基于 Stacking 的序列后向搜索特征选择方法

王海荣*，徐贞顺，林淑飞

北方民族大学计算机科学与工程学院，宁夏银川 750021

摘要： 针对数据处理中存在的噪音以及不相关特征过滤问题，提出了一种基于 Stacking 框架的序列后向搜索特征选择方法。使用 K-Fold 交叉验证方式训练并保存 DNN、SVM 基学习器，基学习器预测结果作为元学习器输入，训练并保存逻辑回归学习模型；综合分析全连接神经网络权重矩阵、支持向量机相关系数，根据元学习器模型学习结果为各基学习器赋予不同权重，计算各特征影响因子并调用序列后向搜索算法（SBS）生成最优特征子集。实验阶段，基于 Kaggle 网站上心脏病研究公开数据集构建了一个疾病诊断模型，调用 Stacking-SBS 生成特征空间中最优特征子集，进行特征选择前后诊断模型性能对比实验，将该方法与信息增益（IG）、卡方检验(Chi)和基于相关性的特征选择方法(CFS)进行对比，结果表明应用该方法不仅能够减少模型训练时间，模型的召回率、F1 值也得到明显提升。此外，该方法在性能提升方面明显优于其他三种特征选择方法。最后使用 Kaggle 网站上心血管研究公开数据集来验证 Stacking-SBS 的泛化能力，实验结果表明该方法也可显著提升疾病诊断模型性能。

关键词： 特征选择; Stacking 框架; K-Fold 交叉验证; 序列后向搜索

DOI: 10.57237/j.cst.2022.01.001

Sequential Backward Feature Selection Method Based on Stacking Framework

Wang Hairong*, Xu Zhenshun, Ling Shufei

School of Computer Science and Engineering, North Minzu University, Yinchuan 750021, China

Abstract: Aiming at the problems of noise and irrelevant feature filtering in data processing, a feature selection method based on the stacking framework is proposed. Use the K-Fold cross-validation method to train and save DNN and SVM-based learners. The prediction results of the base learners are used as the input of the meta-learner, and the logistic regression learning model is trained and saved; comprehensively analyze the correlation coefficients of the fully connected neural network weight matrix and support vector machine, According to the learning results of the meta-learner model, assign different weights to each base learner, calculate the influence factors of each feature, and call the sequence backward search algorithm (SBS) to generate the optimal feature subset. In the experimental stage, a disease diagnosis model was constructed based on the open data set of heart disease research on the Kaggle website, and Stacking-SBS was called to generate the optimal feature subset in the feature space, and the performance comparison experiment of the diagnostic model before and after feature selection was performed, and the method was improved with

基金项目：北方民族大学重大教育教学改革项目 (2021 年) 和北方民族大学科研项目 (2021XYZJK06).

*通信作者：王海荣, bmdwhr@163.com

收稿日期: 2022-08-09; 接受日期: 2022-09-15; 在线出版日期: 2022-11-01 http://www.computscitech.com

information. (IG), Chi-square test (Chi) and correlation-based feature selection method (CFS) are compared. The results show that the application of this method can not only reduce model training time, but also significantly improve the model's recall rate and F1 value. In addition, this method is significantly better than the other three feature selection methods in terms of performance improvement. Finally, the open data set of cardiovascular research on the Kaggle website is used to verify the generalization ability of Stacking-SBS. The experimental results show that this method can also significantly improve the performance of the disease diagnosis model.

