准。

此部分数据筛选工作需根据数据集 1 中各分子描述符与生物活性值之间的相关性对数据集 1 中的分子描述符进行更深层次的筛选，为此，我们拟采用皮尔森相关系数（PLCC）作为衡量分子描述符与生物活性值间相关性高低的标准。

1）皮尔森相关系数

皮尔森相关系数也称皮尔森积矩相关系数（Pearson product-moment correlation coefficient），是一种线性相关系数，用于反映两个变量 X 和 Y 之间的线性相关程度，其计算公式如下：

$$\rho_{X,Y}=\frac{cov(X,Y)}{\sigma_X\sigma_Y}=\frac{\sum_{i=1}^{n}(X_i-\bar{X})(Y_i-\bar{Y})}{\sqrt{\sum_{i=1}^{n}(X_i-\bar{X})^2}\sqrt{\sum_{i=1}^{n}(Y_i-\bar{Y})^2}} \tag{4.3}$$

皮尔森相关系数反映了两个变量的线性相关性的强弱程度，ρ的绝对值越大说明 X 与 Y 之间的相关性越强。通常情况下，通过以下取值范围判断 X,Y 之间的相关强度：

ρ的绝对值处于 0.8-1.0，表示 X，Y 之间极强相关；
ρ的绝对值处于 0.6-0.8，表示 X，Y 之间强相关；
ρ的绝对值处于 0.4-0.6，表示 X，Y 之间中等强度相关；
ρ的绝对值处于 0.2-0.4，表示 X，Y 之间弱相关；
ρ的绝对值处于 0.0-0.2，表示 X，Y 之间极弱相关或无相关；

2）分子描述符筛选

首先向 Matlab 中导入数据集 1 与 1974 种化合物的生物活性数据，分别计算数据集 1 中各分子描述符数据与生物活性值数据之间的线性相关系数 PLCC。定义F_i为数据集 1 中第 i 个分子描述符数据，S为文件“ERα_activity.xlsx”中的全部生物活性值数据，则数据集 1 中第 i 个分子描述符的 PLCC 值ρ_i计算如下：

$$\rho_i = PLCC(F_i, S) \tag{4.4}$$

编写 Matlab 程序循环遍历数据集 1 中各分子描述符，计算各分子描述符数据与生物活性值之间的线性相关系数，并取其绝对值进行统计。由于数据集 1 中共包含 403 组分子描述符数据，因此经过循环计算可得到 403 组 PLCC 值。为了更加直观地从 403 组分子描述符中筛选出与生物活性线性相关度较高的分子描述符，我们选择统计 403 组 PLCC 值中分布于不同范围的数据个数，在此基础上绘制频率分布直方图。403 组 PLCC 值的频率分布直方图结果如图 4.3 所示。

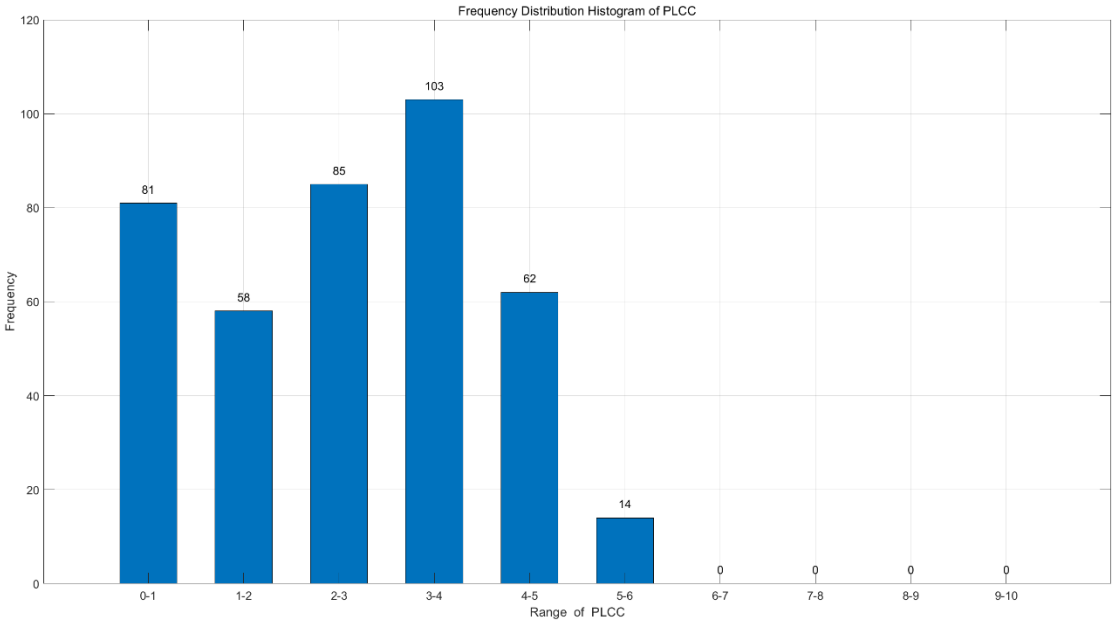

图 4.3 数据集 1 中 PLCC 频率分布直方图

通过对图 4.3 中的直方图进行分析不难看出，数据集 1 中各分子描述符的 PLCC 值仅分布于 0 到 0.6 之间，分布于 0.3-0.4 之间的 PLCC 值最多。此外，通过观察直方图中的数据分布，可以看出，以 PLCC 值为 3 作为分界线，可以大致将 403 组数据进行等比例划分。综合考虑直方图中数据分布情况，以及 1）中提到的 PLCC 值与线性相关度的对应关系，本题选择使用 0.3 作为线性相关性筛选

的阈值。对于数据集 1 中 PLCC 值低于 0.3 的分子描述符进行剔除，同时保留 PLCC 值高于 0.3 的分子描述符，构成新数据集 2。低相关性分子描述符筛选的结果如表 4.2 所示。

表 4.2 低相关性分子描述符剔除结果

操作类别	分子描述符
剔除部分数据	nAcid ALogP … WPATH
总计	涉及 224 个分子描述符

如前所述，数据集 1 中共包含 403 个分子描述符，经过低相关性分子描述符剔除后，剩余的 179 个分子描述符构成新数据集 2。图 4.4 展示了数据集 2 中部分分子描述符与生物活性值之间的关系。对比分析图 4.4 与图 4.3 中的散点图，可以得出以下结论：

（1）与数据集 1 相比，数据集 2 中分子描述符的 0 值比率进一步降低，这表明剔除低相关性分子描述符可以进一步优化数据分布，筛选出的分子描述符的有效性进一步增强。

（2）对比数据集 1 可以看出，数据集 2 中分子描述符与生物活性值的线性相关性更高，如图 4.4 中，ATSc3，BCUTp-1h 和 C1SP2 等多个分子描述符与生物活性间均呈现一定的线性相关关系，这表明筛选出的分子描述符对生物活性值的线性影响程度更高。

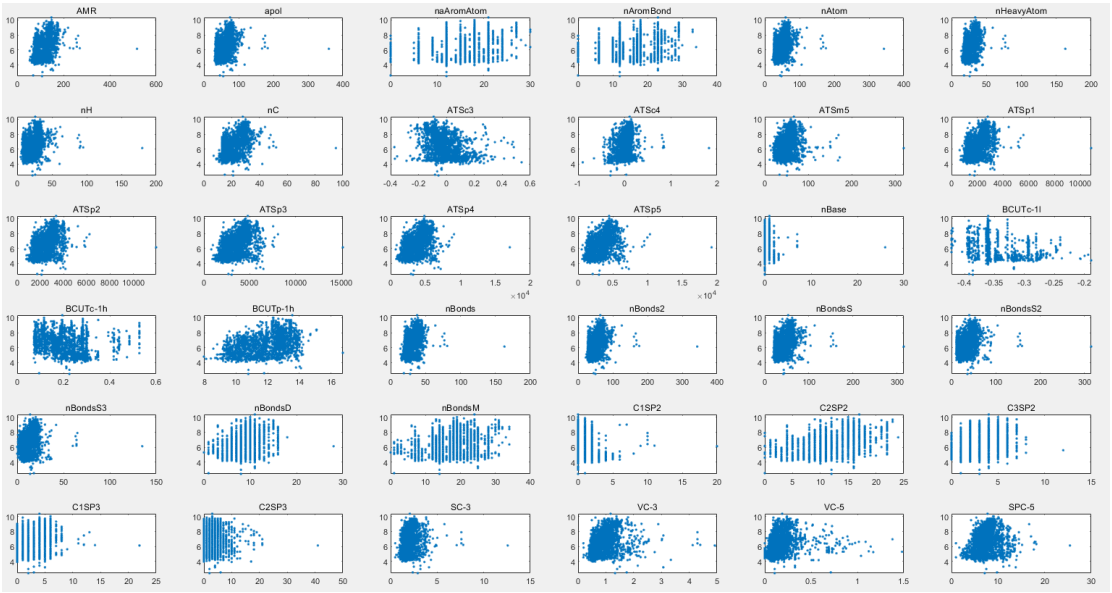

图 4.4 数据集 2 中分子描述符与生物活性关系可视化

4.2.3 基于随机森林的分子描述符重要性排序

剔除分子描述符中与生物活性相关度较低的数据可以剔除数据集 1 中对生物活性影响较小的分子描述符。然而，分子描述符与生物活性间的线性相关度是评估分子描述符对生物活性影响度的最直观的标准，但却并非唯一标准。因此，以线性相关系数 PLCC 作为标准的筛选，对于分子描述符重要性筛选而言只是初步筛选的过程，需在此基础上设计更加有效合理的算法对数据集 2 中的分子描述符进行更深层次的筛选。为此，本题拟采用随机森林（RF）算法完成对分子描述符的最终筛选过程。

1）随机森林算法简介

随机森林是以决策树为基学习器的集成学习算法。随机森林结构简单，易于实现，计算开销小，同时在解决分类和回归问题上表现出了十分惊人的性能，因此，随机森林也被誉为“代表集成学习技术水平的方法”。

随机森林有如下几个特点：

（1）随机森林在分类与回归问题上可以取得极好的准确率；

（2）能够有效地在大规模数据集上运行；

（3）具有处理高维特征的能力；

（4）能够评估各个特征在分类或回归问题上的重要性；

（5）在生成过程中，可以得到内部生成误差的无偏估计；

（6）对于缺省值问题也能得到较好的结果。

图 4.5 直观地展现了随机森林算法，整个随机森林算法可以用以下几个步骤加以概括：

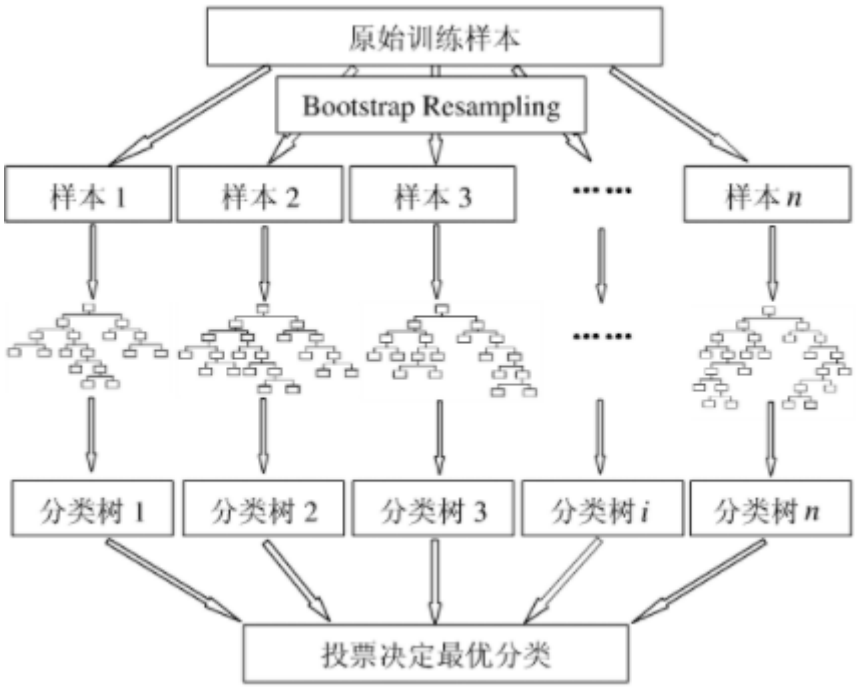

图 4.5 随机森林算法示意图[2]

（1）采用有放回抽样（bootstrap）的方法从样本集中选取 n 个样本作为一个训练集；

（2）用抽样得到的样本集生成一棵决策树。在生成的每一个结点上随机不重复地选择 d 个特征，利用 d 个特征分别对样本集进行划分，找到最佳的划分特征；

（3）重复步骤（1）到步骤（2）共 k 次，k 为随机森林中决策树的个数；

（4）用训练得到的随机森林对测试样本进行预测，并用票选法确定预测结果。

2）RF 特征重要性评估原理

本题采用随机森林作为筛选高重要性分子描述符的算法，究其原因，在于随机森林算法可以对输入特征进行重要性排序。因此，本小节对随机森林评估特征重要性的原理加以分析说明。

