软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2011,22(4):709−721 [doi: 10.3724/SP.J.1001.2011.03752]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

基于 FP-Tree 的快速选择性集成算法*

赵强利, 蒋艳凰+, 徐 明

(国防科学技术大学 计算机学院,湖南 长沙 410073)

Fast Ensemble Pruning Algorithm Based on FP-Tree

ZHAO Qiang-Li, JIANG Yan-Huang+, XU Ming

(School of Computer, National University of Defense Technology, Changsha 410073, China)

+ Corresponding author: E-mail: yhjiang@nudt.edu.cn, http://www.nudt.edu.cn

Zhao QL, Jiang YH, Xu M. Fast ensemble pruning algorithm based on FP-Tree. *Journal of Software*, 2011, 22(4):709−721. http://www.jos.org.cn/1000-9825/3752.htm

Abstract: By selecting parts of base classifiers to combine, ensemble pruning aims to achieve a better generalization and have less prediction time than the ensemble of all base classifiers. While, most of the ensemble pruning algorithms in literature consume much time for classifiers selection. This paper presents a fast ensemble pruning approach: CPM-EP (coverage based pattern mining for ensemble pruning). The algorithm converts an ensemble pruning task into a transaction database process, where the prediction results of all base classifiers for the validation set are organized as a transaction database. For each possible size *k*, CPM-EP obtains a refined transaction database and builds a FP-Tree to compact it. Next, CPM-EP selects an ensemble of size *k*. Among the obtained ensembles of all different sizes, the one with the best predictive accuracy for the validation set is output. Experimental results show that CPM-EP reduces computational overhead considerably. The selection time of CPM-EP is about 1/19 that of GASEN and 1/8 that of Forward Selection. Additionally, this approach achieves the best generalization, and the size of the pruned result is small.

Key words: ensemble learning; ensemble pruning; frequent pattern tree (FP-tree); bootstrap aggregation (Bagging); back-propagation neural network (BPNN)

摘 要: 选择性集成通过选择部分基分类器参与集成,从而提高集成分类器的泛化能力,降低预测开销.但已有的选择性集成算法普遍耗时较长,将数据挖掘的技术应用于选择性集成,提出一种基于 FP-Tree(frequent pattern tree)的快速选择性集成算法:CPM-EP(coverage based pattern mining for ensemble pruning).该算法将基分类器对校验样本集的分类结果组织成一个事务数据库,从而使选择性集成问题可转化为对事务数据集的处理问题.针对所有可能的集成分类器大小,CPM-EP 算法首先得到一个精简的事务数据库,并创建一棵 FP-Tree 树保存其内容;然后,基于该 FP-Tree 获得相应大小的集成分类器.在获得的所有集成分类器中,对校验样本集预测精度最高的集成分类器即为算法的输出.实验结果表明,CPM-EP 算法以很低的计算开销获得优越的泛化能力,其分类器选择时间约为 GASEN 的 1/19 以及 Forward-Selection 的 1/8,其泛化能力显著优于参与比较的其他方法,而且产生的集成分类器具有较少的基分类器.

* 基金项目: 国家自然科学基金(60773017, 60905032)
收稿时间: 2009-02-14; 修改时间: 2009-05-21; 定稿时间: 2009-10-19

关键词: 集成学习;选择性集成;频繁模式树;Bagging;误差反向传播神经网络
中图法分类号: TP181 文献标识码: A

机器学习常用于解决分类问题,通过对训练数据的学习,获取其中蕴涵的规律,然后利用学习的结果(分类器)对新数据进行类别预测[1,2].一个好的机器学习系统必须具有较强的泛化能力,也就是说,根据已有数据建立的分类器应该能够很好地处理新的数据.泛化能力一直是评价机器学习系统最重要的标准.集成学习(ensemble learning)[3]在对训练样本的学习过程中获取若干分类器(称为基分类器),然后对这些分类器进行某种方式的组合,共同解决同一个学习任务.集成学习可以分为两大阶段:一是构造基分类器,二是对这些基分类器的预测结果进行组合.对于基分类器的构造,常用的方法有 Bagging[4],Boosting[5],交叉验证[6]以及纠错输出编码(error-correcting output code,简称 ECOC)[7,8]等;对于分类器组合方式,传统的方法有多数投票法[9]、权重投票法[10]、分层组合法[11]等.这些研究结果均表明,集成学习可以有效地提高泛化能力.

选择性集成(ensemble pruning or selective ensemble)的思想是:从众多基分类器中选择一部分进行集成,期望获得优于集成所有的基分类器的效果[12].近年来,选择性集成得到了研究者们的日益关注.文献[13]指出,选择性集成具有两方面的优势:(1) 提高泛化能力.由于基分类器中可能存在预测能力较低的分类器,它们对集成分类器的预测能力具有负面影响,剔除这些基分类器能够提高集成分类器的预测精度;(2) 降低预测阶段的开销.选择部分基分类器进行集成,可以减少所需的存储空间,并降低预测阶段的运算量.但是目前大部分已有的选择性集成算法由于分类器选择所需时间较长,难以应用到在线学习等实时性要求较高的领域.

本文将数据挖掘的思想应用到选择性集成,提出一种高效选择性集成算法:CPM-EP(coverage based pattern mining for ensemble pruning).该算法将所有基分类器对校验样本集的分类结果用一个预测结果表来表示,表中的每一行对应着校验样本集中的一个样本,其中保存着对该样本分类正确的基分类器标识.将预测结果表中的每一行类比于购物篮问题中的一个事务[14],则分类器选择问题可以转化为一个对事务数据库的处理问题.如果希望选择 k 个基分类器,则 CPM-EP 首先获得一个新的事务数据库,其所有事务的长度均为$\lfloor k/2\rfloor+1$,并在此基础上创建一棵 FP-Tree 树,以便获取统计信息和压缩存储所需的内存空间;然后,将其中的路径信息保存为路径表 Path-Table;最后,通过对 Path-Table 的操作获取给定 k 值的选择结果.在所有可能的 k 值所对应的选择结果中,对校验样本集的预测精度最高的分类器选择结果即为 CPM-EP 算法的输出.实验结果表明,与 Select-Best[15],Bagging[4],GASEN[12]和 Forward-Selection[16−18]算法相比,CPM-EP 算法以很小的计算开销获得了优越的泛化能力,且选择的基分类器数据较少,预测开销较低.

本文第 1 节介绍相关概念.第 2 节对 CPM-EP 算法进行详细阐述.第 3 节讲述实验数据、比较算法和实验方法.第 4 节对实验结果进行分析.第 5 节总结全文.

1 CPM-EP 算法

1.1 算法思想

CPM-EP 算法的主要思想是,将分类器选择问题转换为对事务数据集的操作问题,然后利用 FP-Tree 来加快统计和数据检索的速度,并减少统计信息时所需要占用的内存空间.这对于内存紧张以及大型应用的场合是非常重要的.