Keywords: Feature Selection; Stacking Framework; K-Fold Cross-Validation; Sequence Backward Search

1 引言

特征选择是数据预处理的重要手段之一，是通过计算源数据中每个特征对最终模型输出结果的影响因子来进行特征选择与过滤，该方法主要应用于数据高维特征空间的降维处理，解决“维度灾难”问题。由于在众多研究领域的模型训练中，可以通过特征选择降低源数据高维信息的语义矩阵维度从而减少模型复杂度，达到缩短模型训练时间、降低训练成本的目的，因此，特征选择算法研究在学术界和行业得到广泛关注。传统特征选择方法主要包括主成分分析法(PCA)[1]、TF-IDF[2, 3]、互信息[4]等。丁雪梅等[5]使用调整的余弦相似度来度量特征间的相关性，提出一种基于 Relief 的无监督特征选择方法。高宝林等[6]引入类内和类间分布因子来降低特征词在类间均匀分布时对分类带来的负贡献并将其应用微博情感分析。周传华等[7]特征相关性和分类能力两个方面对特征进行综合度量，调用序列前向选择来删除冗余特征并用实验证明其有效性。胡峰等[8]动态地将原始特征集划分为若干个特征子空间，提出了一种基于特征聚类的封装式特征选择算法并证明该算法可提升分类器性能。陈谌等[9]提出一种基于随机森林 Gini 指标和卡方检验的最优特征子集的特征选择方法并应用于支持向量机[10]算法模型中，解决了传统机器学习分类算法在非平衡数据集上准确率降低的问题。雷海锐等[11]提出一种基于 filter-wrapper 模型的混合式特征选择方法并通过实验证明了该方法选择的特征子集具有更好的分类能力。Chen 等[12]针对高维数据提出一种 SFR 特征选择方法，该方法首先进行子空间特征聚类来判别每个特征对每个类别重要性，然后使用分层特征加权方法对特征排序。Kewen Li 等[13]针对正负样本不均衡数据集提出一种加权互信息的 WMI 特征选择方法，该方法使用模糊 C 均值聚类为样本分配不同权重，根据权重计算互信息，最后用 NASA 四个不均衡数据来验证 WMI 方法有效性。

综上，现有特征选择算法通过分析单个特征信息增益（IG）、平均下降指数（Gini）等指标来衡量该特征与学习目标的相关性，根据相关性大小来过滤冗余特征，没有考虑模型训练时源数据高维语义矩阵线性变换和非线性变换过程中不同维度间相互影响的关系。本文结合 Stacking 学习模型能够融合多个机器学习模型的优势，提出的基于 Stacking 的序列后向搜索的特征选择算法，通过综合分析多个学习模型训练过程中生成的学习参数并做加权处理，能够更细粒度地提取和分析源数据特征空间中每个特征影响因子大小，在本文搭建的疾病诊断模型训练中得到了较好的体现。

我们将在第 2 节概述特征选择方法的相关理论及方法；第 3 节详细阐述本文提出的特征选择方法；第 4 节详细介绍特征选择过程及实验结果分析；最后是对本文的工作总结。

2 特征选择方法相关理论

特征选择是指将高维空间的样本通过映射或者是变换的方式转换到低维空间，以避免“维度灾难”问题。首先从特征全集开始按搜索策略产生一个特征子集，运用评价函数对该特征子集进行评估，将评价结果与停止准则进行比较，如果满足则输出最优特征子集，否则产生下一组特征子集继续进行特征选择，特征选择流程如图 1 所示。

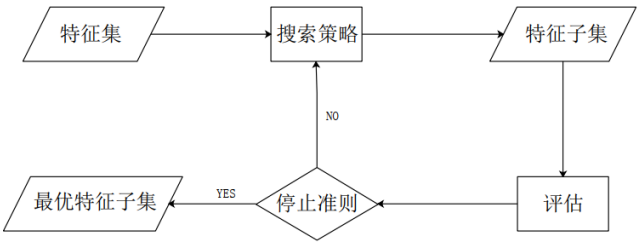

图 1 特征选择流程

本文通过构建并综合分析 Stacking 框架学习参数对原数据特征空间各维度进行重要性度量，采用序列后向搜索（SBS）策略生成最优特征子集，Stacking 模型各基学习器采用 K-Fold 交叉验证思想进行训练。