用随机森林进行特征重要性评估，其主要思想在于，首先计算每个输入特征在随机森林中每颗决策树上的贡献值，而后对于某个具体的特征，取其对所有决策树贡献值的平均值，作为该特征的贡献值，最后根据各个输入特征的贡献值大小，完成对特征重要性的排序。对于特征的贡献值，通常用基尼指数（Gini Index）或袋外数据（OOB）错误率作为评价指标进行衡量。这里以基尼指数为例，对特征重要性评估的过程加以介绍。

定义特征重要性评分（feature importance measures）为FIM，并将基尼指数用GI表示。假定有m个特征X_1，X_2，X_3，…，X_c，需要计算出每个特征X_j的基尼指数FIM_j^{Gini}，即第 j 个特征在 RF 所有决策树中节点分裂不纯度的平均改变量。基尼指数的计算公式如下：

$$GI_m = 1 - \sum_{k=1}^{|K|} p_{mk}^2 \tag{4.5}$$

其中，K 表示共有 K 个类别，p_{mk}表示节点 m 中类别 k 所占的比例。特征X_j在节点 m 的重要性，即节点 m 分枝前后的基尼指数变化量为：

$$FIM_{jm}^{Gini} = GI_m - GI_l - GI_r \tag{4.6}$$

其中，GI_l和GI_r分别表示分枝后两个新节点的基尼指数。若特征X_j在决策树 i 中出现的节点为集合 M，则X_j在第 i 棵树的重要性为：

$$FIM_{ij}^{Gini} = \sum_{m \in M} FIM_{jm}^{Gini} \tag{4.7}$$

若 RF 中共有 n 棵树，则X_j在整个 RF 中的基尼指数为：

$$FIM_j^{Gini} = \sum_{i=1}^{n} FIM_{ij}^{Gini} \tag{4.8}$$

最后，将各个特征的重要性评分进行归一化处理：

$$FIM_j = \frac{FIM_j^{Gini}}{\sum_{i=1}^{c} FIM_i^{Gini}} \quad (4.9)$$

3）分子描述符重要性排序

第 1 步：导入数据集和标签

首先向 Matlab 中循环读入经过线性相关度筛选的新数据集 2 和生物活性值数据。数据集 2 中包含 1974 种化合物，每种化合物包含 179 种分子描述符。文件“ERα_activity.xlsx”中包含 1974 种化合物各自的生物活性值。因此，将 1974 种化合物视为训练样本，179 种分子描述符视为特征，并将 1974 种化合物对应的生物活性值 pIC_{50} 作为标签。

第 2 步：确定随机森林参数

在建立随机森林模型前，需要先完成随机森林的参数寻优工作。对于随机森林而言，待寻优参数包括叶子节点数和树数。对于叶子节点数，本题拟从 5，10，20，50，100，200，500 中选择最优的叶子节点数。对于随机森林中树的个数，寻优的范围设定为 1 到 2000。将第 1 步中训练样本分别送入叶子节点数不同的随机森林进行训练，并计算各个随机森林的预测值与真实生物活性值之间的均方误差。此外，为了防止最优结果受到随机干扰，上述过程重复进行 5 次。

图 4.6 展示了随机森林的 5 次参数寻优结果。分析 5 次结果不难看出，相同条件下，当叶子节点数为 5 时均方误差最小，因此最优叶子节点数为 5。当树数大于 200 时，均方误差几乎不再下降，因此最优树数为 200。

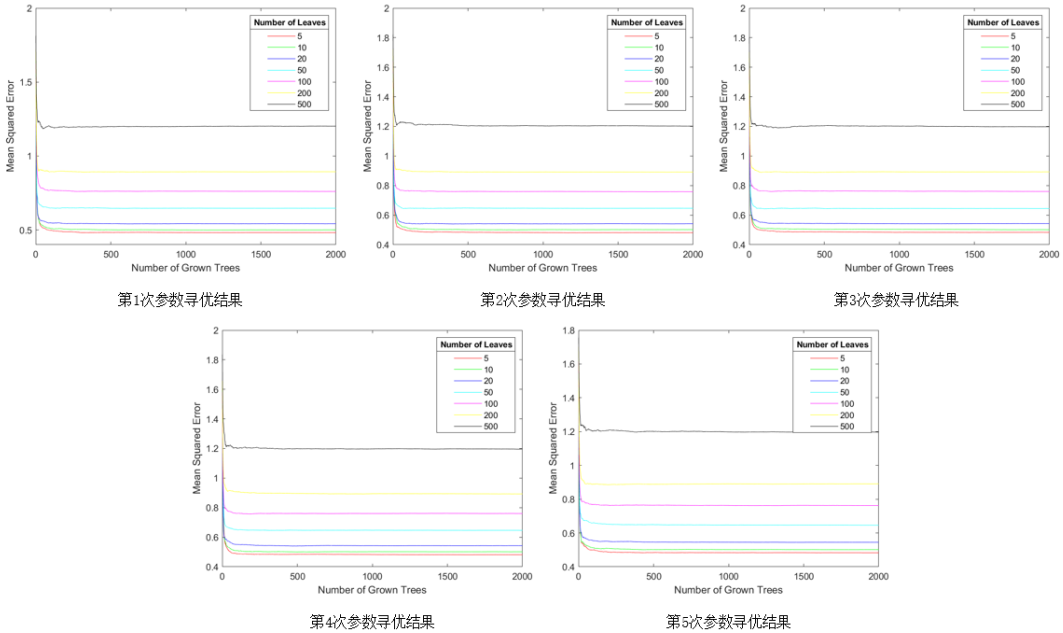

图 4.6 随机森林参数寻优结果

第 3 步：划分训练集和测试集

对第 1 步中构建的数据集根据化合物按照训练集：测试集=4：1 的比例进行划分，为后续随机森林的训练测试准备数据。值得一提的是，随机森林可以用 OOB 误差衡量自身性能，因此一般无需划分训练集和测试集。但由于本题筛选出的 20

个分子描述符会作为训练数据参与问题二的求解。因此，本题设计的随机森林模型仍然划分训练集和测试集，以便筛选出的 20 个分子描述符更加适用于问题二的求解。

第 4 步：构建随机森林模型，完成特征重要程度排序

根据第 2 步中确定的最优参数，搭建随机森林模型，并导入第 3 步中的训练集和测试集依次进行训练测试，最终在测试集上 RMSE 值为 0.7138，这表明随机森林在分子描述符到生物活性的回归问题上取得较好的性能。在此基础上，利用随机森林可以评估特征重要性的功能，对随机森林输入的特征，亦即分子描述符按照重要性进行排序，结果如图 4.7 所示。

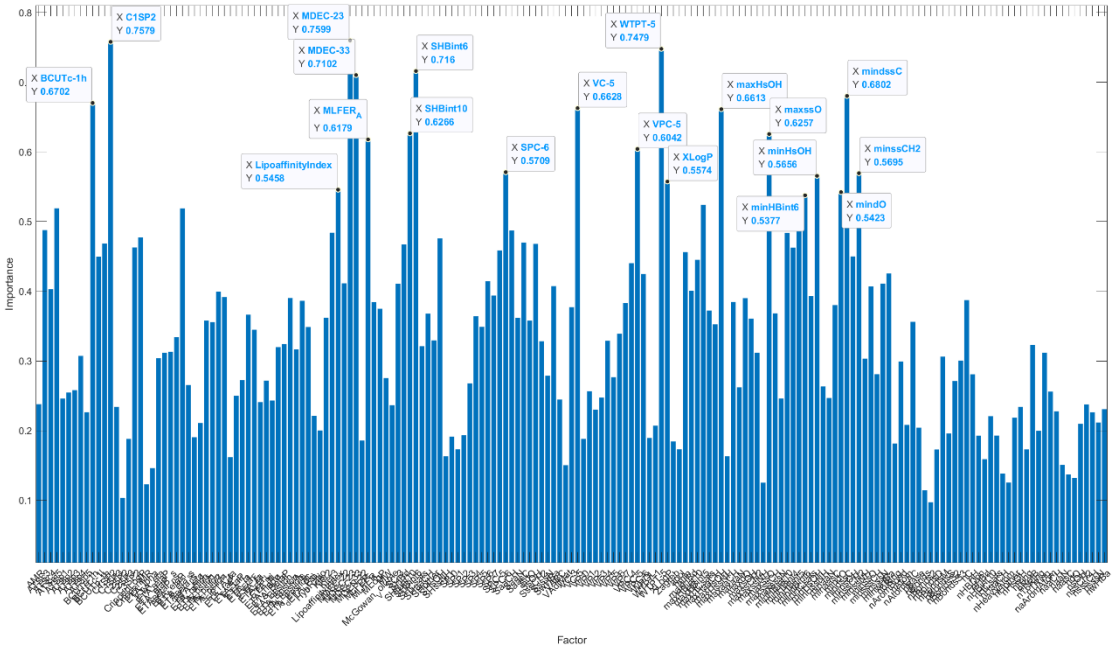

图 4.7 数据集 2 中各分子描述符的重要性排序结果

图 4.7 对数据集 2 中重要性排在前 20 的分子描述符进行标注，为了更加清晰直观地展示最终筛选出的 20 种分子描述符，我们在表 4.3 中对 20 种重要性最高的分子描述符进行集中展示，并在图 4.8 中对 20 个分子描述符与生物活性值之间的关系进行可视化。

4.3 小结与讨论

（1）本题的数据来源为乳腺癌治疗化合物研究的历史数据，涉及文件“Molecular_Descriptor.xlsx”和“ERα_activity.xlsx”中的分子描述符数据和对应的生物活性值数据。

（2）从分子描述符数据 0 值比率、与生物活性值的线性相关程度和随机森林重要性排序三个方面进行数据筛选，最终确定 20 个对生物活性值影响最大的分子描述符。

表 4.3　20 种重要性最高的分子描述符汇总结果

分子描述符	RF 重要性指标	分子描述符	RF 重要性指标
MDEC-23	0.7599	maxssO	0.6257
C1SP2	0.7579	MLFER_A	0.6179
WTPT-5	0.7479	VPC-5	0.6042
SHBint6	0.716	SPC-6	0.5709
MDEC-33	0.7102	minssCH2	0.5695
mindssC	0.6802	minHsOH	0.5656
BCUTc-1h	0.6702	XLogP	0.5574
VC-5	0.6628	LipoaffinityIndex	0.5458
maxHsOH	0.6613	mindO	0.5423
SHBint10	0.6266	minHBint6	0.5377

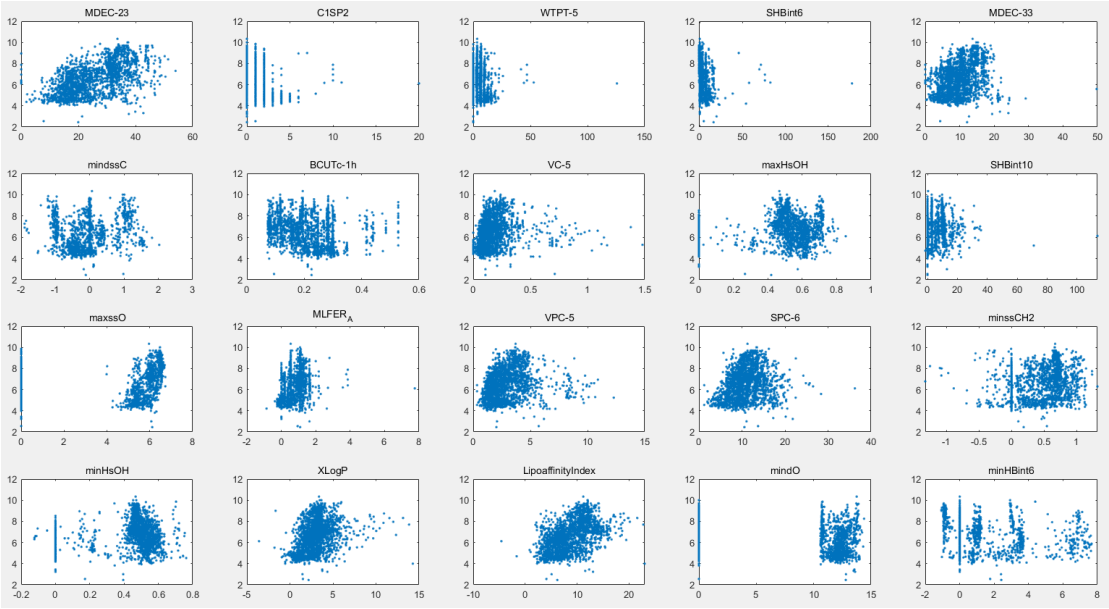

图 4.8 20 个分子描述符与生物活性关系可视化

5. 问题二分析与求解

5.1 问题分析

经过问题一对 729 个分子描述符进行的特征重要性筛选，本题采用上述 20 个对生物活性影响最大的分子描述符作为特征，构建回归模型实现从分子描述符到化合物对 ERα 的生物活性值的预测。