在该算法中,所有基分类器对校验样本集中各样本的分类结果保存在预测结果表 T 中,T 的每一行保存着一个样本的标识号以及对这个样本预测正确的基分类器的标识集合.如果将表 T 的每一行类比于购物篮问题中的一个事务,保存的分类器标识则对应着购买的商品标识(项),那么预测结果表 T 就可以类比于一个事务数据库.根据多数投票的原理,如果给定集成分类器大小 k,则长度小于$\lfloor k/2\rfloor+1$ 的事务所对应的样本将无法被正确分类.因此,我们首先对原事务数据库进行精简,剔除不必要的分类结果,然后创建一棵 FP-Tree 树,最后利用贪婪算法获得对应的集成分类器.CPM-EP 算法在所有可能的 k 值对应的选择性集成结果中选取对校验样本集预测精度最高者作为最终的输出.

假设通过某种学习策略对训练样本集 LS 进行学习,已经获得 L 个基分类器,现在利用校验样本集 VS 从这 L 个分类器中选择若干进行多数投票法集成.令 VS 中的样本数目为 n,对应的样本分别为 $X_1,X_2,\ldots,X_n$,L 个基分类器分别为 $h_1,h_2,\ldots,h_L$,则 CPM-EP 算法的伪代码如算法 1 所示.

算法 1. CPM-EP 算法.

```
CPM-EP(BC,VS,PR)
Input: BC, base classifiers constructed in the training phase;
       VS, the validation set;
Output: PR, pruned result.
{
1.  Initializing the pruned result: PR.set=∅; PR.correct=0;
2.  Getting classifying results: T=get_classifying_result(BC,VS);
3.  for each k in [1,L]
4.      T_k=refine_table(T);          //Refine the prediction results, delete the useless rows in T
5.      Tree=BuildFP-Tree(T_k);       //Construct a FP-Tree based on T_k
6.      S=select_classifiers(Tree,k); //Obtain the pruned ensemble with size of k
7.      If (S.correct>PR.correct) then
8.          PR.correct=S.correct;     //Record the best combined accuracy
9.          PR.set=S.set;             //Record the classifiers set with the best combined accuracy
10.     end if
11. end for
}
```

在 CPM-EP 算法中,首先对选择结果进行初始化(算法 1 中的第 1 行);然后,将对 VS 中各样本分类正确的基分类器标识保存在预测结果表 T 中(算法 1 中的第 2 行);对所有可能的分类器大小 $k\in[1,L]$,获得精简后的预测结果表 T'(算法 1 中的第 4 行);并基于 T' 创建一棵等高的 FP-Tree 树(算法 1 中的第 5 行),然后基于该 FP-Tree 获得 k 对应的选择性集成结果(算法 1 中的第 6 行);最后,从所有结果中选取对 VS 的预测精度最高的结果作为算法的输出(算法 1 中的第 7 行~第 11 行).

下面我们对分类结果获取、预测结果表精简、FP-Tree 的构建以及基分类器选择这 4 个步骤予以详细介绍.

1.2 获取分类结果

现在利用 L 个基分类器对校验样本集中的样本进行分类,分类结果保存在预测结果表 T 中,每个样本的分类结果形成表格 T 的一行.表 T 的每一行是一个三元组(*CID*,*Itemset*,*Num*),其中,*CID* 是样本标识,*Itemset* 保存着对该样本分类正确的所有基分类器标识,*Num* 记录了 *Itemset* 中分类器的数目.为了描述方便,我们用(X_i,$itemset_i$,L_i)描述表 T 中样本 X_i 对应的分类结果.

我们举一个简单的例子:假设 L=8,对应的基分类器标识分别为 $h_1,h_2,\ldots,h_8$,校验样本集 VS 中的样本数为 12,样本标识分别为 $X_1,X_2,\ldots,X_{12}$.这 8 个基分类器对 VS 中的所有样本的分类结果见表 1.以表 T 的第 2 行为例,该行表明有 5 个基分类器对样本 X_2 的预测结果正确,它们分别是 h_2,h_3,h_4,h_5,h_7.

Table 1 Predictive result table T

表 1 预测结果表 T

CID	*Itemset*	*Num*	*CID*	*Itemset*	*Num*
X_1	$h_1,h_2,h_3,h_4,h_5,h_6,h_7,h_8$	8	X_7	h_5,h_6,h_7	3
X_2	h_2,h_3,h_4,h_5,h_7	5	X_8	h_2,h_5,h_7	3
X_3	h_2,h_5,h_6	3	X_9	h_3,h_4,h_5,h_7	4
X_4		0	X_{10}	h_1,h_2,h_5,h_6	4
X_5	h_1,h_2,h_6,h_8	4	X_{11}	h_2,h_5,h_6	3
X_6	$h_1,h_2,h_3,h_4,h_5,h_6,h_7,h_8$	8	X_{12}	h_1,h_2,h_4,h_6,h_7	5

1.3 精简预测结果表

若集成分类器的大小为 k,根据多数投票原则,如果集成分类器中对某样本分类正确的基分类器数目大于等于$\lfloor k/2\rfloor+1$,那么该集成分类器肯定能够正确识别该样本.因此,对于给定的集成分类器大小 k,我们仅考虑分类器数目大于等于$\lfloor k/2\rfloor+1$ 的样本,将其余样本的分类结果删除.同时,对于样本 $X_i(1\leqslant i\leqslant n)$,如果满足

$$L_i=L \tag{1}$$

即,L 个基分类器对样本 X_i 均分类正确.因此,无论选择哪些基分类器进行集成,集成分类器对样本 X_i 的分类结果肯定正确.也就是说,满足条件(1)的样本对基分类器的选择不会起作用,可将它们的分类结果信息也从表 T 中删除.然后,根据每个分类器在所有入选的 *Itemset* 中的出现频率对各 *Itemset* 中的分类器进行排序.最后,对每个已排序的 *Itemset*,截取前面$\lfloor k/2\rfloor+1$ 个分类器,得到精简的预测结果表 T_k.

表 2 的第 2 列给出,当 k=5 时,根据表 1 入选的 *itemset*;第 3 列为排序后的结果;第 4 列为最后得到的精简预测结果表 T_5.由于当 k=5 时有

$$\lfloor k/2\rfloor+1=\lfloor 5/2\rfloor+1=2+1=3,$$

因此,T_5 中各行的分类器数目均为 3.