2.1 K-Fold 交叉验证

K-Fold 交叉验证（k-fold cross validation）是机器学习划分数据集和验证集的一种方法，可以从有限的数据中集中获得尽可能多的数据信息，在一定程度上减少过拟合。其主要思想为：将全部数据集分成 k 个不相交的子集，即 K 折。假设 S 中的训练样例个数为 m，那么每一个子集有 m/k 个训练样例，相应的子集称作{s1,s2,…,sk}；每次从分好的子集中取一个作为测试集，其它 k-1 个作为训练集；取 k 次模型在测试集上精确率的平均值作为该模型 k 折交叉验证的精确率。

2.2 Stacking 学习模型

Stacking 是一种分层模型集成框架，由基学习器（base-learner）和元学习器（meta-learner）组成。第一层使用 K-Fold 方法交叉训练多个基学习器，第二层元学习器则以第一层基学习器预测结果构造新的特征空间进行再训练，从而构建完整的 Stacking 模型，Stacking 框架如图 2 所示。

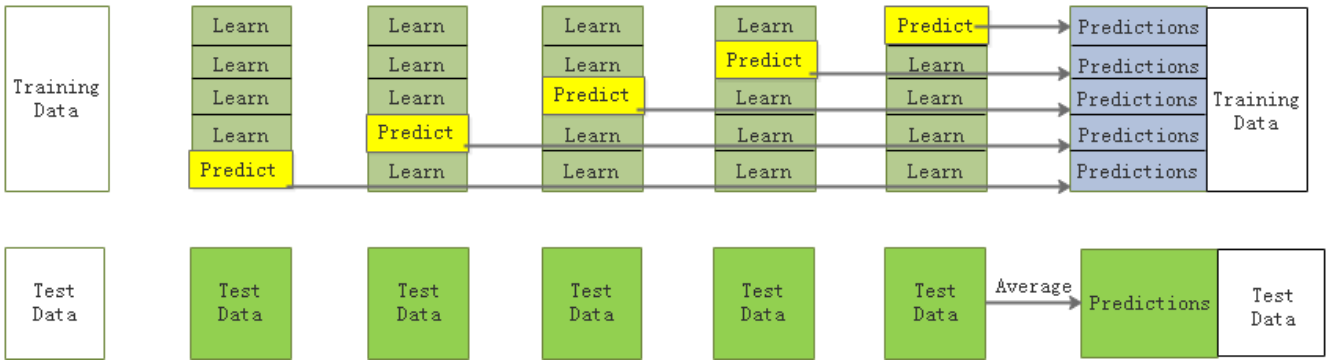

图 2 Stacking 框架

2.3 序列后向搜索（SBS）

序列后向搜索是一种启发式的特征子空间搜索算法，从特征全集开始，每次从特征集删除一个特征，重复该过程使得删除特征后评价函数值达到最优，本文中评价函数使用分类器精确度。具体过程如算法 1 所示(N 为原数据特征空间中属性个数)：

算法 1 序列后向搜索算法

输入：Feature set F;Max_Accuracy /* 特征全集 F; 最大精确率（初始值为 0）*/;

输出：Max_Accuracy, Max_Accuracy corresponds to the optimal feature subset /*最大精确率, 最优特征子集*/;

1. for t =1, 2,…, N-1
2. Train the classifier on the feature set F;
3. Classify on the validation set, calculate the classification accuracy rate Acc for each iteration/*每次迭代过程中计算分类器精确率*/;
4. if (Acc> Max_Accuracy)

Then Max_Accuracy = Acc;

5. Delete a feature in the feature set to get a new feature set F;
6. end for

序列后向搜索的输入是特征空间全集和最大准确率。在迭代过程中，如果将其中一个特征删除后，评价函数值优于删除前，则删除该冗余特征，算法输出为最大精确率和与之对应的最优子集。

3 Stacking-SBS 特征选择方法

本文基于 Stacking 设计特征选择算法，分别分析 Stacking 框架中基学习器学习参数，得到各特征因子大小并加权求和，得到最终特征重要性度量值，然后调用序列后向搜索算法生成最优特征子集。Stacking-SBS 特征选择方法分为两个模块，即构建 Stacking 框架学习模型模块和特征选择模块，Stacking-SBS 方法框架如图 3 所示。