回归问题的本质是从一组或多组数据入手，寻找与目标数据之间的映射关系，即建立模型估计目标数据。随着机器学习的快速发展，涌现出了诸多回归算法，诸如线性回归算法[3]，k 近邻回归算法[4]，岭回归[5]，决策树回归[6]，梯度提升回归[7]，Lasso 回归[8]和支持向量回归（SVR）[9]等等，由于深度学习越来越被各个领域深入研究，基于卷积神经网络的回归算法也逐渐得到应用[10]，然而深度学习算法取得准确效果的前提需要大量数据的训练，对于小数据集问题则不太适用。目前针对回归问题的研究多数采用单一的回归算法同时进行预测，最后选择最优的模型，对于分布较为简单的数据而言，此种方法可以取得很好的效果，当数据分布较为分散时难以达到令人满意的效果。实际上每一种回归算法都有其适应数据结构的规律属性，能够充分利用多种回归算法发提取的信息会是预测结果更加逼近真实结果，因此针对于本问题我们构建了一个基于 SVR 的组合 K 近邻、决策树和梯度提升的回归模型。

SVR 算法与线性回归算法相比目的更为宽松，即线性回归需要所有数据均拟合在线性函数上，若落在线性函数之外的则记为损失，此种算法针对非线性分布的数据而言难以拟合，而 SVR 算法相当于在线性函数两侧添加辅助线，只要落在辅助线内的数据均算做成功拟合，如图 5.1 所示，绿色圆为拟合数据，橙色圆为未拟合数据。

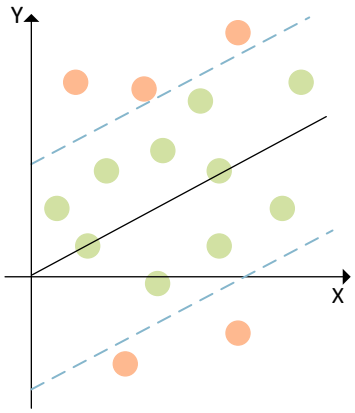

图 5.1 SVR 示意图

SVR 的目标函数简化来讲即为找到一个超平面，使得给定的数据到此超平面距离最短，单纯考虑线性情况时，如图 5.2 所示，定义一个新的概念：残差 ζ，定义虚线内的点残差为 0，虚线外的点到虚线的距离为 ζ，所以，目的即为使得所有 ζ 最小。

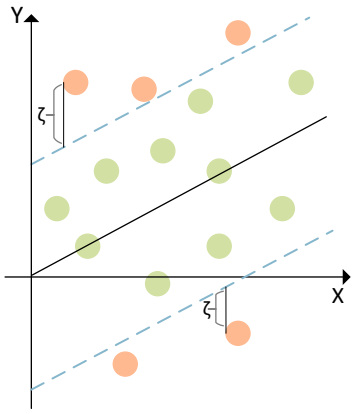

图 5.2 SVR 目标超平面图

对于非线性数据而言，则需要使用核函数将其映射到线性特征空间，再做相同的最小化 ζ 处理，如图 5.3 所示

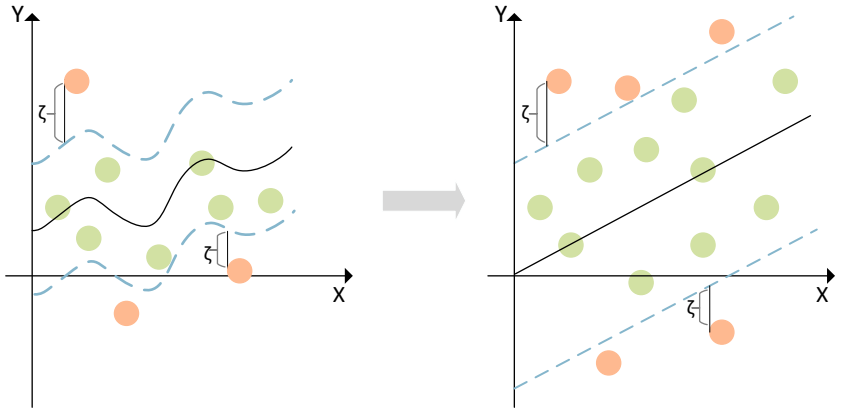

图 5.3 SVR 非线性目标超平面图

K 近邻算法先计算与目标最近的 K 个邻居，计算 K 个邻居的每一个属性的平均值，并将其赋值给样本得到样本相对应属性的值，即最终的回归结果。常用欧式距离计算，

$$d = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \cdots + (p_n - q_n)^2} \qquad (5.1)$$

其中，假设两个特征 $P(p_1, p_2, \ldots, p_n)$，$Q(q_1, q_2, \ldots, q_n)$，d 为用于度量相似性的距离结果。决策树回归采用树状的回归结构，将整个特征空间划分为若干个单元，且对应的每个单元都有一个输出，其中每一个节点都用是或否判断，则对于任意的输入特征，沿着树状结构将其归类到所属的单元即得到最终的输出。梯度提升回归继承了 Boosting 思想，集成多个弱回归器构建一个新的强回归器，利用同样的训练集训练多层的弱回归器，每层都使用训练集训练一个弱回归模型，并从训练中得到回归结果，下一层的回归计算的目的是减少上一层模型的残差，重复不断地计算直至残差值减小到可以接受的范围，最后将所有的回归器加权结合到一起组成了最终的梯度提升回归算法。

基于本问题数据集小的现实情况，为了得到更多的提取信息，我们构建了一

个基于 SVR 的组合 K 近邻、决策树和梯度提升的回归模型。

5.2 模型建立

我们构建的回归模型包含数据预回归，SVR 回归预测两部分组成，在提取特征信息的基础上进行更加有效的回归，模型结构如图 5.4 所示

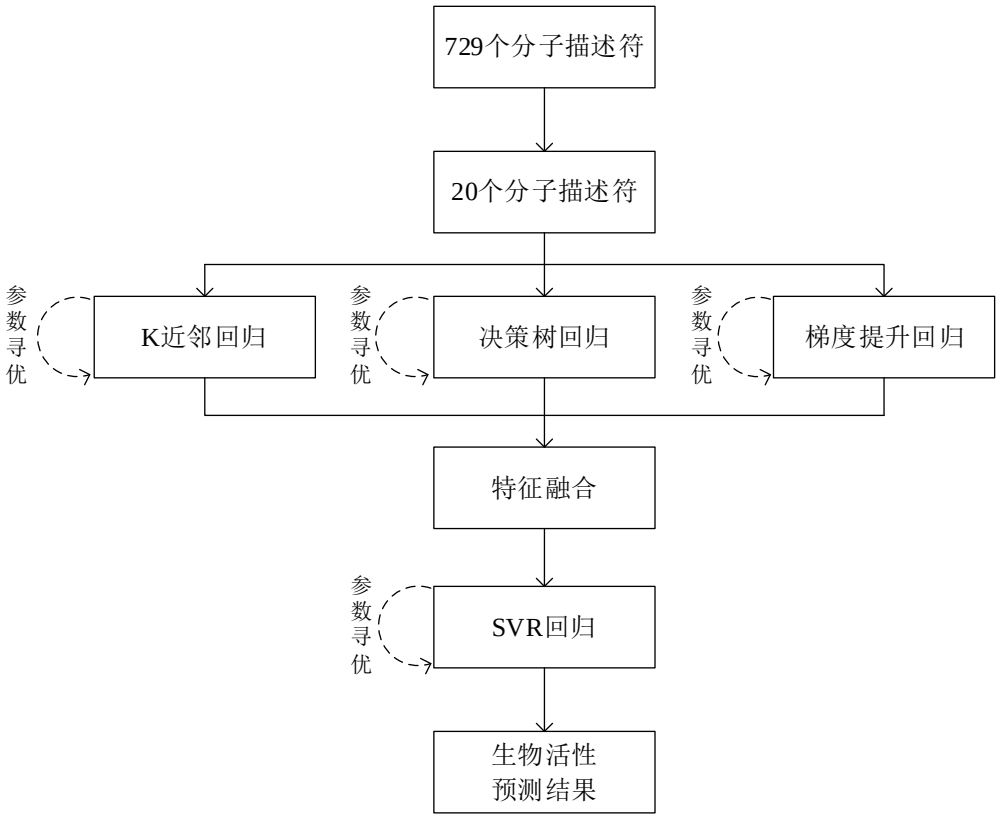

图 5.4 预测模型结构图

根据题目中给定的化合物的 729 个分子描述符，在第一问的基础上选择对生物活性最具有显著影响的 20 个分子描述符作为第二问的输入特征变量。首先对输入的特征变量做标准化处理，然后将其分别输入到 K 近邻回归、决策树回归和梯度提升回归三个回归模型中，由于在机器学习算法中参数寻优对于结果的存在很大的影响，因此将特征输入后，需要对回归模型分别寻找最优的参数，利用最优的参数分别得到三个回归分数，将三个回归分数再次做标准化处理输入到 SVR 模型中寻优，利用最优的参数得到最终的生物活性预测结果。

其中，每个模型的参数寻优过程异常重要，在实验中，我们利用网格搜索法分别寻找 K 近邻回归、决策树回归、梯度提升回归和 SVR 回归的最优参数，各个最优参数如表 5.1 所示：

表 5.1 网格搜索法寻优结果

回归方法	最优参数
K 近邻回归	K=1，weights=distance，leaf_size=30，p=1
决策树回归	splitter=best，min_samples_leaf=1,min_weight_fraction_leaf=0
梯度提升回归	n_estimators=80, max_depth=13, min_samples_split=100
SVR	kernel=rbf，C=10，degree=3，gamma= 1 / n_features，

5.3 生物活性预测结果与消融实验

5.3.1 划分训练测试集验证算法

在预测测试集生物活性之前，我们先将给定的训练集按照 8:2 的训练测试比划分为训练集和测试集用于验证我们采用的混合模型的有效性，训练测试结果如 5.5 所示：

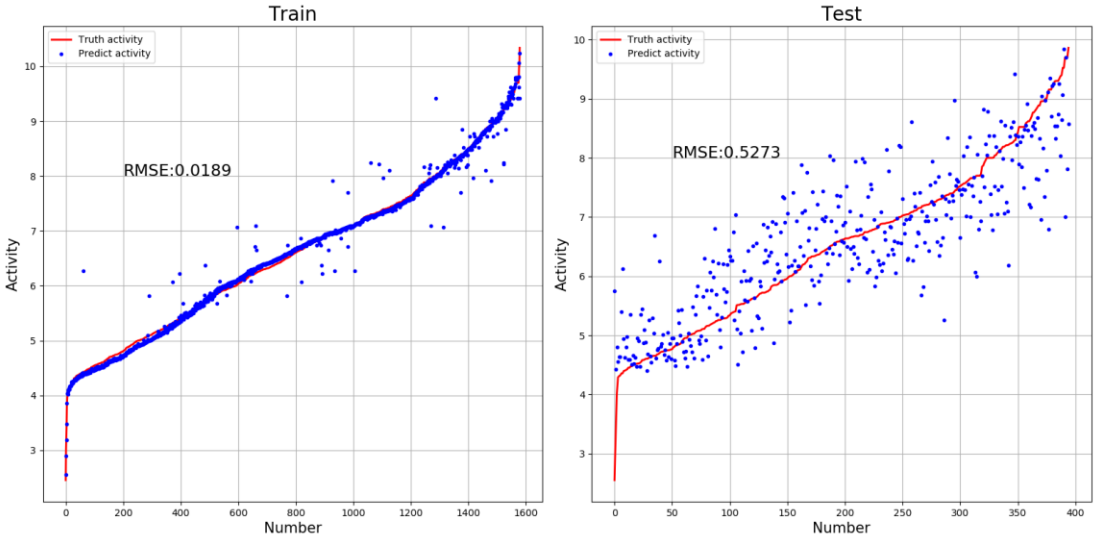

图 5.5 重新划分训练测试集验证结果

由上图可以得知，将给定的训练集按照 8:2 重新划分为训练和测试集后，我们的模型表现优异，在训练集上能够基本拟合所有数据，其余数据也非常接近真实数据，其均方根误差能够达到 0.0189。在测试集中，我们的模型仍然有着不错的表现，能够基本拟合数据，所有数据均没有出现太大误差，紧紧围绕在真实数据周围，测试集中的均方根误差达到 0.5291，显示了我们采用的 SVR 混合回归模型的有效性。