Table 2 Refining of predictive result table

表 2 精简预测结果表

CID	*Itemset*	Sorted *Itemset*	Refined *Itemset* (T_5)
X_2	h_2,h_3,h_4,h_5,h_7	h_2,h_5,h_7,h_4,h_3	h_2,h_5,h_7
X_3	h_2,h_5,h_6	h_2,h_5,h_6	h_2,h_5,h_6
X_5	h_1,h_2,h_6,h_8	h_2,h_6,h_1,h_8	h_2,h_6,h_1
X_7	h_5,h_6,h_7	h_5,h_6,h_7	h_5,h_6,h_7
X_8	h_2,h_5,h_7	h_2,h_5,h_7	h_2,h_5,h_7
X_9	h_3,h_4,h_5,h_7	h_5,h_7,h_4,h_3	h_5,h_7,h_4
X_{10}	h_1,h_2,h_5,h_6	h_2,h_5,h_6,h_1	h_2,h_5,h_6
X_{11}	h_2,h_5,h_6	h_2,h_5,h_6	h_2,h_5,h_6
X_{12}	h_1,h_2,h_4,h_6,h_7	h_2,h_6,h_7,h_1,h_4	h_2,h_6,h_7

1.4 构建FP-Tree

对于我们研究的选择性集成问题,经过前面两节的处理,预测结果表中的样本标识可以类比于事务标识号,*Itemset* 列类比于购物篮问题中的事务[14],表中的基分类器则类比于购买的商品.这样,T 和 T_k 均可看作是事务数据库.

FP-Tree 结构在数据挖掘领域中频繁项集的获取方面取得了重要应用[14].由于 FP-Tree 合并了事务数据库中具有相同频繁项集的事务,不仅节约了存储空间,而且提高了频繁项集获取的效率.下面我们基于 T_k 创建一棵 FP-Tree 树.由于预先不知道将选择哪些基分类器,因此在创建 FP-Tree 的过程中,我们不设置支持度的限制.FP-Tree 树中的每个结点由 4 个域组成:分类器标识 *item-name*、经过该路径的样本计数 *count*、结点链 *node-link* 以及父结点指针 *parent*.另外,为了方便对树进行操作,还创建了一个头表 *HTable*,表中的记录包括 3 个域:分类器标识 *item-name*、分类器出现的次数 *item-num* 和结点链头 *node-head*.其中,*node-head* 指向 FP-Tree 中首次创建的具有相同分类器标识的结点.*HTable* 表中的记录按照对应项在所入选 *Itemset* 中的出现次数降序排列.通过调用 FP-Tree 创建函数 *BuildFP-Tree*(T_k),我们即可得到预测结果表 T_k 对应的 FP-Tree.FP-Tree 构造算法的具体描述见文献[14].

继续前面的例子,基于表 2 中的 T_5 所创建的 FP-Tree 如图 1 所示,其中,*HTable* 中的统计信息根据表 2 的第 2 列获得.图中带箭头的实线描绘出 FP-Tree 的结构;从 *HTable* 表中 *node-head* 域出发的虚线链指出了相应分类器在 FP-Tree 中对应的结点,链接的顺序即为该分类器对应结点的创建顺序.

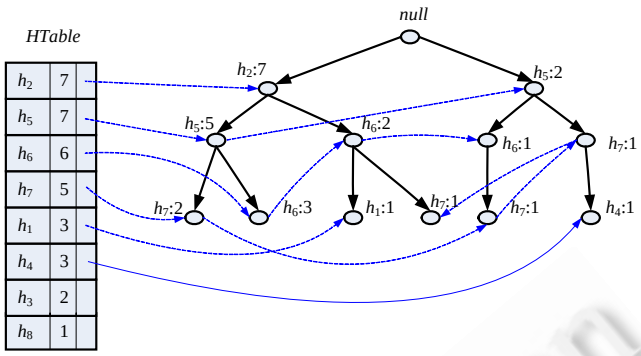

Fig.1 FP-Tree (k=5)

图 1 k=5 时的 FP-Tree

1.5 选择基分类器

在上一步,对于设定的 k,我们构造了对应的 FP-Tree,该树所有从根开始至叶结点的路径其深度都等于$\lfloor k/2 \rfloor$+1(不含根).下面,我们在此基础上进行基分类器的选择.由于我们希望选择 k 个基分类器进行集成,令得到的集成分类器为 E_k,根据多数投票法的规则可知:如果 T_k 中某事务 t 对应的所有分类器均在 E_k 中(我们称其为 t 被 E_k 覆盖),那么事务 t 对应的所有样本均可被 E_k 正确识别.针对这一思想,我们提出一种贪婪式方法进行分类器的选择.下面,我们对基分类器选择的核心部分,即算法 1 的第 6 行,*select_classifiers*(*Tree*,*k*)函数的功能进行详细阐述.

令选择的结果保存在参数 S 中,该参数包括两个域:*S.set* 记录了入选的基分类器集合,*S.correct* 记录了由 *S.set* 中的元素组成的集成分类器对表 T_k 中的样本分类正确的数目.为了描述方便,我们将任意一个集合 D 中的元素个数记为|D|.选择 k 个基分类器的算法描述如下:

步骤 1. 初始化入选集合:*S.set*=∅,*S.correct*=0.

步骤 2. 创建 Path-Table 表:按照从左至右的顺序,将 FP-Tree 中从根结点到叶结点的将各路径上出现的分类器及该路径叶结点的 *count* 值记录在路径表 Path-Table 中.Path-Table 表的每一行对应一条路径,并用 *classifiers*[i]表示 Path-Table 表中第 i 行对应的分类器集合,*count*[i]表示第 i 行对应的 *count* 值.

步骤 3. 分类器选择:从 Path-Table 表中选择 *count* 值最大的路径 i,如果有多行的 *count* 值均最大,则选取在 Path-Table 表中靠前的路径,如果有

$$|classifiers[i]|+|S.set|>k,$$

则说明,仅选择 k 个基分类器进行集成,无法对 Path-Table 表中第 i 行所包含的 *count*[i]个样本进行正确分类,因此将该行从 Path-Table 表中删除.否则,如果路径 i 满足

$$|classifiers[i]|+|S.set|\leqslant k,$$

则执行公式(2)和公式(3)的操作,将 *classifiers*[i]中的所有分类器加入集合 *S.set* 中,并更新可被正确分类的样本数目.

$$S.set=S.set\cup classifiers[i] \tag{2}$$

$$S.correct=S.correct\cup count[i] \tag{3}$$

最后删除该行,并将新选入 *S.set* 中的基分类器从 Path-Table 表的各行中删除.

步骤 4. 更新 Path-Table 表:如果 Path-Table 中某一行 i 的分类器集合 *classifiers*[i]为空,说明该行对应的 *count*[i]个样本均已被分类正确,执行公式(3)操作,然后将该行从 Path-Table 表中删除;如果存在分类器集合相同的行,则将它们合并为一行,该行的 *count* 值为参与合并的各行 *count* 值之和.

步骤 5. 重复步骤 3 和步骤 4,直到所选的分类器数目等于 k 或 Path-Table 表为空.

步骤 6. 返回 S.

上述算法是固定 k 值的分类器选择方法.为了获得泛化能力好的选择性集成结果,CPM-EP 算法对所有可能的 k 值分别进行基分类器选择,从它们的返回结果中,选取 *S.correct* 值最大的 k 值所对应的 *S.set* 中的基分类器参与集成.如果有多个 k 值对应的 *S.correct* 均最大,则选取最小 k 值对应的 *S.set* 中的分类器进行集成.显然,k 的所有可能的取值为区间[1,L]中的整数,其中,L 为原始基分类器数目.