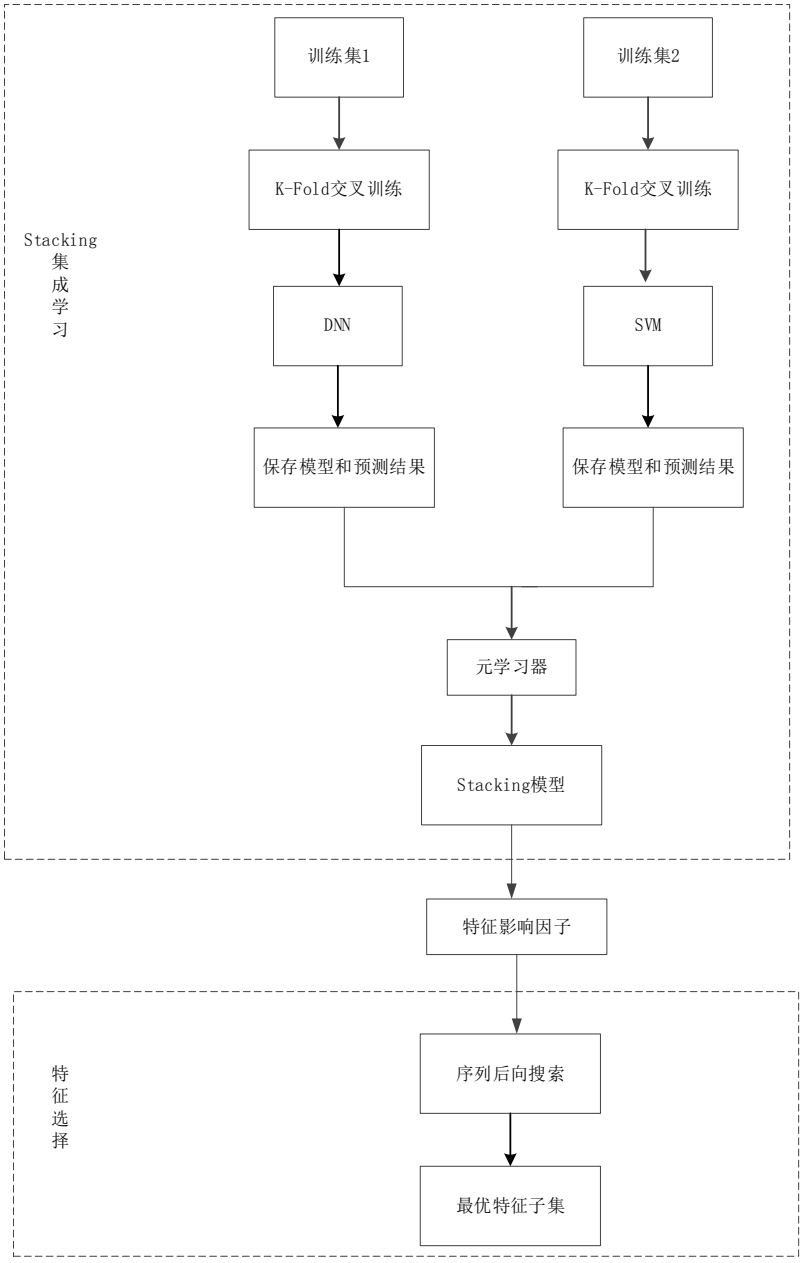

图 3 Stacking-SBS 方法框架

3.1 构建 Stacking 框架学习模型

使用 K-Fold 交叉验证方式构建 Stacking 模型，其基学习器采用全连接神经网络（DNN）、支持向量机（SVM），元学习器采用逻辑回归（LR）。具体流程描述如算法 2 所示。

算法 2 构建 Stacking 框架算法

输入：D = {(x_1, y_1), (x_2, y_2),…, (x_m, y_m)}, Base-Learner:L_1, L_2, L_3,Meta-Learner: L; /* D 为训练集,Base-Learner 为基学习器，Meta-Learner 为元学习器*/;