5.3.2 预测给定测试集的生物活性

经过训练集的初步验证，我们对给定的测试集进行了生物活性的预测。首先对整个训练集做训练，训练集结果如图 5.6 所示：

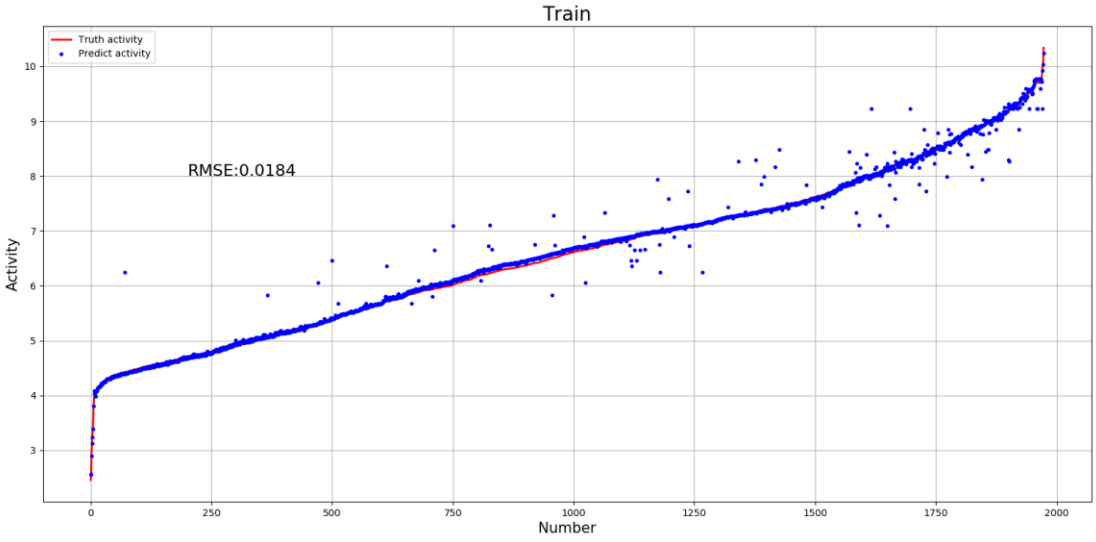

图 5.6 训练集结果

然后根据训练的模型预测测试集结果，如表 5.2 所示，出于对化合物名称长度和表格大小的考虑，我们按照测试集中化合物的顺序依次标号为 1-50。

表 5.2 测试集生物活性预测结果

化合物	IC50_nM	pIC50	化合物	IC50_nM	pIC50
01	14.72524502	7.83193747	26	75.41210786	7.12255892
02	72.17589866	7.14160780	27	49.90756947	7.30183358
03	70.91097552	7.14928654	28	167.1546914	6.77688143
04	36.28624763	7.44025794	29	224.8487616	6.6481095
05	24.09206022	7.61812606	30	142.1989370	6.84710365
06	120.3616354	6.91951192	31	18369.62751	4.73589965
07	241.2522320	6.61752866	32	17299.54816	4.76196524
08	163.5495185	6.78635073	33	24346.99071	4.61355471
09	27.37505308	7.56264503	34	18292.04580	4.73773772
10	14.48787966	7.83899517	35	25281.42154	4.59719851
11	15.01896966	7.82335986	36	32.73161544	7.48503256
12	22.14289650	7.65476557	37	42.18023160	7.37489104
13	35.56215004	7.44901199	38	90.76288041	7.04209173
14	35.13638685	7.45424290	39	546.8694338	6.26211635
15	26.53613940	7.57616226	40	102.9477952	6.98738295
16	29.25248532	7.53383723	41	103.3011104	6.98589501
17	9.804596379	8.00857028	42	103.0756946	6.98684373
18	381.2740521	6.41876275	43	103.6377820	6.98448189
19	27.55185093	7.55984922	44	102.0459610	6.99120418
20	0.188801613	9.72399430	45	103.3011104	6.98589501
21	129.6999414	6.88706022	46	305.0008502	6.51569895
22	69.82087546	7.15601471	47	105.3932944	6.97718702
23	65.51882345	7.18363391	48	160.4865755	6.79456129
24	82.84329039	7.08174266	49	434.3905812	6.36211960
25	244.5631672	6.61160895	50	52.26680944	7.28177401

5.3.3 不进行参数寻优的模型效果实验

为验证参数寻优对模型的作用，我们进行了实验验证，如图 5.7 所示, 分别为进行参数寻优和不进行参数寻优时的实验结果：

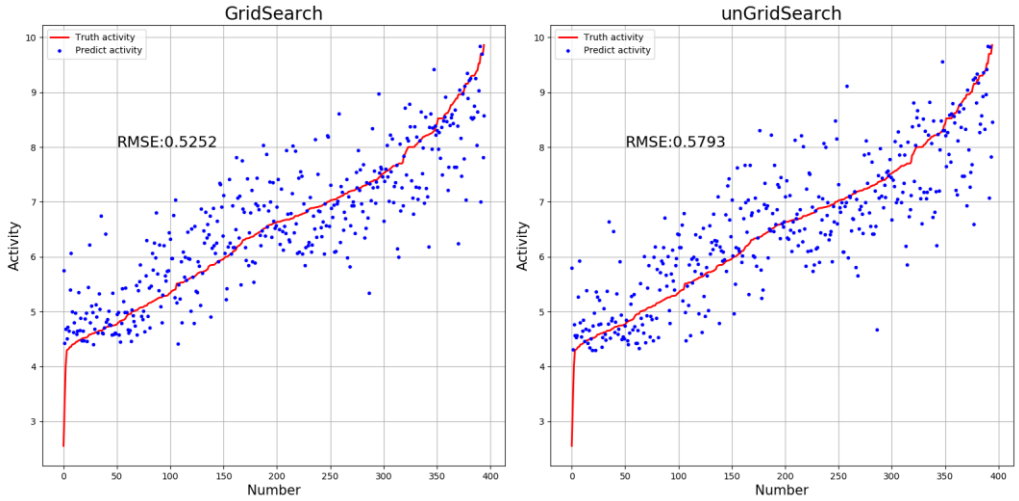

图 5.7 参数寻优消融实验

对数据集进行 8:2 的训练测试划分后，在测试集中进行实验，从图中可以看出，经过参数寻优后的模型对于数据的拟合效果更好，未经过参数寻优的模型预测的结果存在一些较大的偏差，如图中右下角和右上角的点远远偏离真实数据的曲线，总体性能不如经过参数寻优后的模型。

5.3.4 混合模型与单个模型比较

我们对采用混合模型和采用单个模型得到的生物活性预测结果进行了实验比较，采用了 RMSE）和 SROCC 来衡量模型预测回归的效果，其中，RMSE 比较模型预测的生物活性值与真实的生物活性值之间的绝对误差，也可衡量模型预测的准确性，SROCC 可衡量模型预测的单调性。RMSE 结果如表 5.3 所示：

表 5.3 混合与单个模型 RMSE 结果比较

模型	训练集 RMSE	测试集 RMSE
KNN 近邻回归	0.407	0.614
决策树回归	**0.016**	0.816
梯度提升回归	0.355	0.612
SVR	0.237	0.542
基于 SVR 的混合模型	0.019	**0.525**

从表 5.3 可以看出，基于 SVR 的混合模型训练集效果第二，测试集效果最优，其中决策树回归算法在训练集的效果最优，然而其在测试集中效果最差，证明其存在很大的不稳定性，综上，本模型在 RMSE 上相较于其他算法而言最优。

SROCC 结果比较如表 5.4 所示：

表 5.4 混合与单个模型 SROCC 结果比较

模型	训练集 SROCC	测试集 SROCC
KNN 近邻回归	0.894	0.833
决策树回归	**0.997**	0.794
梯度提升回归	0.908	0.826
SVR	0.942	0.854
基于 SVR 的混合模型	**0.997**	**0.859**

从表 5.4 可以看出，对于 SROCC 指标而言，基于 SVR 的混合模型在训练和测试集中均表现最优，且综合来看 SVR 算法排名第二，证明了把 SVR 放在模型最后一步做总回归的正确性。

5.3.5 鲁棒性验证

为检验混合模型的鲁棒性，在数据集中我们采用了 9:1,8:2,7:3,6:4,5:5 的训练测试比进行了实验，结果如表 5.5，表 5.6 所示：

表 5.5 RMSE 的鲁棒性实验比较

训练测试比	训练集 RMSE	测试集 RMSE
9:1	0.018	0.530
8:2	0.019	**0.525**
7:3	0.018	0.592
6:4	**0.016**	0.608
5:5	0.017	0.685

表 5.5 展示了改变训练测试比后的模型 RMSE 结果，从结果可以看出，模型在各种训练测试比例的情况下均能保持很小的误差，在 5:5 的情况下仍能达到 0.685 的小误差，显示了我们采用的模型具有很好的鲁棒性。

表 5.6 SROCC 的鲁棒性实验比较

训练测试比	训练集 SROCC	测试集 SROCC
9:1	0.997	**0.986**
8:2	0.997	0.859
7:3	0.997	0.838
6:4	0.997	0.837
5:5	0.997	0.823

在改变训练测试比的情况下，我们的模型在训练集上保持在了 0.997 的极高 SROCC 结果，当训练测试比为 9:1 时，测试集中的 SROCC 结果最好，为 0.986，即使采用 5:5 的训练测试比，我们的模型在测试集中仍然能达到 0.823 的较高 SROCC 结果，再一次验证了我们模型具有较好的鲁棒性。

6. 问题三分析与求解

6.1 问题分析

首先，分析题目提供的数据，确定是否需要做相应的预处理，然后我们建立两种分类模型，对数据进行模拟预测。下面我们分步介绍建模过程。

6.2 数据分析与筛选

观察文件“Molecular_Descriptor.xlsx”提供的数据不难看出，部分分子描述符数据均为 0 值，这些低方差的数据会对模型的训练产生干扰，因此我们采用 matlab 编程，对 729 组分子描述符数据进行初步筛选，若某种分子描述符的数据值均等于 0，则去除该分子描述符。数据处理前后，数据分布情况分别如表 6.1 与 6.2 所示：

表 6.1 原始数据分布情况

	训练数据	测试数据
ADMET 数据	1974×729	50×729
Caco-2 标签	1974×1	无
CYP3A4 标签	1974×1	无
hERG 标签	1974×1	无
HOB 标签	1974×1	无
MN 标签	1974×1	无

表 6.2 筛选后数据分布情况

	训练数据	测试数据
ADMET 数据	1974×505	50×505
Caco-2 标签	1974×1	无
CYP3A4 标签	1974×1	无
hERG 标签	1974×1	无
HOB 标签	1974×1	无
MN 标签	1974×1	无

6.3 SVM 二分类模型建立

传统的机器学习分类通常基于支持向量机（SVM）[11]线性判别分析,随机森林，决策树等模型实现。为了方便使用，这些模型都很好的集成到 MATLAB 中。我们基于筛选后的数据建立 SVM 二分类模型，最直观的想法是将每个化合物的 505 种分子描述符信息作为特征值，送入 SVM 分类器进行分类。

考虑到需要区分的两类数据可能有相似的特性，因此对该问题进行转换，将数据看成一串序列，进而采用公共空间模式（CSP）算法[12]将不同类别数据的差异扩大。首先采用 CSP 算法设计空间滤波器使得两组信号矩阵滤波后，方差值差异最大化，从而得到具有较高区分度的特征向量。在下一步将其送入 SVM 分类器进行分类。模型流程图如图 6.1 所示。

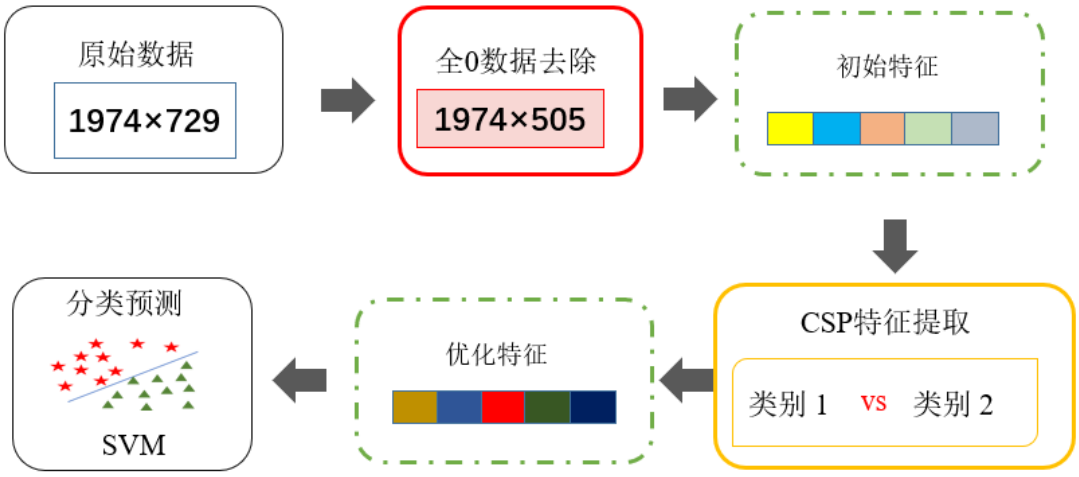

图 6.1 模型流程图

6.3.1 CSP 特征提取

共空间模式（CSP）作为一个二分类的特征选择方法，可以从数据中选择出每一种信号的空域组成状况。CSP的数学原理是计算特征矩阵的对角矩阵，然后找到最优的一组使两种信号的方差相差值最大的空间滤波器，最后获取区分程度比较大的特征向量。它的详细过程如下：

设E_1，E_2是两种$N \times T$大小的信号，N为通道个数，T为特征个数。如果不算信号噪声产生的影响，EEG 能够写为多种初始信号的结合，具体如下公式（6.1）所示：

$$E_1 = [D_1 D_M][Z_1 Z_M]^T, E_2 = [D_2 D_M][Z_2 Z_M]^T \tag{6.1}$$

式(6.1)中，D_1包括一定数量的只与 E_1相关的空间分量，D_2包含一定数量的只与E_2相关的空间分量，D_M包括一定数量的与 E_1、E_2都有关的空间形式。Z_1和Z_2是两种有独立线性的不同信号。Z_M代表两类行为一起有的初始信号，假设Z_1是由M_1个行为初始信号组成，Z_2是由M_2个组成。D_1和D_2是M_1和M_2个与Z_1和Z_2有关的共同空间模式组和而成的，因为所有的空间模式均是$N \times 1$大小的向量，所以单个源产生的信号在N导联上分布权重是用这样的一个向量来代表。D_M代表和Z_M对应的共同的空间模式。CSP 算法的思想是构造 F_1和 F_2两个空间滤波器来获得表示权重的空间因子 W。