对于前面讨论的例子,由图 1 中 FP-Tree 建立的路径长度为 3 的 Path-Table 表如图 2(a)所示.Path-Table 表中第 2 行的 *count* 值最大,其对应的分类器集合为{h_2,h_5,h_6}.将这 3 个分类器加入到集合 *S.set* 中,此时 *S.corrrect*=3,删除该行,并从 Path-Table 表中删除这 3 个分类器;然后,对分类器集合相同的行进行合并,得到更新后的 Path-Table 如图 2(b)所示.继续选择 *count* 值最大的行,h_7 入选,更新后的 Path-Table 表如图 2(c)所示.在图 2(c)中,由于 h_1 和 h_4 对应的 *count* 值相等,选择在 Path-Table 表中先出现的行,即 h_1 入选,最终得到的结果为 *S.set*={h_2,h_5,h_6,h_7,h_1}, *S.correct*=8.

对于这一例题,k 的所有可能的取值为区间[1,8]中的整数.其中,k=3 时具有最佳结果,得到 *S.set*={h_2,h_5,h_6},此时 *S.correct*=8.注意到 k=3 和 k=5 对应的 *S.correct* 值相同,我们选择较小的分类器集合进行集成.因此,对于该例题,CPM-EP 算法选择的基分类器为 h_2,h_5 和 h_6.

Classifiers	Count
h_2, h_5, h_7	2
h_2, h_5, h_6	3
h_2, h_6, h_1	1
h_2, h_6, h_7	1
h_5, h_6, h_7	1
h_5, h_7, h_4	1

(a) Original Path-Table (*length*=3) (a) 原始Path-Table(路径长度设为3)

Classifiers	Count
h_7	4
h_1	1
h_7, h_4	1

(b) Path-Table after the first selection (b) 第1次选择后的Path-Table

Classifiers	Count
h_1	1
h_4	1

(c) Path-Table after the second selection (c) 第2次选择后的Path-Table

Fig.2 Selecting base classifiers by using Path-Table

图 2 利用路径表 Path-Table 进行基分类器选择

2 实验设定

为了验证 CPM-EP 算法的有效性,在实验中,我们从泛化能力、分类器选择时间、参与集成的基分类器数目这 3 个方面对 Select-Best[15],Bagging[4],GASEN[12],Forward-Selection[16−18]和 CPM-EP 算法进行了比较.

2.1 数据集

实验中,我们使用了 UCI 机器学习数据库中 40 个来自不同领域的数据集.在实验过程中,某些数据集中存在对分类不起作用(如样本的标识属性)的属性,有些样本缺少某些属性值,于是我们将相应的属性和样本删除.经过上述处理,各数据集的特征见本节后的表 3.

2.2 参与比较的算法

我们在实验中选择了 Select Best(SelB),Bagging,GASEN 和 Forward Selection(FS)这 4 种算法与 CPM-EP 算法进行比较.其中,Select Best 是各集成学习算法比较的基础,Bagging 是最传统、最常用的集成算法,GASEN 和 Forward Selection 是两种典型的选择性集成算法.

- Select Best

Select Best 算法[15]不属于集成学习,主要用于作对比测试,它首先获得所有基分类器对校验样本集的预测精度,然后选择预测精度最好的一个基分类器.

- Bagging

Bagging 方法[4]直接对所有基分类器的预测结果进行多数投票法组合.

- GASEN

GASEN 算法由南京大学周志华等人提出,其主要思想是基于遗传算法.GASEN 将基分类器视为一个个的基因,一组基因组成一个个体,而多个个体又构成种群,随后利用交叉、变异等算子对个体进行改变,经过多代的进化,最后选择各代中具有最高适应度的个体作为算法的输出.遗传算法的核心是适应度函数的设计,GASEN 算法使用了一个校验数据集合作为适应度计算的依据,适应度函数是集成分类器在校验集上预测错误率的倒数.为了比较选择的效果,我们为 GASEN 增加了新的搜索目标,即,在搜索适应度更高个体的同时寻求更小的集成分类器.

根据文献[12]的描述,我们利用实数编码实现了标准的遗传算法,每一种群包括 30 个个体,每个个体是一个具有 50 个元素的数组,其中的元素是实数值,对应着相应基分类器的投票权重.算法中的相关参数设置如下:交叉概率为 0.5,变异概率为 0.05,分类器选择的权重阈值λ为 0.05.GASEN 算法的终止条件是完成 100 代的进化.

- Forward Selection(FS)

Forward Selection 算法[16−18]是一种贪婪式选择性集成算法,其算法的基本过程如算法 2 所示.

算法 2. Forward Selection 算法.

$FS(C,E_s)$

Input: C, set of all base classifiers;

Output: E_s, pruned result.

{

1. Initialization: $A_{\max}\leftarrow 0$, $E_s\leftarrow\phi$, $E\leftarrow\phi$;
2. while (C is not empty)
3. $c_1 \leftarrow \arg\max_{c\in C}\{f(\{c\}\cup E)\}$;
4. Remove all classifiers of c_1 from C;
5. $E\leftarrow\{c_1\}\cup E$;
6. if ($A_{\max}<f(E)$)
7. $A_{\max}\leftarrow f(E)$;
8. $E_s\leftarrow E$;
9. end if
10. end while

其中,浮点函数 $f(x)>=0$ 即是对集成分类器的判断标准(x 是一个集成分类器).FS 有多种变体,主要的变化都集中在贪婪算法所使用的启发式方法,也就是 $f(x)$的选择上.如,文献[16,17]提出的启发式方法是基于基分类器的多样性和性能,文献[18]提出的启发式方法是基于投票法的集成分类器预测精度.我们在实验中采用文献[18]的方法实现 FS 算法.

2.3 实验方法

实验采用 10 次交叉验证的方法,即样本被随机等分成 10 份,每次交叉取其中的一份作为测试集,其他 9 份作为训练集(称为交叉训练集).每次交叉共训练 50 个基分类器,各基分类器的训练集是从交叉训练集中采用 bootstrap 重采样获得.为了提高基分类器的多样性,我们设定基分类器的训练集大小为交叉训练集的一半,在基分类器的选择过程中使用的校验样本集即为交叉训练集.

我们采用异构的基分类器进行集成学习实验,在所生成的基分类器中有 15 个是 BPNN 神经网络[19],15 个是 C4.5 决策树[20],10 个是简单贝叶斯[2],10 个是 SVM(支持向量机),且各种基分类器的学习条件如下:

- 所有的 BP 神经网络均有一个隐含层,且隐含层结点为 5 个.神经网络的训练终止条件为均方差连续 5

个训练周期不发生改变(定义为相邻两次训练的 *MSE* 相差小于 10^{-5}),或达到最大训练周期 2 000.上述终止条件能够在一定程度上减轻神经网络对训练集的过适应;

- C4.5 决策树使用原始 C4.5 过程进行创建和剪枝,剪枝置信度为 0.25;
- 支持向量机采用 C-SVM,核函数采用径向基函数(RBF,其中,gamma 取值为数据集属性数目的倒数),训练终止条件 epsilon 设为 0.001.

