第32卷 第17期 计 算 机 工 程 2006年9月
Vol.32 № 17 Computer Engineering September 2006
·专题论文· 文章编号：1000—3428(2006)17—0100—03 文献标识码：A 中图分类号：TP181

集成学习中完全随机学习策略研究

俞 扬，周志华
(南京大学软件新技术国家重点实验室，南京 210093)

摘 要：以完全随机树(不包含属性选择过程的决策树)作为基学习器的集成，具有很好的性能。该文探讨了完全随机学习策略推广情况，实现了完全随机决策树桩算法和完全随机规则算法，分析有效的原因。实验表明，性能良好的完全随机算法，易于被许多初学者所掌握。

关键词：机器学习；集成学习；完全随机策略

Research on Complete Random Learning Scheme in Ensemble Learning

YU Yang, ZHOU Zhihua
(National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210093)

【**Abstract**】Ensemble of complete random trees, i.e. decision trees without any split selection, has high performance. This paper investigates whether the complete random learning scheme can by applied to other types of base learners. It realizes complete random decision stump and complete random rule algorithms, analyzes why complete scheme work. Experiments show that complete random scheme works for different types of base learners.

【**Key words**】Machine learning; Ensemble learning; Complete random scheme

1 概述

集成学习在同一问题上，训练基学习器的多个版本，再将其预测结果结合，得出集成学习器的预测结果。由于集成学习能够有效地提高预测性能，因此成为一个研究热点并受到了广泛的关注[1]。

集成学习中的个体学习器往往通过传统学习算法，进行随机化处理而形成。例如 Bagging 算法[2]通过可重复采样将基学习器的输入随机化；随机子空间(Random Subspace)算法[3]将基学习器的输入空间随机化；随机森林(RandomForests)算法[4]将决策树的属性进行挑选，并进行一定程度的随机化处理。

然而，随机化产生的个体学习器往往带有其在传统算法中使用的优化过程。例如，在基于 Bagging 或基于随机子空间的 C4.5[5]决策树训练过程中，每一个树结点都会挑选当前最优属性作为划分属性；在随机森林的训练过程中，每一个树结点会在一个随机属性子集中，挑选最优属性作为划分属性，因此仍然需要对属性进行评价。

最近的研究发现，以完全随机决策树(Complete Random Decision Tree)[6,7]作为基学习器的集成学习器，其性能优于基于 C4.5 的 Bagging 算法和随机森林。具体来说，完全随机决策树的每一个树结点使用的划分属性都是随机选取的，而不是像 C4.5 算法中，使用信息熵对属性进行挑选，因此在完全随机树的训练过程中，没有任何优化过程。这一研究结果揭示，在集成学习框架中，决策树的属性选择机制不是必要的，它可以被效率更高的完全随机机制替代。

在集成学习中，为了验证这种完全随机学习策略推广情况，本文将完全随机学习策略应用到其它学习算法中，构造了完全随机决策树桩和完全随机规则，这是两种具有不同数据拟合程度的算法。在现实世界数据集上与决策树桩算法和规则学习算法进行的实验比较结果表明，完全随机策略可以应用于不同类型的基学习器中，并作出了分析。

2 完全随机决策树

文献[8]揭示了集成学习器，具有良好的泛化性能的两点要素：(1)基学习器的精度要高；(2)基学习器之间差异要大。

在此之后，研究者们提出了许多扰动方法，以增加基学习器之间的差异。其中，随机森林算法[4]尝试了对决策树节点的属性选择进行扰动，并且取得了很好的性能。

完全随机树算法如下：

```
Input:
D : Data set
Process:
if |D|>=1 then stop
if instances in D are in one class then stop
   A := random selected attribute
if A is a nominal attribute then
   D^s := split D into subsets by A
   for each data set D^s_i in D^s
   construct a sub-tree on D^s_i
else
```