（1）计算两类数据的混空间协方差矩阵

下面算出两类数据的归一化协方差矩阵 G_1和G_2，计算通式如下:

$$G_i = E_i {E_i}^T / trace\left(E_i {E_i}^T\right) \tag{6.2}$$

这里，${E_i}^T$是E_i的转置，$trace\left(E_i {E_i}^T\right)$运算表示矩阵的秩，i=1,2。下式(6.3)给出两者的混合协方差矩阵G。

$$G = \overline{G_1} + \overline{G_2} \tag{6.3}$$

其中 G_1和 G_2是两种行为类别的平均协方差矩阵。

（2）采取主成分分析法，计算白化特征矩阵 Q

G 是正定矩阵，对其分解出的特征值可以表示成：

$$G = AHA^T \tag{6.4}$$

其中 A 是 G 的特征向量，H 为对角阵，它是由 A 所对应的特征值构成的。白化矩阵 Q 的表示：对角阵 A 中的元素进行由大到小排列，相应调整特征向量，如公式（6.5）所示。

$$Q = \sqrt{H^{-1}}A^T \tag{6.5}$$

（3）构造空间滤波器

如式（6.6）、（6.7）所示，分别对 G_1和 G_2做白化处理。

$$Z_1 = QG_1Q^T,\ Z_2 = QG_2Q^T \tag{6.6}$$

然后主分量分解 Z_1和 Z_2，有下面的公式：

$$Z_1 = K_1H_1{K_1}^T,\ Z_2 = K_2H_2{K_2}^T \tag{6.7}$$

从式子（6.7）能够看出矩阵 Z_1的特征向量和矩阵 Z_2的特征向量矩阵是一样的，即公式（6.8）：

$$K = K_1 = K_2 \tag{6.8}$$

并且有，两个对角阵H_1和H_2相加是单位矩阵，如公式（6.9）：

$$H_1 + H_2 = I \tag{6.9}$$

因为两类矩阵的特征值之和恒为 1，所以当某个特征向量使 Z_1的最大特征值时，Z_2有最小特征值，相反也是这样。如果H_1的特征值由大到小排列，那么H_2的相应的特征值则由小到大排列，根据这个关系可以得出H_1和H_2具有下面公式（6.10）的表示：

$$H_1 = diag(I_1, \sigma_M, 0),\ H_2 = diag(0, \sigma_M, I_2) \tag{6.10}$$

为了找到与H_1和H_2中特征值最大时相对的特征向量，常常用对 EEG 做白化处理来实现这种变化，这是对两个信号矩阵之中的方差划分开的最好的方法。最后得到的空间滤波器相对的投影矩阵 W 表示为公式（6.11）：

$$W = K^TQ \tag{6.11}$$

（4）特征提取

上述分析针对与三维数据，而我们的数据集只有二维，将训练集中的数据扩

展一个维度得到矩阵Z，通过把Z送入得到的相应滤波器 W 进行滤波处理有特征 S，公式为（6.12）：

$$S = W \times Z \tag{6.12}$$

我们选择F作为特征向量，公式如下（6.13）：

$$F = var(S)/sum\big(var(S)\big) \tag{6.13}$$

而对测试数据Z_r来说，它的特征向量F_r的选取方法如下:

$$S_r = W \times Z_r,\ F_r = var(S_r)/sum\big(var(S_r)\big) \tag{6.14}$$

公式如（6.14），得出测试的F_r特征。

通过上述计算，数据中的不同特征的鉴别性得到了很大的提升，然后利用处理后的特征F, F_r进行之后的分类操作。

6.3.2 SVM 模型分类

SVM 是一种二分类模型，这种方法的目标为找到一个超平面进而将样本分割成间距最大的两部分，最后转变成凸二次规划问题来分析。分析线性不可分的数据，核函数的选取优劣直接决定了高维特征空间的好坏，使得到的 SVM 的分类性能存在较大影响。

SVM 作为一种十分经典的机器学习算法，具有优秀的分类能力和快速的计算效率。其基本思路是通过核函数将输入数据从一个低维空间转换到高维空间以找到一个最佳超平面，它不仅能正确地划分样本，而且能最大化几何区间。如图 6.2 所示，图中的 H 就是使分类效果最优的超平面。SVM 算法比较适合样本数量较小的数据集，可以有效地解决低维度信号的非线性问题。

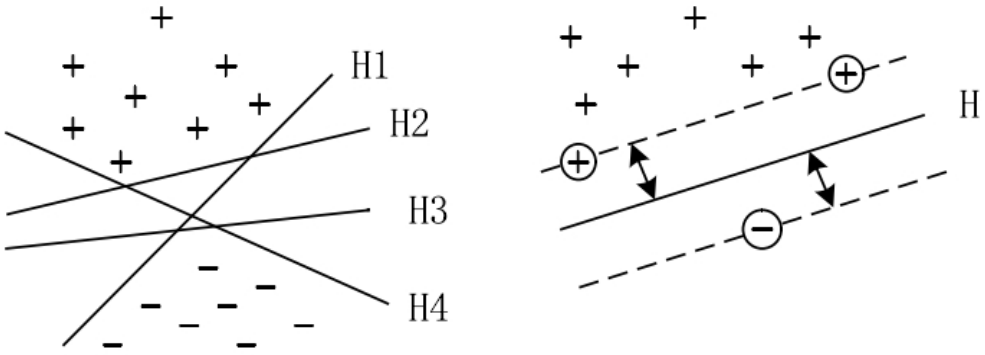

图 6.2 SVM 原理图

我们采用 RBF 核函数的 SVM 对提取出的 CSP 特征进行二分类，将带有标签的 1974 个化合物按 9：1 的比例分成训练集和验证集。建立的 5 个二分类模型分析模型性能。分类准确率结果如表 6.3 所示。

表 6.3 5 个不同 SVM 模型的验证集准确率

Caco-2	CYP3A4	hERG	HOB	MN
89.09%	91.35%	85.24%	80.37%	75.76%

分析表 6.3 可知，我们的模型具有较好的性能，随后我们将测试集的 50 组数

据送入该数学模型，将 5 个二分类模型对测试集的 50 个预测结果汇总在表 6.4 中。其中 L1-L5 分别对应 Caco-2，CYP3A4，hERG，HOB，MN。

表 6.4　50 个待测化合物在 SVM 模型下的分类结果

	L1	L2	L3	L4	L5		L1	L2	L3	L4	L5
1	0	1	1	0	1	26	0	1	1	0	1
2	0	1	1	0	1	27	0	1	1	0	1
3	0	1	1	0	1	28	0	1	1	0	1
4	0	1	1	0	1	29	0	1	1	0	1
5	0	1	1	0	1	30	0	1	1	0	1
6	0	1	1	0	1	31	1	1	0	0	1
7	0	1	1	0	1	32	1	1	0	0	1
8	0	1	1	0	1	33	1	1	0	0	1
9	0	1	1	0	1	34	1	1	1	0	1
10	0	1	1	0	1	35	1	1	1	0	1
11	0	1	1	0	1	36	0	1	0	0	1
12	0	1	1	0	1	37	0	1	0	0	1
13	0	1	1	0	1	38	0	1	1	0	1
14	0	1	1	0	1	39	0	1	1	0	1
15	0	1	1	0	1	40	0	1	1	0	1
16	0	1	1	0	1	41	0	1	1	0	1
17	0	1	1	0	1	42	0	1	1	0	1
18	0	1	1	0	1	43	0	1	1	0	1
19	0	1	1	0	1	44	0	1	1	0	1
20	0	1	1	0	1	45	0	1	1	0	1
21	0	1	1	0	1	46	0	1	1	0	1
22	0	1	1	0	1	47	0	1	1	0	1
23	1	1	0	0	1	48	0	1	1	0	1
24	1	1	0	0	1	49	0	1	1	0	1
25	1	1	1	0	1	50	0	1	1	0	1

通过对结果进行分析，我们发现 5 个不同的分类模型性能有差异，Caco-2，CYP3A4，hERG 标签分类性能较好，HOB，MN 标签的分类性能次之。这可能是数据和标签的本身的问题，为了进一步探索原因和改进模型，我们采取神经网络的方法建立新模型再次分类预测。

6.4 基于自注意力的多尺度 MLP 二分类模型建立

随着神经网络的发展，许多数学问题都可以通过神经网络有效地解决。我们受到自注意力机制[13]的影响，将每种化合物的不同描述符信息进行全局自注意力加权，获取每一个指标与其他指标之间的相关性。然后采用多尺度的一维卷积，对加权后的特征进行融合。提高不同数据的鉴别性。最终将数据特征送入两层全连接的 MLP[14]进行分类预测。模型框架如图 6.3 所示。

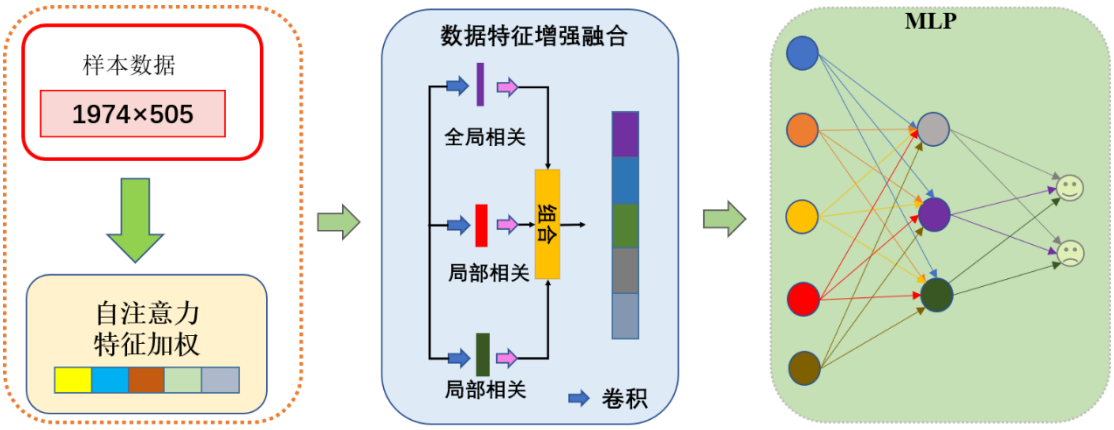

图 6.3 网络框架流程图

注意力机制一直深受科研工作者的喜爱，但传统的注意力机制在处理高度全局相关的数据时存在不足。随着人们对数据本身的相关性的需求，自注意力机制成为当前的研究热点，谷歌公司的科研人员首先在自然语言处理领域提出了一种自注意力机制，用于机器翻译等任务。自注意力机制的原理如图 6.4 所示

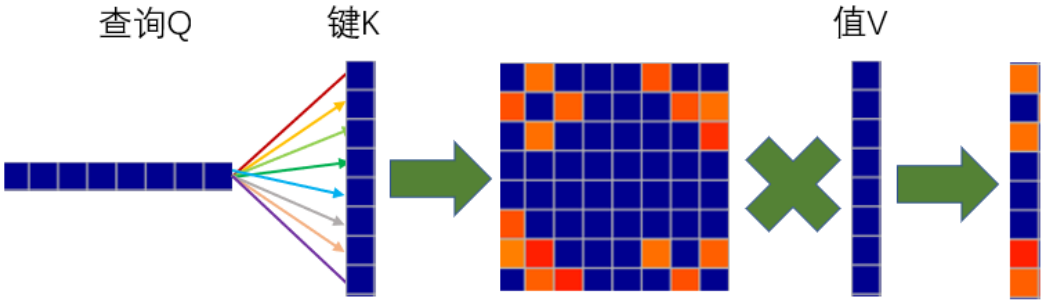

图 6.4 自注意力加权示意图

多层感知机（MLP）把获得的特征再次通过权值网络组合为完整的特征矩阵，其基本思想是通过一种线性变换的方式将一个特征空间变到另一个特征空间。

训练神经网络时，由于训练数据比较少或者模型过于复杂等等原因，网络经常发生数据过拟合情况[15]。因此为了预防或者减小神经网络过拟合，进行训练的时候，我们随机丢弃一些神经元，如图 6.5、6.6 所示，训练时通过一定的比例对部分神经元输出乘零，进而删除该神经元。被选择丢弃的神经元的权值被保留，训练时不再进行更新。该操作多用于全连接层，在本模型中，我们在模型中选择带有 dropout 方式的全连接网络层。