实验中,程序代码均采用 C/C++语言实现,测试平台为 AMD 4000+,2GB 内存,Linux 操作系统.基分类器的数目为 50.

Table 3 Data sets used for classification

表 3 数据集特征

Data set	Size	Attribute	Class	Data set	Size	Attribute	Class
Abalone	4 177	8	29	Iris	150	4	3
Agaricus-Lepiota (mushroom)	5 644	22	2	Kr-vs-Kp	3 196	36	2
Austra	690	14	2	Letter-Recognition	20 000	16	26
Balance-Scale	625	4	3	Optdigits	3 823	64	10
Breast-Cancer-Wisconsin	683	9	2	Page	5 473	10	5
Bupa	345	6	2	Pima	768	8	2
Cancer	699	9	2	Poker-Hand	25 010	10	10
Car	1 728	6	4	Segmentation	2 100	19	7
Cleveland	303	13	5	Sick	1 947	21	2
Cmc	1 473	9	3	Sick-Euthyroid	2 000	18	2
Crx	465	15	2	Spambase	4 601	57	2
Dermatology	358	34	6	Splice	3 190	60	3
German-Numeric	1 000	24	2	Tic-Tac-Toe	958	9	2
Glass	214	9	6	Transfusion	748	4	2
Hayes-Roth	132	4	3	Vehicle (statlog)	846	18	4
Heart	270	13	2	Waveform	5 000	21	3
House-Votes-84	232	16	2	Wdbc	569	30	2
Hypothyroid	2 000	18	2	Wine	178	13	3
Imports-85 (autos)	159	25	7	Yeast	1 484	8	10
Ionosphere	351	34	2	Zoo	101	16	7

3 实验结果分析

我们从 3 个方面对 SelB,Bagging,GASEN,FS 和 CPM-EP 进行比较,即预测精度(泛化能力)、分类器选择时间、参与集成的基分类器数目.后文的表 4 给出了预测精度的结果,后文的表 6 给出了分类器选择时间和基分类器数目的结果.两个表中最好的结果均用黑体标出,最后一行 Average 为所有数据集相应结果的均值.

3.1 预测精度

表 4 中的预测精度是对测试集的预测精度,结果为 10 次交叉验证的均值和方差.根据文献[21],在多个数据集上比较多种分类方法,较好的选择是比较各方法在各数据集上分类精度的名次,而不是直接比较各方法的分类精度,这样可以排除单个数据集对均值结果影响过大的现象.因此,表 4 的右边给出了 5 种方法按精度排序的名次.从表 4 中显示的结果可以看出:在预测精度方面,40 个数据集中有 17 个是 CPM-EP 算法具有最佳值,Bagging 算法在 14 个数据集上占优,FS 在 11 个数据集上获得最优结果,而 Select Best 和 GASEN 仅各在 2 个数据集上获得好的结果.总体来看,相比其他方法,CPM-EP 算法在预测精度方面具有较明显的优势,FS 好于 Bagging, Bagging 又略优于 GASEN 算法,Select Best 的结果最差.

下面,我们将基于 Friedman 测试[21]对所有算法的平均名次进行显著性测试.首先设定空假设 H_0:所有算法都是等价的,然后计算 χ_F^2 值:

$$\chi_F^2 = \frac{12N}{k(k+1)}\left[\sum_j R_j^2 - \frac{k(k+1)^2}{4}\right],$$

其中,N 是测试集的数目,k 是对比测试的方法数目,R_j 是方法 j 在所有测试集上的平均名次.本实验中计算得到 χ_F^2 =37.915,下一步计算 F_F 值:

$$F_F=\frac{(N-1)\chi_F^2}{N(k+1)-\chi_F^2}.$$

得到 F_F=12.1119.由于 F_F 符合自由度为(k–1,(k–1)(N–1))的 F 分布,在显著性水平 P=0.05 的情况下,查表得到 F(4,156)=2.46,小于 F_F 值.因此,空假设 H_0 被拒绝,即对比的 5 种方法存在显著差别.因此,我们可以进入后续分析.