基金项目：国家杰出青年科学基金资助项目(60325207)；教育部优秀青年教师资助计划基金资助项目；霍英东基金资助项目(91067)
作者简介：俞 扬(1982—)，男，硕士生，主研方向：机器学习，演化计算；周志华，博导、教授
收稿日期：2006-04-29 **E-mail**：yuy@lamda.nju.edu.cn

v := random selected split value

$\{D_<, D_\geq\}$:= split D according to v

construct sub-trees on $D_<$ and $D_\geq$

进一步对决策树结点的属性选择加强扰动，得到了完全随机树[6]。在完全随机树的构造过程中，一个树结点首先随机选择一个属性作为划分属性，如果该属性是名词性属性，则对每一个属性值生长一棵子树；如果该属性是连续属性，则随机选择一个属性值，将数据划分为两部分，对每一部分生长一棵子树。

实验表明[7]，基于加权投票结合方式的完全随机树集成的性能优于 C4.5、Bagging 和随机森林。

3 完全随机策略的推广

为了验证完全随机策略的适用性，这里将完全随机策略推广到不同类型的算法上。注意到完全随机树可以拟合到每一个样本上，因此构造了两种具有不同拟合程度的算法，即完全随机决策树桩和完全随机规则。

决策树桩[9]只取一个属性进行划分的决策树，完全随机单层决策树随机的选择一个属性作为划分属性，并随机选择一个属性值作划分。

(1)如果该属性是离散属性，则将数据划分为等于划分值和不等于划分值两部分子空间；

(2)如果该属性是连续属性，则将数据划分为小于划分值和大于等于划分值的两个子空间。

然后统计每一部分数据上的类分布。当进行预测时，首先判断预测样本落在哪个划分子空间中，然后返回，在本子空间中，训练样本类分布。

完全随机单层决策树算法如下：

Input: D : Data set

Process:

A := random selected attribute

if A is a nominal attribute then

v := random selected nominal value

$\{D_{=v}, D_{\neq v}\}$:= split D according to v

count distribution on $D_{=v}$, and $D_{\neq v}$

else

v := random selected split value

$\{D_<, D_\geq\}$:= split D according to v

count distribution on $D_<$ and $D_\geq$

规则学习从输入样本中归纳出一组覆盖规则，这些规则以属性值条件作为前件，以类别作为后件[10]。完全随机规则，即在输入空间用属性值随机地划分出一块子空间，统计该子空间中的类分布。作预测时，如果预测样本落在划分该子空间中，则返回类分布，否则返回空。

从数据拟合程度的角度来看，完全随机树可以进行任意多次划分，拟合到每一个样本，拟合程度最高；完全随机决策树桩由于只作一次划分，拟合程度最低；完全随机规则可能产生只使用一个属性的规则，也可能产生使用全部属性的规则，因此拟合程度介于完全随机树与完全随机决策树桩之间。

完全随机规则算法如下：

Input:

D:Data set

Process:

N := random selected number of attributes

$\{A_1,\ldots,A_N\}$:= random selected attributes

$\{v_1,\ldots,v_N\}$:= random selected

values for each attribute

$\{c_1,\ldots,c_N\}$:= random selected expression

(ci = '≤' or '≥' when A_1 is continuous

ci = '=' when A_1 is nomial)

count distribution in the rule region

4 实验测试

4.1 实验设置

本文在 20 个 UCI[11]数据集上进行了实验比较。实验使用 5 次 3 倍交叉验证，估计学习器的性能。将数据集随机等分为 3 份，依次使用其中 1 份作为测试集，而其它 2 份作为训练集，得到平均测试误差。

将上面实验重复 5 次，得到这 5 次的平均测试误差，作为对学习起泛化误差的估计。所有集成算法使用的个体分类器数量为 500。

4.2 与传统算法的比较

为验证基于完全随机单层决策树和完全随机规则的集成学习器的性能，将完全随机决策树桩的集成(CR-DS)与标准的决策树桩(DS)进行了比较，将完全随机规则的集成(CR-rule)与一种规则学习算法 PRISM[10]进行了比较。这 4 种算法在 20 个数据集上的误差见表 1。