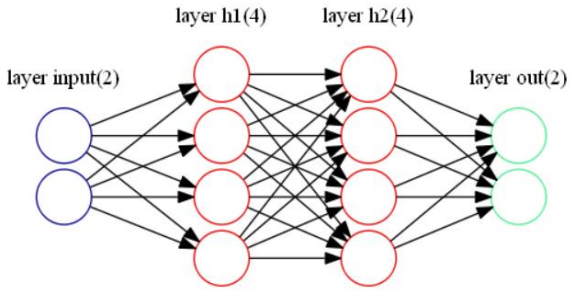

图 6.5 全连接方式

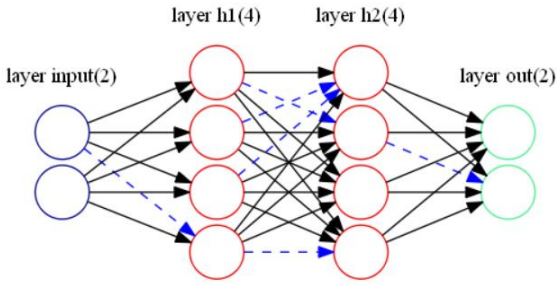

图 6.6 随机 dropout 方式

最终输出层使用 Softmax 函数，其确保全部输出之和是 1，输出概率是每个输出对应的[0,1]区间的数值，在应用时取最大概率输出作为最终的预测[16]，将网络特征选取完成之后，就能够进行分类了。其分类原理如图 6.7 所示。

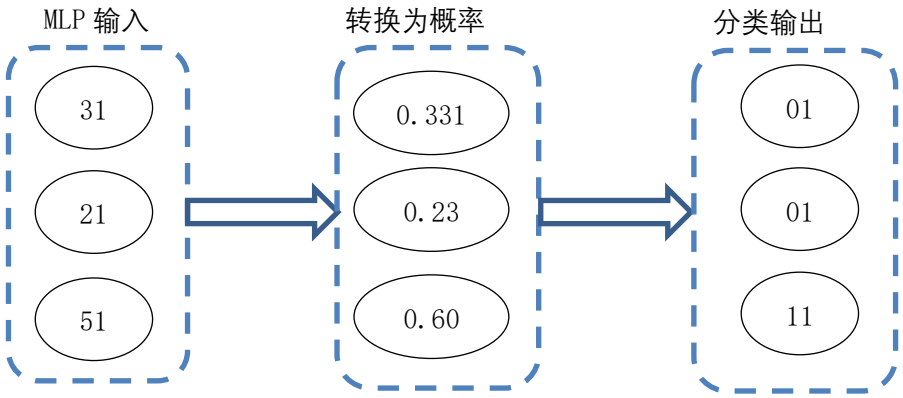

图 6.7 softmax 分类器工作原理

全连接的最终输出节点个数等于 2， softmax 分类器在全连接层的每个输出转化为每个节点的概率分布，选取概率最大的节点作为最终分类结果，所有样本在 MLP 中输出为预测类别 0 或 1。为了更加直观地展示网络结构，我们将模型的主要参数总结于表 6.5 中。

表 6.5 基于自注意力的多尺度 MLP 网络主要参数

模块	层数	输出	操作
自注意力多尺度卷积	input	(1,1,505)	
	Self-attention	(1,1,505)	一维卷积 RELU 激活
		(1,1,505)	自注意力
	多尺度一维度卷积	(8,1,20)	卷积 RELU 激活
		(8,1,128)	卷积 RELU 激活
		(8,1,300)	卷积 RELU 激活
	特征输出	(8,1,128)	池化
MLP	input	(8,1,128)	
	Fc1	256	RELU 激活 dropout=0.2
	Fc2	2	SoftMax

此外，网络选择交叉熵损失进行优化，网络的学习率为 0.01.优化器选择 SGD。在模型输入上，我们将带有标签的 1974 个化合物按 9：1 的比例分成训练集和验证集，得出最终在验证集上的分类准确率验证模型的性能。性能结果如表 6.6。

表 6.6 5 个不同 MLP 模型的验证集准确率

Caco-2	CYP3A4	hERG	HOB	MN
89.42	92.89	88.62	85.02	78.87

5 个 MLP 模型在测试集上的可视化结果如图 6.8 所示。

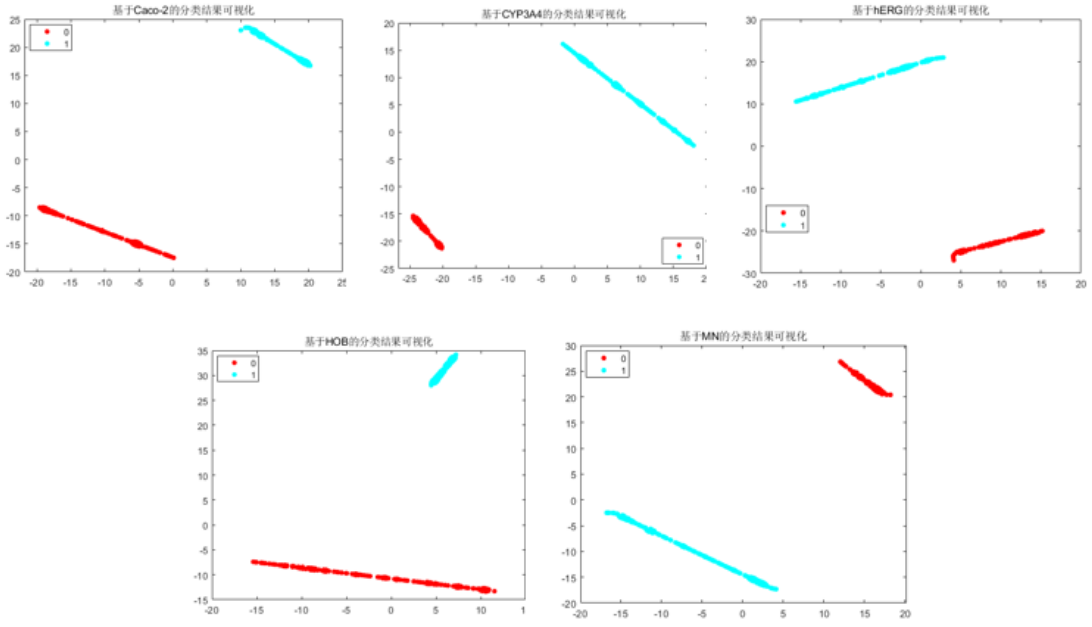

图 6.8 五个 MLP 模型测试集可视化

最终在 50 个化合物上的预测情况如表 6.7 所示。

表 6.7 MLP 模型预测的 50 个化合物的对应标签

	L1	L2	L3	L4	L5		L1	L2	L3	L4	L5
1	0	1	1	0	1	26	1	1	0	0	0
2	0	1	1	0	1	27	0	1	1	0	1
3	0	1	1	0	1	28	1	1	1	0	1
4	0	1	1	0	1	29	0	1	1	0	1
5	0	1	1	0	1	30	1	1	1	0	0
6	0	1	1	0	0	31	0	1	1	0	1
7	0	1	0	0	1	32	0	1	1	0	1
8	0	1	1	0	1	33	0	1	0	0	1
9	0	1	0	0	1	34	0	1	1	0	1
10	0	1	1	0	1	35	0	1	0	0	1
11	0	1	1	0	1	36	0	1	1	0	1
12	0	1	1	0	1	37	0	1	1	0	1
13	0	1	1	0	1	38	0	1	1	0	1
14	0	1	1	0	1	39	0	1	1	0	1
15	0	1	1	0	1	40	0	1	1	0	1
16	0	1	0	0	1	41	0	1	1	0	1
17	0	1	1	0	1	42	0	1	1	0	1
18	0	1	1	0	1	43	0	1	1	0	0
19	0	1	1	0	1	44	0	1	1	0	1
20	0	1	1	0	1	45	0	1	1	0	1
21	0	1	1	0	1	46	0	1	0	0	1
22	0	1	1	0	1	47	0	1	1	0	1
23	0	1	1	0	1	48	0	1	1	0	1
24	0	1	1	0	1	49	0	1	0	0	1
25	0	1	0	0	1	50	0	1	1	0	1

6.5 模型评价

通过与SVM模型比较，MLP模型的验证集上的分类准确率优于SVM模型。两个模型在最后两个标签 HOB，MN 上的分类效果较差，为了进一步分析其背后原因，我们通过对原始数据进行分析，可视化了原始数据的样本组成，如图 6.9 所示。

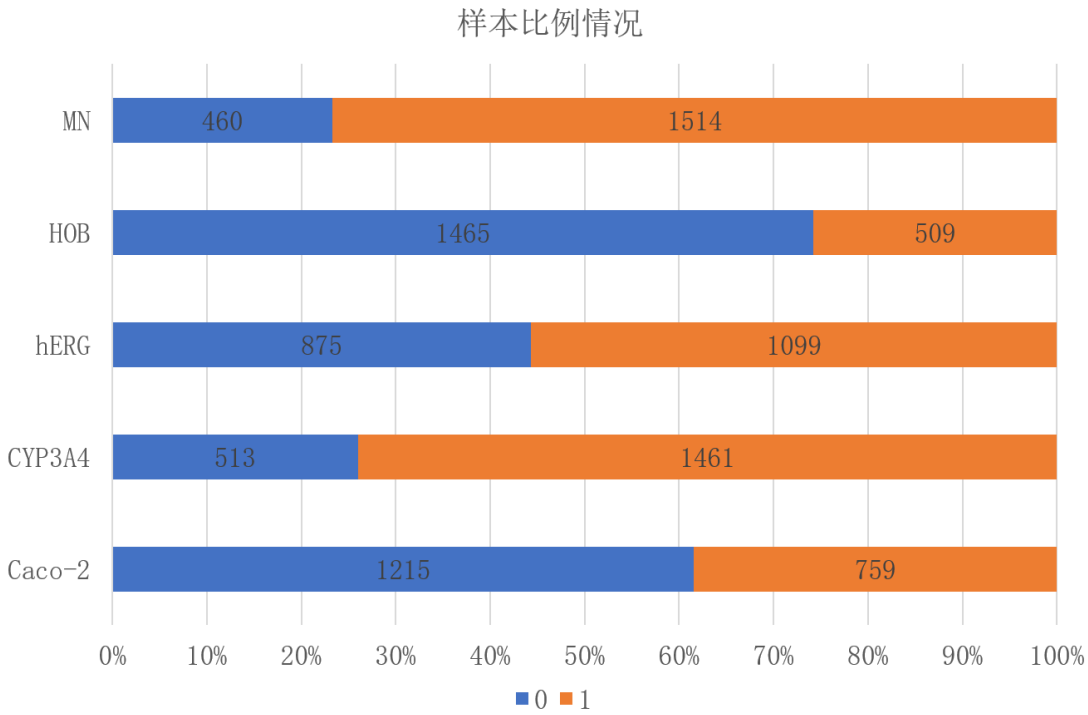

图 6.9　1974 个化合物中的正负样本数统计图

由图 6.9 可知，本题提供的训练数据存在类别不平衡，这可能是造成模型分类性能缺失的一大因素。解决数据偏斜的方法有许多，比如利用现有的集成学习分类器，如随机森林或者 XGBOOST 算法调整分类阈值。

7. 问题四分析与求解

7.1 问题分析

本题是对问题 1，2，3 的综合利用，问题一中通过数据筛选，得到了 20 个对生物活性影响程度最高的分子描述符，本题针对问题一中 20 个分子描述符进行分析。根据题目要求，本题需在保证 ADMET 性质较好的情况下尽可能获得较好的生物活性，因此需首先确定 5 个 ADMET 性质的最优组合。通过对材料分析可知，当化合物取得较好的 ADMET 性质时，化合物的小肠上皮细胞渗透性较好，不能被 CYP3A4 代谢，不具有心脏毒性，口服生物利用度较好，且不具有遗传毒性。根据上述原则，可以确定 5 个 ADMET 性质的最佳组合，如表 7.1 所示。

表 7.1 ADMET 性质最佳组合结果

性质	Caco-2	CYP3A4	hERG	HOB	MN
标签	1	0	0	1	0

本题整个思路如图 7.1 所示。

第一步：以问题 1 中筛选出的 20 个分子描述符作为原始输入数据，将 50 种待测化合物送入问题 2 中构建的回归模型中，预测 50 种化合物各自的生物活性值，并取出其中生物活性值最高的 3 种化合物，作为问题 3 中分类模型的待测数据。

第二步：将 1974 种化合物对应的 20 个分子描述符数据作为训练集，重新训练问题 3 中的分类模型，随后将第一步中得到的 3 种化合物输入分类模型，预测其 ADMET 性质。

第三步：将生物活性最高的三种化合物的 ADMET 性质与表 7.1 中性质最好的情况进行对比，若化合物的 5 个 ADMET 性质中有 3 个及以上性质较好，则记录下该化合物的分子描述符号取值。若不满足上述条件，则根据问题 1 的前 20 个分子描述符的重要性排序，剔除得分最低的分子描述符，继续重复上述步骤。实验中观察到，用于训练的分子描述符过少时，模型的性能会出现不稳定的情况，因此我们选择将分子描述符数量为 10 作为上述循环过程的终止条件，最终以筛选出的性质较好的化合物为基础，确定分子描述符的取值。