输出：$H(x) = h'(h_1(x), h_2(x), \ldots, h_T(x))$;

/*训练 Stacking 学习模型*/;

1. for t = 1,2,…,T do
2. $h_t = L_1(D)$ /*训练基学习器*/;
3. end for
4. D' = ∅
5. for i = 1,2,…,m do
6. for t = 1,2,…,T do

7. $z_{it} = h_t(x_i)$
8. end for
9. D’ = D’ ∪ ((z_{i1},z_{i2},…,z_{it}), y_i) /*重新构造特征空间*/;
10. end for
11. h’ = L(D’); /*训练元学习器*/;

Stacking 集成学习框架第一层使用 K-Fold 交叉验证方式训练 DNN 和 SVM 基学习器，并在测试集上进行预测，重新构造预测结果的特征空间并作为元学习器的输入，训练并保存 LR 模型，从而构建完整的 Stacking 模型。

3.2 特征选择模块

首先构建 Stacking 框架模型，然后分别加载各学习器并分析训练参数，进行特征重要性度量，分析 DNN 模型生成的权重矩阵和 SVM 模型特征系数，根据元学习器 LR 模型保存的系数矩阵为各基学习器赋予权重，将基学习器各特征影响因子加权求和，得到该特征最终的影响因子大小，采用序列后向搜索算法（SBS）生成最优特征子集。特征选择框架如图 4 所示。