Table 4 Accuracy and rank of all methods

表 4 预测精度及其排名

Data set	Predictive accuracy (%)					Rank				
	SelB	Bagging	GASEN	FS	CPM-EP	SelB	Bagging	GASEN	FS	CPM-EP
Abalone	19.09±1.01	**26.08±1.11**	24.38±1.27	23.25±1.01	22.94±1.04	5.00	**1.00**	2.00	3.00	4.00
Agaricus-Lepiota (mushroom)	**100.00±0.00**	99.81±0.09	99.73±0.13	**100.00±0.00**	**100.00±0.00**	**2.00**	4.00	5.00	**2.00**	**2.00**
Austra	77.10±1.94	**86.09±1.42**	85.22±1.61	80.72±1.72	82.75±1.57	5.00	**1.00**	2.00	4.00	3.00
Balance-Scale	89.05±3.16	87.94±1.38	86.51±2.57	**92.06±1.81**	91.27±1.67	3.00	4.00	5.00	**1.00**	2.00
Breast-Cancer-Wisconsin	95.65±1.80	**97.68±1.03**	96.38±1.38	97.54±1.08	97.25±1.05	5.00	**1.00**	4.00	2.00	3.00
Bupa	65.14±3.12	64.57±3.68	64.57±3.39	64.57±4.30	**67.71±2.86**	2.00	4.00	4.00	4.00	**1.00**
Cancer	95.29±1.47	**97.29±1.21**	96.14±1.36	95.71±1.11	96.14±1.36	5.00	**1.00**	2.50	4.00	2.50
Car	93.29±0.99	91.56±1.17	91.97±1.56	95.49±1.02	**95.66±0.78**	3.00	5.00	4.00	2.00	**1.00**
Cleveland	50.32±5.11	58.06±3.46	54.84±3.46	**58.39±4.81**	56.13±4.46	5.00	2.00	4.00	**1.00**	3.00
Cmc	45.00±2.36	**49.19±2.30**	48.24±1.96	49.05±2.10	47.30±2.27	5.00	**1.00**	3.00	2.00	4.00
Crx	85.74±2.65	**89.15±2.35**	88.94±2.32	84.68±2.85	86.81±2.81	4.00	**1.00**	2.00	5.00	3.00
Dermatology	96.94±1.31	96.39±1.53	95.28±1.40	95.83±2.43	**97.78±1.21**	2.00	3.00	5.00	4.00	**1.00**
German-Numeric	66.10±1.47	71.80±1.76	72.10±1.13	**73.90±1.65**	72.50±1.99	5.00	4.00	3.00	**1.00**	2.00
Glass	60.91±6.03	55.91±6.67	55.91±4.55	**62.27±5.84**	**62.27±5.48**	3.00	4.50	4.50	**1.50**	**1.50**
Hayes-Roth	79.29±4.64	75.71±7.35	76.43±7.83	**82.14±5.36**	80.71±4.53	3.00	5.00	4.00	**1.00**	2.00
Heart	77.78±3.70	**83.33±2.90**	82.22±2.72	80.74±4.60	81.11±2.92	5.00	**1.00**	2.00	4.00	3.00
House-Votes-84	96.67±1.82	**98.75±1.33**	97.08±1.33	96.67±1.82	97.08±1.63	4.50	**1.00**	2.50	4.50	2.50
Hypothyroid	98.35±0.45	98.70±0.37	98.35±0.49	98.45±0.51	**98.80±0.39**	4.50	2.00	4.50	3.00	**1.00**
Imports-85 (autos)	52.50±4.24	53.75±5.80	**58.13±5.94**	53.75±4.68	**58.13±6.10**	5.00	3.50	**1.50**	3.50	**1.50**
Ionosphere	88.33±1.94	**92.22±2.55**	91.11±3.39	88.89±3.67	91.11±2.55	5.00	**1.00**	2.50	4.00	2.50
Iris	**96.67±1.67**	96.67±1.67	90.67±6.00	**96.67±1.67**	**96.67±1.67**	**2.50**	2.50	5.00	**2.50**	**2.50**
Kr-vs-Kp	98.28±0.48	97.16±0.49	97.66±1.12	98.22±0.37	**98.78±0.40**	2.00	5.00	4.00	3.00	**1.00**
Letter-Recognition	79.40±0.37	81.22±0.32	84.64±1.97	88.10±0.40	**88.41±0.24**	5.00	4.00	3.00	2.00	**1.00**
Optdigits	96.66±0.52	95.95±0.49	95.33±0.78	95.77±0.64	**96.71±0.54**	2.00	3.00	5.00	4.00	**1.00**
Page	95.99±0.34	95.91±0.42	95.86±0.78	**96.93±0.22**	96.73±0.20	3.00	4.00	5.00	**1.00**	2.00
Pima	69.87±2.59	**76.62±1.79**	73.51±1.93	73.90±2.10	74.03±2.15	5.00	**1.00**	4.00	3.00	2.00
Poker-Hand	49.56±0.79	49.83±0.43	51.75±0.95	55.19±0.61	**55.97±0.71**	5.00	4.00	3.00	2.00	**1.00**
Segmentation	89.90±1.13	91.33±0.89	91.86±1.27	**93.76±0.82**	93.24±1.01	5.00	4.00	3.00	**1.00**	2.00
Sick	96.56±0.56	96.46±0.48	96.26±0.63	**97.03±0.60**	96.97±0.51	3.00	4.00	5.00	**1.00**	2.00
Sick-Euthyroid	97.45±0.48	97.35±0.51	96.65±0.86	97.40±0.67	**97.50±0.62**	2.00	4.00	5.00	3.00	**1.00**
Spambase	92.80±0.73	92.86±0.47	92.93±1.65	94.01±0.59	**94.21±0.66**	5.00	4.00	3.00	2.00	**1.00**
Splice	93.82±0.54	**96.39±0.26**	95.77±0.34	95.36±0.57	95.80±0.42	5.00	**1.00**	3.00	4.00	2.00
Tic-Tac-Toe	91.88±2.51	77.08±2.49	79.69±2.26	92.29±1.65	**95.10±1.34**	3.00	5.00	4.00	2.00	**1.00**
Transfusion	73.20±2.45	**77.73±1.61**	76.53±1.69	75.60±1.84	76.00±2.35	5.00	**1.00**	2.00	4.00	3.00
Vehicle (statlog)	66.00±3.69	70.12±1.37	73.06±1.69	**73.29±2.54**	73.18±3.45	5.00	4.00	3.00	**1.00**	2.00
Waveform	70.56±1.19	85.80±0.88	**86.08±0.95**	80.08±1.25	80.26±0.97	5.00	2.00	**1.00**	4.00	3.00
Wdbc	94.91±1.07	**96.32±1.21**	95.09±1.02	93.86±1.53	95.26±1.30	4.00	**1.00**	3.00	5.00	2.00
Wine	96.11±1.78	**98.89±1.11**	95.56±2.08	96.11±1.78	96.67±1.84	3.50	**1.00**	5.00	3.50	2.00
Yeast	50.00±1.04	56.71±1.42	54.50±1.77	56.85±2.14	**57.32±1.76**	5.00	3.00	4.00	2.00	**1.00**
Zoo	85.45±8.67	81.82±7.33	80.91±7.17	84.55±8.39	**86.36±7.94**	2.00	4.00	5.00	3.00	**1.00**
Average	80.44	82.16	81.69	82.73	**83.22**	3.95	2.79	3.55	2.74	**1.98**

在多个数据集上进行多种分类器的对比,一般采用全配对对比实验(all pairwise comparison),即将所有分类器进行两两对比,以发现所有可能的相似性.全配对对比方法主要有 Nemenyi 测试、Holm 测试、Shaffer 测试、Bergmann-Hommel 测试等,这些方法都采用如下统计量进行检验:

$$z=\frac{R_i-R_j}{\sqrt{\frac{k(k+1)}{6N}}}.$$

由于 z 符合正态分布,因此可以利用查表得到对应的概率值 p,该值反映了对应空假设的置信度水平.即,p 值越小,则空假设的可信度越低,被拒绝的概率越大.计算所有两两配对对应的 p 值,结合一定的显著性水平设置,我们就可以得出是否拒绝对应的空假设.使用 p 值进行两两对比检验是足够的,但在全配对对比实验中,在得出一个结论时还需要考虑其他对比实验的可能结果,否则,多个结论可能出现不一致的情况.因此,人们提出了 APV(adjusted P-value)思想,首先对 p 值进行修正得到 APV 值,然后使用 APV 值,再结合显著性水平设定得出结论.

在前述配对对比测试中,Bergmann-Hommel 是一种基于穷举集合(exhaustive set)的检测方法,它的目标是尽可能地发现所有不能被拒绝的空假设.该方法是目前检测能力最为强大的,能够在较少测试数据集的情况下得出更精确的结论.在本次实验中,我们将采用 Bergmann-Hommel 动态过程进行配对测试.该过程较为复杂,这里我们仅将计算结果显示于表 5,详情可参考文献[22].其中,H_0 为空命题,z 是统计量,p 是 z 对应的正态分布概率密度,APV 是根据调整后的概率密度值(APV 的计算请参考文献[22]),显著性水平 α<0.1,表格按照 p 值从小到大排序.