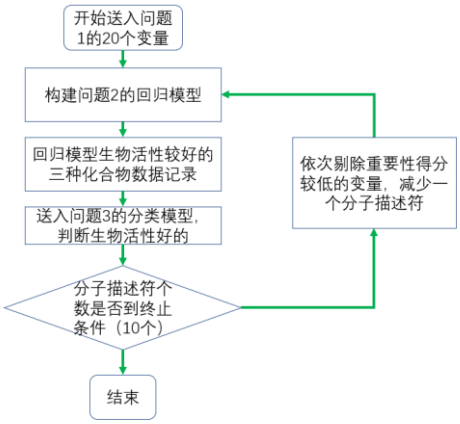

图 7.1 问题 4 整体流程图

7.2 模型建立

该问题建立的模型与问题 2，问题 3 原理一致，本题不再赘述。主要区别在于，模型的输入从 20 个分子描述符依次剔除对生物活性影响重要性得分低的描述符，同时每次遍历过程中，需要对问题 2 和问题 3 中构建的模型进行重新训练，训练数据集为 1974 种化合物中当前分子描述符的全部数据。依据问题 1 中的重要性排序从低到高依次减少分子描述符，统计生物活性值最高的 3 种化合物中 ADMET 性质的分布情况，分析不同分子描述符在何取值时，在满足高生物活性的同时 ADMET 性质最好。

表 7.2 展示了使用 20 个分子描述符作为训练数据的情况下，测试集 50 个化合物在问题 3 中分类模型下各自的预测结果。

表 7.2 20 个分子描述符问题 3 模型下的预测结果

	L1	L2	L3	L4	L5		L1	L2	L3	L4	L5
1	1	1	0	0	0	26	1	0	0	0	0
2	1	1	0	0	0	27	1	1	1	0	0
3	1	1	0	0	0	28	1	1	1	0	0
4	1	1	0	0	0	29	1	1	1	0	0
5	1	1	0	0	0	30	1	1	1	0	0
6	1	1	0	0	0	31	1	0	0	0	0
7	1	1	0	0	0	32	1	0	0	0	0
8	1	1	0	0	0	33	1	0	0	0	0
9	1	1	0	0	0	34	1	0	0	0	0
10	1	1	0	0	0	35	1	1	1	0	0
11	1	1	0	0	0	36	1	1	1	0	0
12	1	1	0	0	0	37	1	1	1	0	0
13	1	1	0	0	0	38	1	1	1	0	0
14	1	1	0	0	0	39	0	1	1	0	1
15	1	1	0	0	0	40	0	1	1	0	1
16	1	1	0	0	1	41	0	1	1	0	1
17	1	1	0	0	0	42	0	1	1	0	1
18	1	1	0	0	0	43	0	1	1	0	1
19	1	1	0	0	0	44	0	1	1	0	1
20	1	1	1	0	0	45	0	1	1	0	1
21	1	1	0	0	0	46	1	1	1	0	0
22	1	1	0	0	0	47	1	1	0	0	0
23	1	0	0	0	1	48	1	1	0	0	0
24	1	0	0	0	0	49	1	1	1	0	0
25	1	0	0	0	0	50	1	1	1	0	0

由上表对比表 6.7 分析可知，当输入化合物的分子描述符不同时，问题三建立的模型得出的分类结果也不同，因此每次依次剔除数据后，由于改变了输入数据，因此模型分类结果也会产生相应的变化。我们统计了输入不同个数分子描述符时，生物活性最好的化合物种，同时具有三个及以上 ADMET 较好性质的化合物序号（1-50）。如表 7.3 所示。

表 7.3 不同分子描述符个数时两个条件同时满足的化合物序号

分子描述符个数	生物活性好，同时 ADMET 性质好的化合物编号
20	10 和 17
19	无
18	无
17	7
16	3 和 4
15	3 和 4
14	7
13	7 和 19
12	7 和 19
11	1 和 7
10	无

7.3 结果分析

根据表 7.3，我们找到了在输入不同数量的分子描述符作为特征时同时满足两个条件的化合物，通过数据分析符合条件的化合物的数据，我们得出其对应的分子描述符，如表 7.4 所示

表 7.4 不同编号化合物的取值表

分子描述符	10	17	1	7	3	4	19
MDEC-23	40.617	44.292	44.466	43.568	39.275	39.275	36.309
C1SP2	1	1	0	0	1	1	1
WTPT-5	3.197	3.038	0	0	0	0	0
SHBint6	0	0	0	0	0	0	0
MDEC-33	11.577	12.322	16.351	11.577	11.577	11.577	10.958
mindssC	-0.010	-0.208	0.767	1.020	-0.981	-1.006	0.935
BCUTc-1h	0.226	0.228	0.278	0.227	0.283	0.283	0.150
VC-5	0.133	0.107	0.208	0.107	0.107	0.107	0.107
maxHsOH	0.521	0.521	0.514	0.521	0.695	0.702	0.546
SHBint10	0	0	17.096	0	0	0	0
maxssO	5.904	5.907		5.925	6.059	5.837	5.814
MLFER_A	0.546	0.455		0.546	1.134	1.134	0.546
VPC-5	2.689	2.428		2.375	2.282	2.202	2.140
SPC-6	10.926	11.296		10.539	10.184	10.155	
minssCH2	0.421	0.410		0.452	0.513	0.398	
minHsOH	0.521	0.521		0.521	0.528	0.538	
XLogP	4.363	4.271		6.311			
LipoaffinityIndex	12.134	12.627					
mindO	12.289	12.259					
minHBint6	0	0					

通过分析表 7.4，可以找到化合物的取值情况，因此得到最终结果，当 20 个分子描述符的取值在如下表中的情况时，满足活性最好条件的同时 ADMET 具有三个以上的较好性质。

表 7.5 最终 20 个分子描述符应该满足的范围

分子描述符	取值范围
MDEC-23	36.309-44.466
C1SP2	1
WTPT-5	0
SHBint6	0
MDEC-33	10.958-16.351
mindssC	-0.981-1.020
BCUTc-1h	0.150-0.283
VC-5	0.107-0.208
maxHsOH	0.521-0.702
SHBint10	0
maxssO	5.904-6.059
MLFER_A	0.455-1.134
VPC-5	2.140-2.689
SPC-6	10.155-11.296
minssCH2	0.398-0.513
minHsOH	0.521-0.538
XLogP	4.271-6.311
LipoaffinityIndex	12.134-12.627
mindO	12.259-12.289
minHBint6	0

表 7.5 即为能够使化合物具有更好的生物活性，同时具有三个及以上更好的 ADMET 性质的分子描述符的取值范围。

8. 模型评价

8.1 模型的优点

优点 1：采用多阶段渐进式数据筛选算法，在保证数据有效性和相关性的基础上通过随机森林判断分子描述符特征的重要性，有效地避免了原始数据中部分无效数据对随机森林性能的影响。

优点 2：采用基于 SVR 的组合 K 近邻、决策树和梯度提升的回归模型能够提取多种有效的特征信息作为中间变量，能够适应特征的变化，进而提高模型的预测准确性。

优点 3：采用基于 CSP 算法建立的 SVM 分类模型一可以提高特征的鉴别性，在数据量有限的情况下，保证分类性能。同时，我们提出了进一步改进的基于自注意力多尺度卷积的 MLP 分类模型，该模型通过自注意力机制获取不同特获的全局相关性，同时采用一位卷积融合局部和全局的特征相关性，送入 MLP 网络实现了很好的分类性能。

8.2 模型的缺点

缺点 1：多阶段数据筛选相比于单阶段消耗的时间成本更高，此外，由于专业的限制，本文对各个化合物分子描述符的化学本质并未深入分析，因此筛选工作仅基于数据，若能融合专业相关指数，可以对问题进行更加透彻的分析。

缺点 2：将输入的特征首先送入三个回归模型最终再送入 SVR 得到预测结果的过程计算量比较大，消耗的时间相比较单一模型而言较多。

缺点 3：由于数据存在类别不平衡的现象，构建的 5 个二分类模型，在数据类别不平衡的情况下分类性能受损，这是数据本身的问题，如果需要改进，可以采用集成学习，设置类别阈值的方法进行优化。

参考文献

[1] 王冬,谈谣,于海洋,韩立峰,王涛.中药多靶点抗乳腺癌的研究进展[J].中南药学,2019,17(10):1600-1607.

[2] 杨凯, 侯艳, 李康. 随机森林变量重要性评分及其研究进展[J]. 2015.

[3] 颜海波,邓罡,姜云卢.基于 MRCD 估计的多元线性回归模型的稳健估计[J/OL].广西师范大学学报(自然科学版):1-10[2021-10-16].

[4] 卢德俊,爨凯旋,张伟峰.局部 k 最近邻加权线性回归的光谱反射率重建[J].光谱学与光谱分析,2018,38(12):3708-3712.

[5] 王泽,张玉敏,吉兴全,徐波,杨明,韩学山.基于深度学习与内核岭回归的电力系统鲁棒状态估计[J/OL].高电压技术:1-12[2021-10-16].

[6] 任首朋,李劲,王静茹,岳昆.基于集成回归决策树的 lncRNA-疾病关联预测方法[J/OL].计算机科学:1-9[2021-10-16].

[7] 陈岩,侯群,关雅琦.梯度提升回归树在风力发电机温度预测的应用研究[J].电子世界,2021(16):91-94.

[8] 邓红涛,贾琼,李绍军,李伟.基于 LASSO 回归的 R-vine copula 模型构建及其在化工过程故障检测中的应用[J/OL].重庆大学学报:1-10[2021-10-16].

[9] T. Silva Barbosa and A. Henrique Kronbauer, "Panorama of Researches Related to the Application of Virtual Reality in the Health Area in SVR," 2019 21st Symposium on Virtual and Augmented Reality (SVR), 2019, pp. 69-76.

[10] 吕超,朱雪阳,丁忠林,丁仪,朱秋阳.基于 5G 与 CNN 的智能电网稳定性预测[J].计算机系统应用,2021,30(07):158-164.

[11] 蒋礼君,张晓格.频谱感知中的 K-D 树 KNN-SVM 算法研究[J].现代电子技术,2021,44(16):7-13.

[12] 谷学静,位占锋,刘海望,郭俊,沈攀.共空间模式结合卷积神经网络的脑电信号分类[J].激光杂志,2021,42(04):100-104.

[13] Ashish Vaswani, Noam Shazeer, Niki Parmar, et al. Attention Is All You Need [C]. 31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

[14] 邓力元,刘桂波,户秋月,顾洁琼,刘衍斌.基于 MLP 模型和模糊控制的盆栽灌溉系统[J].计算机时代,2021(09):34-38

[15] 周飞燕，金林鹏，董军，卷积神经网络研究综述[J].计算机学报，2017，40(06):1229-1251

[16] 蒋昂波，王维维。ReLU 激活函数优化研究[J].传感器与微系统，2018,37(02):50-52.

附录

问题 1 重要性筛选主要程序：

第一步剔除 0

```
select_zero = [];
for i=1: 729
    count = 0;
    rate = 0;
    total = 1974;
    for j=1:1974
        if Input(j, i)==0
            count = count+1;
        end
    end
    rate = count / total;
    select_zero = [select_zero; rate];
end

Input_wo_zero = [];
Factor_wo_zero = cell(729,1);
tmp = 1
for k=1:729
    if select_zero(k, 1) < 0.95
        Input_wo_zero = [Input_wo_zero, Input(:,k)];
        Factor_wo_zero{tmp, 1} = Factors{k,1};
        tmp = tmp+1;
    end
end
```

第二步 剔除分子描述符中 PLCC 值较低的元素

```
plcc = [];
for i=1:403
    index = IQAPerformance(Input_wo_zero(:,i), Output, 'p');
    plcc = [plcc;index];
end
count0 = 0;
count1 = 0;
count2 = 0;
count3 = 0;
count4 = 0;
```

```
count5 = 0;
count6 = 0;
for j=1:403
    if plcc(j,1) > 0 & plcc(j,1) < 0.1
        count0 = count0+1;
    end
    if plcc(j,1) > 0.1 & plcc(j,1) < 0.2
        count1 = count1+1;
    end
    if plcc(j,1) > 0.2 & plcc(j,1) < 0.3
        count2 = count2+1;
    end
    if plcc(j,1) > 0.3 & plcc(j,1) < 0.4
        count3 = count3+1;
    end
    if plcc(j,1) > 0.4 & plcc(j,1) < 0.5
        count4 = count4+1;
    end
    if plcc(j,1) > 0.5 & plcc(j,1) < 0.6
        count5 = count5+1;
    end
    if plcc(j,1) > 0.6 & plcc(j,1) < 0.7
        count6 = count6+1;
    end
end
Input_high_plc = [];
Factor_high_plc = cell(403,1);
tmp = 1
for k=1:403
    if plcc(k,1) > 0.3
        Input_high_plc = [Input_high_plc,Input_wo_zero(:,k)];
        Factor_high_plc{tmp,1} = Factor_wo_zero{k,1};
        tmp = tmp+1;
    end
end
```