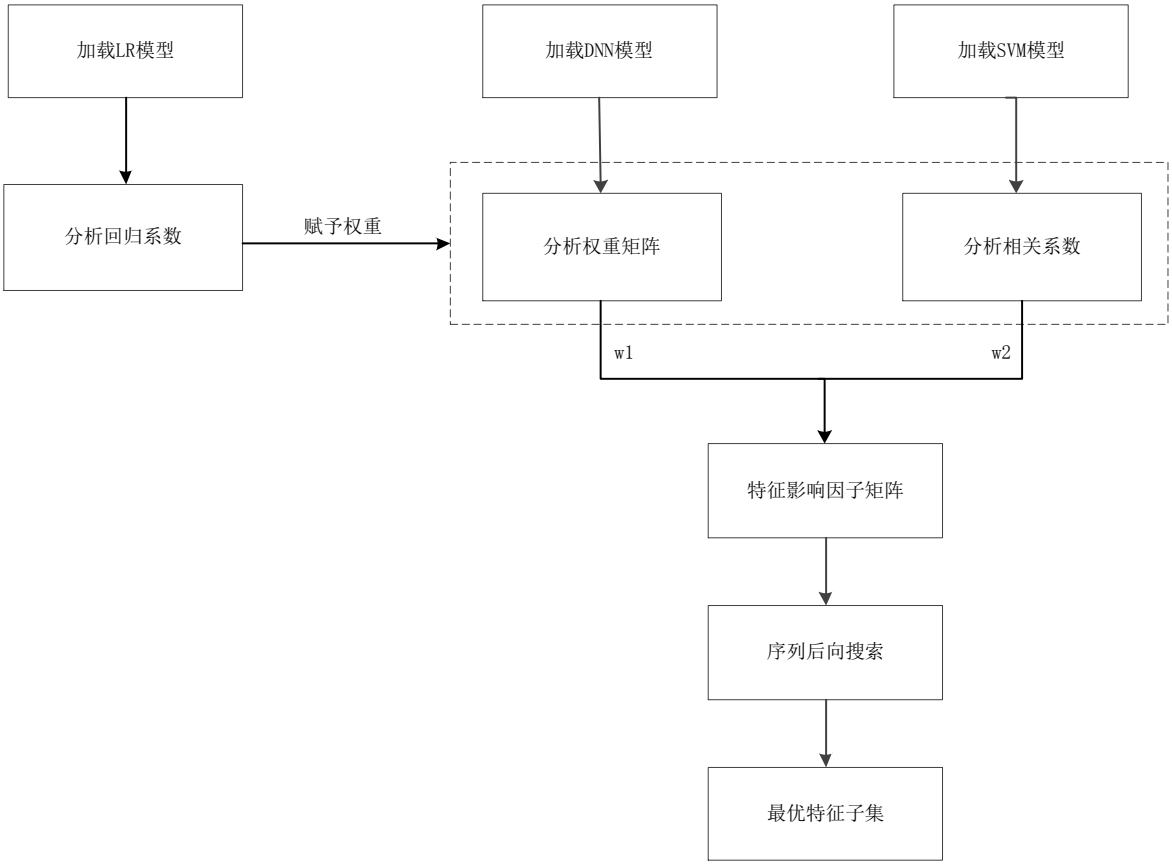

图 4 特征选择框架

特征选择模块通过分析 Stacking 模型中的学习参数，将其加权求和得到特征空间中特征影响因子矩阵，然后调用序列后向搜索算法，从特征全集开始，依次删除特征影响因子最小的一个特征，生成新的特征空间并重新构建 Stacking 模型，通过迭代输出分类器中全局最高精确率和最优特征子集。

4 方法验证及结果分析

本文通过构建 Stacking 学习模型并综合分析训练过程中保存的学习参数，得到特征空间中各维度影响因子，调用序列后向搜索方法生成最优特征子集，用最优特征子集重新构建 Stacking 模型并进行特征选择前后性能对比分析。

*实验数据：*采用网上某镇居民心脏病研究公开数据集，原始数据特征空间包含患者年龄、教育程度、血糖值、心率、BMI 等 15 种特征属性，分类目标是预测患者未来 10 年内是否有患冠心病（CHD）风险。为便于实验分析，本实验调用 sklearn 和 pandas 相关库对

数据进行清洗，分析各特征维度数据缺失情况并进行填充，然后对数据进行标准化并存储为 numpy 文件。

*实验环境：*本文在 Windows 环境下应用 Python、pycharm 等语言工具构建基于 Stacking 框架疾病诊断模型并调用本文提出的特征选择算法进行对比分析。

*方法评价：*使用精确率（precision，见公式 1）、召回率（recall，见公式 2）、F1 值（见公式 3）3 个指标来评估诊断模型性能。

$$\text{precision} = \frac{TP}{TP+FP} \tag{1}$$

$$\text{recall} = \frac{TP}{TP+FN} \tag{2}$$

$$F1 = \frac{2TP}{2TP+FP+FN} \tag{3}$$

其中，TP、FP、FN 分别为真正例、假正例、真反例。

本文采用心脏病研究公开数据集构建基于 Stacking 框架疾病诊断模型，Stacking 模型基学习器采用全连接神经网络（DNN）、支持向量机（SVM），元学习器采用逻辑回归（LR），K-Fold 交叉验证过程 K 设置为 5。DNN 和 SVM 在测试集上性能评估结果如表 1 所示，Stacking 框架学习模型与基学习器性能对比如表 2 所列（基学习器 5 次性能评估结果均值作为该学习器最终性能评估结果）。

表 1 Stacking 框架基学习器 5 折交叉验证过程性能

基学习器	评价指标					
DNN	pre	0.618	0.730	0.670	0.692	0.625
	recall	0.741	0.548	0.739	0.692	0.659
	F1	0.674	0.626	0.703	0.692	0.642
SVM	pre	0.647	0.631	0.705	0.628	0.714
	recall	0.611	0.654	0.667	0.678	0.657
	F1	0.629	0.642	0.685	0.652	0.684

表 2 Stacking 框架与基学习器模型性能对比

Model	precision	recall	F1
DNN	0.66688	0.67564	0.66712
SVM	0.66492	0.65338	0.65844
Stacking	0.7316	0.6881	0.7092