Table 5 Result of Bergmann-Hommel test

表 5 Bergmann-Hommel 检验结果

No.	H_0	z	p	APV	Is H_0 rejected?
0	SelB vs. CPM-EP	5.586 1	2.129e–008	2.129e–007	Yes
1	GASEN vs. CPM-EP	4.454 8	8.397e–006	5.038e–005	Yes
2	SelB vs. FS	3.429 5	0.000 604 8	0.003 629	Yes
3	SelB vs. Bagging	3.288 0	0.001 009	0.004 035	Yes
4	Bagging vs. CPM-EP	2.298 1	0.021 56	0.086 23	Yes
5	GASEN vs. FS	2.298 1	0.021 56	0.064 67	Yes
6	Bagging vs. GASEN	2.156 7	0.031 03	0.064 67	Yes
7	FS vs. CPM-EP	2.156 7	0.031 03	0.086 23	Yes
8	SelB vs. GASEN	1.131 4	0.257 9	0.515 8	No
9	Bagging vs. FS	0.141 4	0.887 5	0.887 5	No

根据表 5,我们得到以下结论:在显著性水平 α<0.1 的情况下,Select Best 与 GASEN 没有显著性差别,Bagging 与 FS 没有显著性差别,而 CPM-EP 与其他方法存在显著性差别.图 3 进一步用图形表示了 Bergmann-Hommel 检验的结论.我们可以看到,对比实验中的各种方法形成了 3 个分组,即,SelB 和 GASEN 一组,FS 和 Bagging 一组,而 CPM-EP 单独形成一组.这说明,CPM-EP 显著优于其他 4 种对比方法.

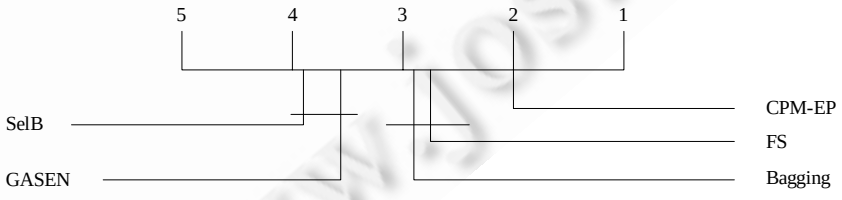

Fig.3 Graphical representation of Bergmann-Hommel test

图 3 Bergmann-Hommel 检验的图形化表示

需要强调的是,本节的测试结果进一步验证了选择性集成的作用,即将所有基分类器进行直接集成并不一定具有好的效果,选择一部分基分类器参与集成往往能够获得更好的泛化能力.同时我们认为,CPM-EP 算法之所以能够取得好的泛化能力,主要在于其选择基分类器的过程类似于频繁模式获取问题,从而能够减轻选择过

程中对校验样本集的过适应现象.

3.2 选择时间

3 种选择性集成算法的基分类器选择时间见表 6,结果为 10 次交叉验证的均值.实验结果显示,CPM-EP 算法的选择时间平均约为 GASEN 的 1/19、FS 的 1/8,明显优于这两种算法.主要原因是,CPM-EP 算法利用事务数据库保存基分类器对校验样本的预测结果,避免了对预测结果的重复获取.此外,利用 FP-Tree 进行分类器选择,进一步提高了基分类器的选择速度.GASEN 采用的遗传算法是一种非常耗时的搜索方法,虽然我们对 GASEN 进行了一定的优化,例如采用查表法来避免每次进化时对校验样本集的重复分类,但表 6 的结果仍然表明,GASEN 是一种运算开销很大的选择性集成方法.FS 算法在每个循环中都尝试着将剩余基分类器一一加入集成分类器,其选择时间介于 GASEN 和 CPM-EP 算法之间.

Table 6 Pruning time and size of pruned ensembles

表 6 分类器选择时间和基分类器数目

Data set	Pruning time (s)			Size of pruned ensemble		
	GASEN	FS	CPM-EP	GASEN	FS	CPM-EP
Abalone	791.85	448.45	**39.31**	**7.50**	20.30	20.80
Agaricus-Lepiota (mushroom)	115.12	51.87	**0.16**	2.70	**1.00**	**1.00**
Austra	22.25	8.84	**0.13**	**6.40**	6.60	9.80
Balance-Scale	20.53	7.39	**0.14**	12.90	**4.30**	6.70
Breast-Cancer-Wisconsin	18.88	7.40	**0.03**	4.10	**2.20**	3.80
Bupa	10.86	3.64	**0.27**	21.50	**8.80**	13.00
Cancer	18.10	7.15	**0.03**	4.10	**2.20**	5.40
Car	46.57	19.22	**0.59**	14.00	**5.70**	6.80
Cleveland	13.35	4.49	**0.33**	17.70	15.70	**8.20**
Cmc	13.28	2.70	**1.29**	12.50	12.80	**11.00**
Crx	17.25	6.28	**0.08**	8.70	**6.20**	7.80
Dermatology	19.77	7.04	**0.04**	5.80	**1.10**	1.20
German-Numeric	28.86	10.14	**1.04**	**14.10**	18.50	17.40
Glass	11.07	3.57	**0.13**	14.10	**4.90**	11.30
Hayes-Roth	0.82	0.27	**0.01**	7.50	**2.40**	3.30
Heart	9.69	3.35	**0.09**	13.70	**4.10**	10.60
House-Votes-84	7.84	2.76	**0.01**	2.90	**1.00**	**1.00**
Hypothyroid	44.80	18.24	**0.09**	6.40	**2.70**	5.60
Imports-85 (autos)	10.38	3.29	**0.09**	14.70	**5.70**	7.80
Ionosphere	11.79	3.68	**0.05**	8.90	**3.90**	5.40
Iris	5.31	1.85	**0.00**	2.90	**1.00**	**1.00**
Kr-vs-Kp	71.56	29.75	**0.26**	9.40	**4.20**	14.90
Letter-Recognition	3 991.89	1 502.11	**143.52**	18.70	19.20	**13.80**
Optdigits	205.69	82.64	**2.19**	26.00	16.40	**3.20**
Page	128.32	60.53	**1.02**	11.00	13.90	**7.20**
Pima	20.09	6.96	**0.67**	11.90	**10.70**	12.60
Poker-Hand	3 900.65	1 444.13	**316.97**	**7.20**	18.50	15.00
Segmentation	74.82	30.31	**0.60**	14.90	**8.90**	16.10
Sick	45.19	17.85	**0.14**	7.30	**3.20**	6.60
Sick-Euthyroid	41.55	17.42	**0.18**	12.10	4.00	**3.20**
Spambase	91.73	36.82	**1.76**	22.10	**14.00**	20.80
Splice	72.45	32.60	**0.60**	11.70	**11.20**	13.00
Tic-Tac-Toe	26.35	9.91	**0.31**	9.10	**3.10**	5.20
Transfusion	17.26	5.95	**0.44**	13.40	8.10	**5.60**
Vehicle (statlog)	29.48	10.36	**0.89**	13.70	**10.80**	11.60
Waveform	103.82	40.51	**7.78**	**14.70**	17.80	17.40
Wdbc	18.47	6.42	**0.05**	5.50	**1.20**	2.20
Wine	7.11	2.51	**0.01**	3.50	**1.00**	**1.00**
Yeast	55.70	22.25	**6.19**	**9.10**	15.10	17.00
Zoo	5.44	1.70	**0.02**	5.50	**1.20**	1.60
Average	253.65	99.56	**13.19**	10.75	**7.84**	8.67

3.3 基分类器的数目

表 6 给出的基分类器数目是各算法得到的集成分类器中参与集成的基分类器数,结果为 10 次交叉验证的均值.由于 Bagging 算法是直接集成法,不用对基分类器进行选择,基分类器数目为原始基分类器的数目 50,在表

中未列出 Bagging 的值.Select Best 的基分类器数目始终为 1,表 6 也未列出其相应数值.