表 1 与传统优化算法的比较

Data set	DS	CR-DS	PRISM	CR-Rule
balance	0.405 4	0.155 5	0.439 3	0.1258
breast-c	0.295 3	0.292 4	0.314 1	0.289 6
breast-w	0.078 2	0.055 1	0.056 2	0.133 8
cmc	0.573 0	0.573 0	0.580 3	0.565 8
credit-a	0.136 3	0.231 5	0.193 6	0.179 8
credit-g	0.300 0	0.300 0	0.279 0	0.300 0
cylinder	0.379 1	0.357 4	0.397 2	0.356 7
haberman	0.269 3	0.264 7	0.268 6	0.264 7
heart-c	0.252 6	0.215 4	0.208 0	0.198 6
heart-s	0.279 3	0.234 1	0.224 4	0.229 6
iris	0.333 3	0.138 8	0.105 1	0.080 7
liver	0.419 1	0.420 3	0.579 1	0.417 4
lymph	0.267 9	0.330 4	0.269 9	0.218 4
primarytumor	0.710 0	0.784 6	0.635 0	0.590 1
promoters	0.234 4	0.131 6	0.263 0	0.408 1
spect	0.273 8	0.273 8	0.217 8	0.207 4
spectf	0.274 5	0.272 2	0.205 6	0.269 9
vehicle	0.599 5	0.486 0	0.394 3	0.404 9
yeast	0.593 4	0.620 5	0.780 3	0.644 6
zoo	0.393 8	0.424 4	0.381 8	0.194 4
average	0.353 4	0.328 1	0.339 6	0.304 0

由表 1 可见，CR-DS 在 11 个数据集(55%)上误差低于 DS，在 6 个数据集(30%)上误差高于 DS；CR-Rule 在 14 个数据集(70%)上误差低于 PRISM，在 6 个数据集(30%)上误差高于 PRISM。这说明了在集成学习的框架下，完全随机策略应用到决策树桩和规则学习上是有效的，即集成学习的基学习器中包含优化过程并不是必需的。

4.3 数据拟合程度对集成学习性能的影响

由前面的讨论，完全随机树、完全随机规则、完全随机决策树桩具有不同的数据拟合程度，即完全随机树>完全随机规则>完全随机决策树桩。

于是可以通过对这 3 种算法的比较来探讨数据拟合程度对使用完全随机策略的集成学习性能的影响。表 2 列出了完全随机决策树桩(CR-DS)、完全随机规则(CR-Rule)和完全随

机树(CR-DT)的集成在 20 个数据上的误差。

由表 2 可见，在 15 个数据集(75%)上 CR-Rule 的误差低于 CR-DS，在 17 个数据集(85%)上 CR-DT 的误差低于 CR-DS 和 CR-Rule。

同时，从表 2 最后一行的平均误差观察得到 3 种算法的性能比较，即 CR-DT>CR-Rule>CR-DS。这说明，数据拟合程度在基于完全随机策略的集成学习中有着重要的作用，拟合程度越高，集成的性能越好。

表 2 不同数据拟合程度的算法的比较

Data set	CR-DS	CR-Rule	CR-DT
balance	0.155 5	0.125 8	0.153 9
breast-c	0.292 4	0.289 6	0.262 2
breast-w	0.055 1	0.133 8	0.026 1
cmc	0.573 0	0.565 8	0.517 9
credit-a	0.231 5	0.179 8	0.128 9
credit-g	0.300 0	0.300 0	0.268 8
cylinder	0.357 4	0.356 7	0.290 3
haberman	0.264 7	0.264 7	0.331 4
heart-c	0.215 4	0.198 6	0.168 8
heart-s	0.234 1	0.229 6	0.166 7
iris	0.138 8	0.080 7	0.050 7
liver	0.420 3	0.417 4	0.288 7
lymph	0.330 4	0.218 4	0.145 0
primarytumor	0.784 6	0.590 1	0.577 2
promoters	0.131 6	0.408 1	0.188 3
spect	0.273 8	0.207 4	0.157 3
spectf	0.272 2	0.269 9	0.127 1
vehicle	0.486 0	0.404 9	0.268 5
yeast	0.620 5	0.644 6	0.398 6
zoo	0.424 4	0.194 4	0.051 2
average	0.328 1	0.304 0	0.228 4

4.4 完全随机策略有效性探讨

由于完全随机决策树桩、完全随机规则和完全随机树的训练过程都不包含任何优化过程，而只是简单地将输入空间进行随机划分。测试样本类分布即以该样本所落划分空间中的训练样本类分布作为估计。