对比分析表 1 和表 2 可知，与各基学习器独立训练结果相比，基于 Stacking 框架的疾病诊断模型 precision 提升了 7%左右，recall 最多提升了 3%，F1 值最多提升了 5%左右，进而说明 Stacking 学习框架能够显著提升疾病诊断模型性能。

Stacking 模型构建完成后，调用 pickle 库加载 K-Fold 交叉验证过程中生成的各基学习器模型，综合分析各基学习器训练过程中生成的学习参数并将其归一化，基学习器各特征影响因子加权求和得到该特征最终特征影响因子，各特征影响因子如图 5 所示，其中横坐标代表特征序号，纵坐标代表该特征影响因子。

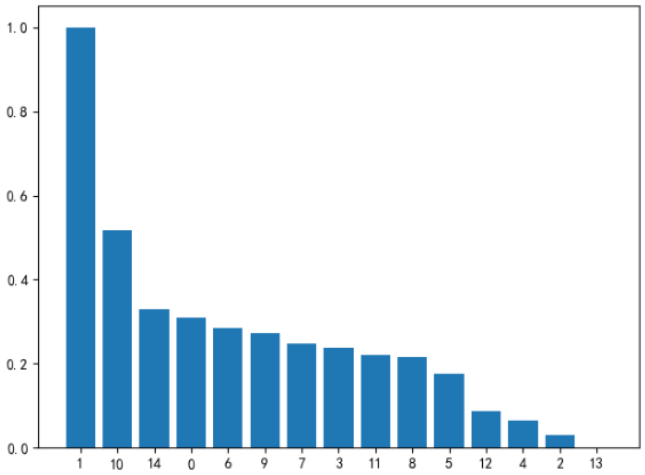

图 5 各特征维度影响因子

得到特征空间中各维度特征影响因子后，调用序列后向搜索（SBS）算法，依次删除影响因子最低的一个特征并生成最优特征子集。本实验的特征选择结果如表 3 所示，特征选择前后 Stacking 模型诊断性能对比如表 4 所示。

表 3 特征选择结果

过滤特征	BMI、每天吸烟数量、学历程度
最优特征子集	性别、年龄、是否吸烟、是否患有高血压、有无吸烟史、有无高压史、是否患有糖尿病、总胆固醇水平、收缩压、舒张压、血糖值、心率

表 4 特征选择前后 Stacking 模型诊断性能

	precision	recall	F1	time
Before	0.7316	0.6881	0.7092	25s
After	0.735	0.7277	0.7313	21s

分析表 4 可知，使用最优特征子集构建的疾病诊断模型 recall 和 F1 指标均有提升，其中 recall 提升了 4%，F1 值提升了 3%。此外，特征选择后训练 Stacking 模型时间约减少了 16%，这说明本文提出的特征选择算法不仅能够提升模型性能，也能够减少模型运行时间成本。

为进一步验证本文方法的有效性，本文基于 Kaggle 网站上心脏病研究公开数集，将本文方法与 IG、Chi、CFS 三种特征选择方法进行对比分析，其实验结果如表 5 所示。

表 5 特征选择方法实验对比结果

特征选择方法	precision	recall	F1
IG	0.7528	0.6634	0.6702
Chi	0.6715	0.6881	0.6797
CFS	0.6779	0.698	0.6878
Stacking-SBS	0.735	0.7277	0.7313

表 5 分析结果可知，本文方法的 precision 较 Chi2 和 CFS 均提升了约 6%；recall 较 IG 提升了 6%，较 CHI2 提升了 4%，较 CFS 提升了约 3%；F1 值较 IG 提升了约 6%，较 CHI2 提升了约 5%，较 CFS 提升了约 4%。对比实验结果表明了本文方法的有效性。