从结果可以看出,选择性集成算法选择出来的基分类器数目远远小于原始的基分类器数目.其中,FS 算法选取的基分类器最少,CPM-EP 次之.均值的结果显示,3 种选择性集成算法选取的基分类器数目均不到原始基分类器数目的一半.结合预测精度的结果,我们可以得到如下结论:实验中的选择性集成算法均能剔除对提高泛化能力不起作用的基分类器.

由于选择性集成算法选取的基分类器数目远小于原始的基分类器数,因此,相比于 Bagging 等直接集成法,选择性集成算法生成的集成分类器在预测新样本的类别时不仅计算量大为减少,而且所需的内存空间也大为减小.在我们的实验中,在对测试样本进行类别预测时,CPM-EP 算法生成的集成分类器在预测阶段所需的计算开销明显小于 Bagging,在很大程度上小于 GASEN,略差于 FS.

4 结 论

本文提出一种新的选择性集成算法 CPM-EP.该算法将基分类器的选择问题转换为对事务数据库的处理问题,从而能够用 FP-Tree 结构保存各基分类器对校验样本的预测结果,避免了选择过程中重复进行基分类器预测结果的获取.CPM-EP 算法不仅计算开销小,而且利用模式挖掘的特点减轻了分类器选择过程对校验样本集的过适应问题.实验结果表明,CPM-EP 算法以很小的运算开销获得了优越的泛化能力,且获得的集成分类器规模小,是一种高效的选择性集成算法.

References:

[1] Mitchell TM. Machine Learning. New York: McGraw-Hill Science/Engineering/Math, 1997.

[2] Jiang YH, Zhao QL. Machine Learning Techniques. Beijing: China Publishing House of Electronics Industry, 2009 (in China).

[3] Dietterich T, Thomas G. Machine learning research: Four current directions. AI Magazine, 1997,18(4):97−136.

[4] Breiman L. Bagging predictors. Machine Learning, 1996,24(2):123−140. [doi: 10.1023/A:1018054314350]

[5] Schapire RE. A brief introduction to boosting. In: Dean T, ed. Proc. of the 16th Int'l Joint Conf. on Artificial Intelligence (IJCAI'99). San Francisco: Morgan Kaufmann Publishers, 1999. 1401−1406.

[6] Parmanto B, Munro PW, Doyle HR. Improving committee diagnosis with resampling techniques. In: Touretzky DS, Mozer MC, Hesselmo ME, eds. Advances in Neural Information Processing Systems. Cambridge: MIT Press, 1996. 882−888.

[7] Allwein EL, Schapire RE, Singer Y. Reducing multiclass to binary: A unifying approach for margin classifiers. Journal of Machine Learning Research, 200,1(1):113−141. [doi: 10.1162/15324430152733133]

[8] Jiang YH, Zhao QL, Yang XJ. A search coding method and its application in supervised classification. Journal of Software, 2005, 16(6):1081−1089 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/16/1081.htm [doi: 10.1360/jos161081]

[9] Ruta D, Gabrys B. Classifier selection for majority voting. Information Fusion, 2005,6(1):63−81. [doi: 10.1016/j.inffus.2004.04.008]

[10] Ali KM, Pazzani MJ. Error reduction through learning multiple descriptions. Machine Learning, 1996,24(3):173−202. [doi: 10.1023/A:1018249309965]

[11] Wolpert DH. Stacked generalization. Neural Networks, 1992,5(2):241−259. [doi: 10.1016/S0893-6080(05)80023-1]

[12] Zhou ZH, Wu J, Tang W. Ensembling neural networks: Many could be better than all. Artificial Intelligence, 2002,137(1-2): 239−263. [doi: 10.1016/S0004-3702(02)00193-5]

[13] Partalas I, Tsoumakas G, Vlahavas I. Pruning an ensemble of classifiers via reinforcement learning. Neurocomputing, 2009,72(7-9): 1900−1909. [doi: 10.1016/j.neucom.2008.06.007]

[14] Han JW, Kamber M. Data Mining: Concepts and Techniques. San Francisco: Morgan Kaufmann Publishers, 2000.

[15] Dzeroski S, Zenko B. Is combining classifiers better than selecting the best one. In: Sammut C, Hoffmann AG, eds. Proc. of the 19th Int'l Conf. on Machine Learning. San Francisco: Morgan Kaufmann Publishers, 2002. 123−130.

[16] Margineantu D, Dietterich T. Pruning adaptive boosting. In: Fisher DH, ed. Proc. of the 14th Int'l Conf. on Machine Learning. San Francisco: Morgan Kaufmann Publishers, 1997. 211−218.

[17] Caruana R, Niculescu-Mizil A, Crew G, Ksikes A. Ensemble selection from libraries of models. In: Brodley CE, ed. Proc. of the 21st Int'l Conf. on Machine Learning (ICML 2004). New York: ACM, 2004. 18. http://portal.acm.org/citation.cfm?id=1015330.1015432 [doi: 10.1145/1015330.1015432]

[18] Fan W, Chu F, Wang H, Yu PS. Pruning and dynamic scheduling of cost-sensitive ensembles. In: Dechter R, Kearns M, Sutton R, eds. Proc. of the 19th National Conf. on Artificial Intelligence. Menlo Park: American Association for Artificial Intelligence, 2002. 146−151.

[19] Rumelhart DE, Hinton GE, Williams RJ. Learning internal representations by error propagation. In: Rumelhart DE, McClelland JL, eds. Proc. of the Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Cambridge: MIT Press, 1986. 318−362.

[20] Quinlan JR. C4.5: Programs for Machine Learning. San Mateo: Morgan Kaufmann Publishers, 1993.

[21] Demsar J. Statistical comparisons of classifiers over multiple data sets. Journal of Machine Learning Research, 2006,7(1):1−30.

[22] García S, Herrera F. An extension on "statistical comparisons of classifiers over multiple data sets" for all pairwise comparisons. Journal of Machine Learning Research, 2008,9(12):2677−2694.

附中文参考文献:

[2] 蒋艳凰,赵强利.机器学习方法.北京:电子工业出版社,2009.

[8] 蒋艳凰,赵强利,杨学军.一种搜索编码法及其在监督分类中的应用.软件学报,2005,16(6):1081−1089. http://www.jos.org.cn/1000-9825/16/1081.htm [doi: 10.1360/jos161081]

赵强利(1973－),男,陕西宝鸡人,博士生,主要研究领域为机器学习,信息安全.

蒋艳凰(1976－)女,博士,副研究员,主要研究领域为机器学习,高性能计算.

徐明(1964－),男,博士,教授,博士生导师,CCF 高级会员,主要研究领域为移动计算,人工智能.