对完全随机算法进行集成后，测试样本的类分布将以该样本可能落到的所有划分空间中的训练样本进行估计。如果测试样本 x 的真实类分布接近 x 的 ε-邻域的类分布，而 x 所在的各个划分空间都包含 x 的 ε-邻域，则在对各个划分空间的类分布进行平均的过程中，x 的 ε-邻域的权重得到了提高，从而使得对 x 的类分布估计变得精确。

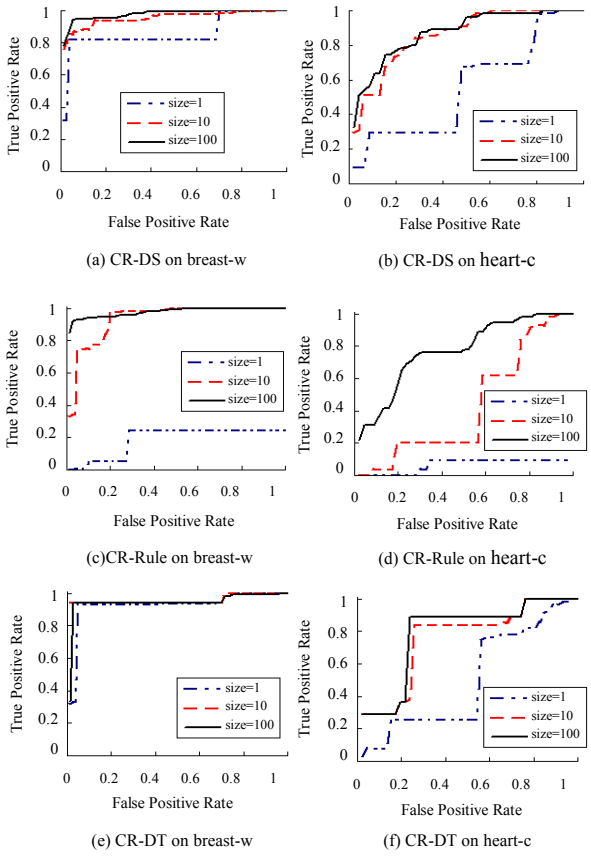
为了验证本解释的正确性，需要观察随着集成规模的增加，对样本分布的估计是否变得更准确。然而，在实验数据集中并不知道每一个样本的真实类分布。因此，这里使用了 ROC[12]曲线来近似得到类分布估计的正确性。

对完全随机决策树桩(CR-DS)、完全随机规则(CR-Rule)和完全随机树(CR-DT)的集成取集成规模(个体学习器数量)为 1、10、100 在 breast-w 和 heart-c 上取两个类别进行实验。实验得到的 ROC 见图 1。

由图 1 可以观察到，随集成规模的增加，整个 ROC 曲线都在提高，而不是只有曲线的一部分得到提高，这说明了当集成规模增加时，对样本类分布的估计变得准确。

从上面的分析可以看出，如果在对各个划分空间的类分布进行平均的过程中，除了 x 的 ε-邻域的权重被提高，还有其它区域的权重同时被提高，则可能有损于对 x 的类分布的估计的精确性。

如此可以说明，对空间的细粒度的随机划分比对空间粗粒度的随机划分能够取得更好的性能，因为细粒度的划分使得在对各个划分空间的类分布进行平均的过程中非 x 的 ε-邻域被包含的可能性较小。



(a) CR-DS on breast-w　(b) CR-DS on heart-c

(c)CR-Rule on breast-w　(d) CR-Rule on heart-c

(e) CR-DT on breast-w　(f) CR-DT on heart-c

图 1 3 种集成算法的 ROC

由图 1 可以观察到，随集成规模的增加，整个 ROC 曲线都在提高，而不是只有曲线的一部分得到提高，这说明了当集成规模增加时，对样本类分布的估计变得准确。

另一方面，从上面的分析可以看出，如果在对各个划分空间的类分布进行平均的过程中，除了 x 的 ε-邻域的权重被提高，还有其它区域的权重同时被提高，则可能有损于对 x 的类分布的估计的精确性。如此可以说明，对空间的细粒度的随机划分比对空间粗粒度的随机划分能够取得更好的性能，因为细粒度的划分使得在对各个划分空间的类分布进行平均的过程中非 x 的 ε-邻域被包含的可能性较小。这就说明了在上一节实验中观察到的现象，即拟合程度越高，集成的性能越好。