此外，为验证本文方法的泛化能力，在公开的心血管研究数据集（Kaggle 网站）上进行实验，其结果如表 6 所示。

表 6 特征选择前后结果对比

	precision	recall	F1
Before	0.7011	0.6948	0.6979
After	0.705	0.7572	0.7302

从表 6 可以看出，本文方法删除冗余特征后模型的 recall 提升了 6%，F1 值提升了 3%，实验结果表明本文提出的特征选择方法具有较好的泛化能力。

5 总结

特征选择算法能够消除源数据中冗余信息，生成最优特征子集，是当前大数据挖掘的研究热点。本文研究了当前主流特征选择算法及相关技术，提出一种基于 Stacking 框架和序列后向搜索特征选择方法，该方法使用 K-Fold 交叉验证方法构建 Stacking 框架学习模型，综合分析各学习器训练过程中生成的学习参数矩阵，加权求和得到各特征影响因子，调用序列后向搜索依次删除影响因子最小的一个特征并生成最优特征子集。为验证提出的方法，本文面向医学领域，采用心血管疾病数据集构建 Stacking 框架疾病诊断模型，调用本文所提出的特征选择方法生成最优特征子集并对比分析特征选择前后疾病诊断模型性能。此外，将本文方法与 IG、Chi、CFS 三种特征选择方法进行了实验对比，实验结果表明本文方法的精准度、召回率、F1 值优于对比的三种方法且其泛化能力更好。

综上，本文方法较好地解决训练数据集维度高、冗余或无关特征过多等问题，可广泛应用于基于机器学习进行模型训练的场景，达到降维、提高效率的目的，以辅助垂直领域，如医学、农业等复杂场景的模型训练。

参考文献

[1] Nasution M Z F, Sitompul O S, Ramli M. PCA based feature reduction to improve the accuracy of decision tree c4.5 classification [J]. Journal of Physics Conference, 2018, 978 (2018): 1-6.

[2] Xuemei Y E, Xuemin M, Jinchun X, et al. Improved Approach to TF-IDF Algorithm in Text Classification [J]. Computer Era, 2019.

[3] Guo A, Yang T. Research and improvement of feature words weight based on TFIDF algorithm [C] // 2016 IEEE Information Technology, Networking, Electronic and Automation Control Conference (ITNEC). IEEE, 2016.

[4] B Z S A, B J Z A, B L D A, et al. Mutual information based multi-label feature selection via constrained convex optimization [J]. Neurocomputing, 2019, 329: 447-456.

[5] Ding Xuemei, Wang Hanjun, Wang Yangguang, et al. Unsupervised feature selection method based on improved ReliefF [J]. Computer Systems & Applications, 2018, 27 (003): 149-155.

[6] Gao Baolin, Zhou Zhiguo, Yang Wenwei, et al. Feature selection method based on the combination of category and improved CHI[J]. Application Research of Computers, 2018, 035 (006): 1660-1662.

[7] Zhou Chuanhua, Liu Zhicai, Ding Jingan, et al. Feature selection algorithm based on filter+wrapper mode [J]. Application Research of Computers, 2019, 036 (007): 1975-1979.

[8] Hu Feng, Yang Meng. Packaging feature selection algorithm based on feature clustering [J]. Computer Engineering and Design, 2018, 039 (001): 230-237.

[9] Chen Chen, Liang Xuechun. Feature selection method based on Gini index and chi-square test [J]. Computer Engineering and Design, 2019 (08): 2342-2345.

[10] Huang S, Cai N, Pacheco P P, et al. Applications of Support Vector Machine (SVM) Learning in Cancer Genomics [J]. Cancer Genomics & Proteomics, 2018, 15 (1): 41-51.

[11] Lei Hairui, Gao Xiufeng, Liu Hui. Hybrid feature selection algorithm based on machine learning [J]. Electronic Measurement Technology, 2018, 41 (16): 42-46.

[12] Chen R, Sun N, Chen X, et al. Supervised Feature Selection With a Stratified Feature Weighting Method [J]. IEEE Access, 2018: 15087-15098.

[13] Li K, Yu M, Liu L, et al. Feature Selection Method Based on Weighted Mutual Information for Imbalanced Data [J]. International Journal of Software Engineering and Knowledge Engineering, 2018, 28 (8): 1177-1194