第三步 确定随机森林的最优叶子节点数与最优树数

```
for RFOptimizationNum=1:5

RFLeaf=[5,10,20,50,100,200,500];
col='rgbcmyk';
```

```
figure('Name','RF Leaves and Trees');
for i=1:length(RFLeaf)

RFModel=TreeBagger(2000,Input_high_plc,Output,'Method','R','OOBPrediction','On','MinLeafSi
ze',RFLeaf(i));
    plot(oobError(RFModel),col(i));
    hold on
end
xlabel('Number of Grown Trees');
ylabel('Mean Squared Error') ;
LeafTreelgd=legend({'5' '10' '20' '50' '100' '200' '500'},'Location','NorthEast');
title(LeafTreelgd,'Number of Leaves');
hold off;

disp(RFOptimizationNum);
end
```

第四步 构建随机森林，对分子描述符按照重要性进行排序

```
%% Cycle Preparation
RFScheduleBar=waitbar(0,'Random Forest is Solving...');
RFRMSEMatrix=[];
RFrAllMatrix=[];
RFRunNumSet=50000;
for RFCycleRun=1:RFRunNumSet

%% Training Set and Test Set Division
RandomNumber=(randperm(length(Output),floor(length(Output)*0.2)))';
TrainYield=Output;
TestYield=zeros(length(RandomNumber),1);
TrainVARI=Input_high_plc;
TestVARI=zeros(length(RandomNumber),size(TrainVARI,2));
for i=1:length(RandomNumber)
    m=RandomNumber(i,1);
    TestYield(i,1)=TrainYield(m,1);
    TestVARI(i,:)=TrainVARI(m,:);
    TrainYield(m,1)=0;
    TrainVARI(m,:)=0;
end
TrainYield(all(TrainYield==0,2),:)=[];
TrainVARI(all(TrainVARI==0,2),:)=[];
```

```
%% RF
nTree=200;
nLeaf=5;
RFModel=TreeBagger(nTree,TrainVARI,TrainYield,...
    'Method','regression','OOBPredictorImportance','on', 'MinLeafSize',nLeaf);
[RFPredictYield,RFPredictConfidenceInterval]=predict(RFModel,TestVARI);
% PredictBC107=cellfun(@str2num,PredictBC107(1:end));

%% Accuracy of RF
RFRMSE=sqrt(sum(sum((RFPredictYield-TestYield).^2))/size(TestYield,1));
RFrMatrix=corrcoef(RFPredictYield,TestYield);
RFr=RFrMatrix(1,2);
RFRMSEMatrix=[RFRMSEMatrix,RFRMSE];
RFrAllMatrix=[RFrAllMatrix,RFr];
if RFRMSE<1000
    disp(RFRMSE);
    break;
end
disp(RFCycleRun);
str=['Random Forest is Solving...',num2str(100*RFCycleRun/RFRunNumSet),'%'];
waitbar(RFCycleRun/RFRunNumSet,RFScheduleBar,str);
end
close(RFScheduleBar);

%% Variable Importance Contrast
VariableImportanceX={};
XNum=1;
for Factor=1:length(Factor_high_plc)
    eval(['VariableImportanceX{1,XNum}="',Factor_high_plc{Factor},'";']);
    XNum=XNum+1;
end

% for i=1:size(Input,2)
%     eval(['VariableImportanceX{1,XNum}="',i,'";']);
%     XNum=XNum+1;
% end

figure('Name','Variable Importance Contrast');
VariableImportanceX=categorical(VariableImportanceX);
bar(VariableImportanceX,RFModel.OOBPermutedPredictorDeltaError)
xtickangle(45);
set(gca, 'XDir','normal')
xlabel('Factor');
ylabel('Importance');
```

```
%% RF Model Storage
RFModelSavePath='F:\math_model\';
save(sprintf('%sRF0410.mat',RFModelSavePath),'nLeaf','nTree',...

'RandomNumber','RFModel','RFPredictConfidenceInterval','RFPredictYield','RFr','RFRMSE',...
    'TestVARI','TestYield','TrainVARI','TrainYield');
```

问题二预测回归关键程序：

```
import scipy.io as scio
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsRegressor
from sklearn.svm import SVR
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import GradientBoostingRegressor
from scipy.stats import spearmanr
from sklearn.metrics import mean_squared_error
from sklearn.preprocessing import StandardScaler #标准化：均值方差

#数据加载
dataFile = r'D:\pycharm\shuxuejianmo\data.mat'
data = scio.loadmat(dataFile)
datanew = data['data']
valueFile = r'D:\pycharm\shuxuejianmo\value.mat'
value = scio.loadmat(valueFile)
valuenew = value['value']
x_train,x_test,y_train,y_test = train_test_split(datanew,valuenew,test_size=0.3,random_state=5)
stdsc1 = StandardScaler()
x_train = stdsc1.fit_transform(x_train)
x_test = stdsc1.transform(x_test)


#模型加载与训练
model1 = KNeighborsRegressor(n_neighbors=8, weights='distance', p=1)
model2 = DecisionTreeRegressor()
model3 = GradientBoostingRegressor(n_estimators=80, max_depth=13, min_samples_split=100)
model1.fit(x_train,y_train)
model2.fit(x_train,y_train)
model3.fit(x_train,y_train)
y_pred1=model1.predict(x_train)
y_pred2=model2.predict(x_train)
y_pred3=model3.predict(x_train)
```

```
y_pred2.resize((len(y_pred2), 1))
y_pred3.resize((len(y_pred2), 1))
y_input1 = np.hstack((y_pred1,y_pred2))
y_input = np.hstack((y_input1,y_pred3))


#模型测试
y_pred1=model1.predict(x_test)
y_pred2=model2.predict(x_test)
y_pred3=model3.predict(x_test)
y_pred2.resize((len(y_pred2), 1))
y_pred3.resize((len(y_pred2), 1))
y_input1 = np.hstack((y_pred1, y_pred2))
x_input = np.hstack((y_input1, y_pred3))

stdsc2 = StandardScaler()
y_input = stdsc2.fit_transform(y_input)
x_input = stdsc2.transform(x_input)


model = SVR(C = 10)
model.fit(y_input, y_train)
y_predtr=model.predict(y_input)
y_pred = model.predict(x_input)
SROCC = spearmanr(y_test, y_pred)
y_pred.resize((len(y_pred),1))
print("训练数据集上的均方根误差:",mean_squared_error(y_train,y_predtr))
print("测试数据集上的均方根误差:",mean_squared_error(y_test,y_pred))
print("训练数据集上的 SROCC:",spearmanr(y_train,y_predtr))
print("测试数据集上的 SROCC:",spearmanr(y_test, y_pred))
```

问题三分类模型主要程序：

分类模型 1 基于 CSP 特征优化 SVM 分类模型程序

CSP 算法程序

```
function CSPMatrix = learnCSP(EEGSignals,classLabels)
nbChannels = size(EEGSignals.x,2);
nbTrials = size(EEGSignals.x,3);
nbClasses = length(classLabels);
if nbClasses ~= 2
    disp('ERROR! CSP can only be used for two classes');
    return;
end
covMatrices = cell(nbClasses,1); %the covariance matrices for each class
%% Computing the normalized covariance matrices for each trial
```

```
trialCov = zeros(nbChannels,nbChannels,nbTrials);
for t=1:nbTrials
    E = EEGSignals.x(:,:,t)';                    %note the transpose
    EE = E * E';
    trialCov(:,:,t) = EE ./ trace(EE);
end
clear E;
clear EE;
%computing the covariance matrix for each class
for c=1:nbClasses
    covMatrices{c} = mean(trialCov(:,:,EEGSignals.y == classLabels(c)),3); %EEGSignals.y==classLabels(c) returns the indeces corresponding to the class labels
end
%the total covariance matrix
covTotal = covMatrices{1} + covMatrices{2};
%whitening transform of total covariance matrix
[Ut Dt] = eig(covTotal); %caution: the eigenvalues are initially in increasing order
eigenvalues = diag(Dt);
[eigenvalues egIndex] = sort(eigenvalues, 'descend');
Ut = Ut(:,egIndex);
P = diag(sqrt(1./eigenvalues)) * Ut';
%transforming covariance matrix of first class using P
transformedCov1 =  P * covMatrices{1} * P';
%EVD of the transformed covariance matrix
[U1 D1] = eig(transformedCov1);
eigenvalues = diag(D1);
[eigenvalues egIndex] = sort(eigenvalues, 'descend');
U1 = U1(:, egIndex);
CSPMatrix = U1' * P;
function features = extractCSP(EEGSignals, CSPMatrix, nbFilterPairs)
nbTrials = size(EEGSignals.x,3);
features = zeros(nbTrials, 2*nbFilterPairs+1);
Filter = CSPMatrix([1:nbFilterPairs (end-nbFilterPairs+1):end],:);
%extracting the CSP features from each trial
for t=1:nbTrials
    %projecting the data onto the CSP filters
    projectedTrial = Filter * EEGSignals.x(:,:,t)';
    %generating the features as the log variance of the projected signals
    variances = var(projectedTrial,0,2);
    for f=1:length(variances)
        features(t,f) = log(1+variances(f));
    end
end
```

SVM 算法程序

```
model = svmtrain(l1(1:1974,:) , dataCSP1(1:1974,:), '-c 2 -g 1');
[predict_label, accuracy,decision_values] = svmpredict(l1(1975:end,:), dataCSP1(1975:end,:), model);
```

分类模型 2 基于自注意力多尺度卷积的 MLP 分类模型

网络框架程序

```
class ScaledDotProductAttention(nn.Module):
    ''' Scaled Dot-Product Attention '''
    def __init__(self, temperature, attn_dropout=0.1):
        super().__init__()
        self.temperature = temperature
        self.dropout = nn.Dropout(attn_dropout)
    def forward(self, q, k, v):
        attn = torch.matmul(q.transpose(2, 3) / self.temperature, k)
        attn = self.dropout(F.softmax(attn, dim=-1))
        output = torch.matmul(v,attn)  #
        return output
class Tblock3(nn.Module):    ###256 采样
    def __init__(self,  input_size, num_T):
        super(Tblock3, self).__init__()
        self.Tception = nn.Sequential(
            nn.Conv2d(input_size[0],  input_size[0],  kernel_size=(1,  1),  stride=(1,  1), padding=0),
            nn.ReLU(), )
        size = self.get_size(input_size)
        self.attention = ScaledDotProductAttention(temperature=size[3] ** 0.5)
        self.Tception1 = nn.Sequential(  #63--->28
            nn.Conv2d(input_size[0], num_T, kernel_size=(1, 7), stride=(1, 1), padding=0),
            nn.ReLU(),
            nn.AvgPool2d(kernel_size=(1, 2), stride=(1, 2)))

        self.Tception3 = nn.Sequential(    #63--->30
            nn.Conv2d(input_size[0], num_T, kernel_size=(1, 128), stride=(1, 1), padding=0),
            nn.ReLU(),
            nn.AvgPool2d(kernel_size=(1, 2), stride=(1, 2)))
        self.Tception4 = nn.Sequential(
            nn.Conv2d(input_size[0], num_T, kernel_size=(1, 400), stride=1, padding=0),
            nn.ReLU(),
            )
        self.BN_t = nn.BatchNorm2d(num_T)  # 进行数据的归一化处理
    def forward(self, x):
        input = self.Tception(x)
```

```
            q,k,v=input,input,input
            out=self.attention(q,k,v)
            input = out+input
            y = self.Tception1(input)
            out = y
            y = self.Tception3(input)
            out = torch.cat((out, y), dim=3)  # 行连接
            y = self.Tception4(input)
            out = torch.cat((out, y), dim=3)
            out = self.BN_t(out)
            return out
        def get_size(self, input_size):  ##加权个数
            data = torch.ones((1, input_size[0], input_size[1], input_size[2]))
            y = self.Tception(data)
            out = y
            #print(out.size())
            return out.size()
class TSCN(nn.Module):
        def __init__(self, num_classes, input_size, num_T, num_S,
                         hiden,dropout_rate):
            super(TSCN, self).__init__()
            self.Get_timefeatures = Tblock3(input_size=input_size, num_T=num_T, )  ###为了得
到时间特征的个数 Tsize
            self.BN_s = nn.BatchNorm2d(num_S)
            self.Sception1 = nn.Sequential(
                nn.Conv2d(num_T, num_S, kernel_size=(1, 1), stride=1, padding=0),
                nn.ReLU(),
                nn.AvgPool2d(kernel_size=(1, 1), stride=(1, 1)))
            size = self.get_size(input_size)
            self.fc1 = nn.Sequential(
                nn.Linear(size[1], hiden),
                nn.ReLU(),
                nn.Dropout(dropout_rate))
            self.fc2 = nn.Sequential(
                nn.Linear(hiden, num_classes),
                # nn.Softmax())
                nn.LogSoftmax())
        def forward(self,X ):
            out = self.Get_timefeatures(X)  ### B*9*30*Tfetures
            z = self.Sception1(out)
            out_final = z
            out = self.BN_s(out_final)
            out = out.view(out.size()[0], -1)
            out= self.fc1(out)
```

```
        out=self.fc2(out)
        return out
    def get_size(self, input_size):
        data = torch.ones((1, input_size[0], input_size[1], input_size[2]))
        out=self.Get_timefeatures(data)
        z = self.Sception1(out)
        out_final = z
        out = self.BN_s(out_final)
        out = out.view(out.size()[0], -1)
        return out.size()
```