5 结束语

本文通过构造 2 种不同数据拟合程度的完全随机算法，即完全随机决策树桩和完全随机规则，分析了在集成学习的框架下完全随机策略的有效性。实验表明完全随机策略在不同数据拟合程度的算法上都是有效的，这说明在集成学习学习的框架下基学习器的优化过程不是必需的。通过对完全随机决策树桩、完全随机规则和完全随机树的比较，发现数据拟合程度越高，集成的性能越好。对完全随机策略有效性的分析得出，对完全随机算法的集成，实际上是对样本真实分布的逼近，并且算法的数据拟合程度越高，对样本真实分布的逼近越好。

（下转第 152 页）

由 George Mason 大学和 SONEX 公司为 US 的军队 CECOM 联合开发。这个专家系统可以自动分析需求的模糊性、完整性、目标冲突、一致性和夸张叙述。

(6)Donald 开发了一种需求波动评价方法(Volatility of Requirements Assessment Method，VRAM)，这种方法是一种辅助决策的结构化方法[2]。VRAM 通过应用需求的历史和需求类型评价其波动。从实际数据中获取波动因素，通过应用层次分析法，对各需求分类的波动性进行评价。该方法用来计算项目的需求波动水平。另外，VRAM 的波动因素，包括前面的需求风险评价方法却没有被考虑到。完整性、正确性、一致性在需求风险文献中是通常都有的，VRAM 也评价了新技术、用户变更需求和需求的不正确分配导致的潜在影响。

以上方法中包括需求风险识别、评价方法以及专门针对需求波动的评价方法。这些方法从不同角度提出了识别问题需求的方法，开发了帮助项目经理评价需求风险的方法，具有一定的实践意义。

4 结束语

需求波动风险的研究在我国还处于起步阶段，目前国内外的研究中不可避免地存在着一些问题：(1)针对性不强。这些分析方法基本都是把各种软件类型作为总体来分析。由于不同软件开发过程中固有的不确定因素和复杂性，这就需要针对不同类别的软件项目开发与之相对应的需求波动风险分析方法。(2)没有与各种软件开发生命周期模型相对应。不同的生命周期模型，所面临的需求波动周期、波动原因是不一样的。比如瀑布模型和原型模型就有很大的区别。(3)可操作性不强。虽然各种方法都声称可以有效识别和评估波动水平，但真正能够容易地用于实践，帮助项目经理分析评价需求波动水平，及时制定预防措施的却很少。对于以上存在的问题，将来的研究可以针对不同软件类型，针对不同的软件生命周期开发模型识别需求波动原因，开发评估需求波动风险的方法，制定有效控制风险的措施。

参考文献

1 Kotanya G， Sommerville I. Requirements Engineering: Processes and Techniques[M]. John Wiley & Sons, Ltd., 1998.

2 Donald M Y. An Early Indicator to Predict Requirements Volatility [D]. Fairfax Virginia: George Mason University, 2001.

3 Nurmuliani N, Zowghi D, Powell S. Analysis of Requirements Volatility During Software Development Life Cycle[C]. Proceedings of the Software Engineering Conference, Australia 2004: 28 -37.

4 Zowghi D, Nurmuliani N. Investigating Requirements Volatility During Software Development: Research in Progress[C]. Proceedings of the 3rd Australian Conference on Requirements Engineering, Geelong, Australia, 1998.

5 Ferreira S. Measuring the Effects of Requirements Volatility on Software Development Projects[D]. Arizona State University, 2002.

6 Houston D X. A Software Project Simulation Model for Risk Management[D]. Arizona State University, 2000.

7 Myers M E. A Knowledge-based System for Managing Software Requirements Volatility[D]. Fairfax Virginia: George Mason University, 1988.

8 Samson D E. Automated Assistance for Software Requirements Definition[D]. Fairfax Virginia: George Mason University, 1989.

9 Armour F J. A Cluster Analysis and Prototyping Approach for the Risk Management of Software Requirements[D]. Fairfax Virginia: George Mason University, 1993.

10 Robinson M J. Risk Assessment in Software Requirements Engineering: An Event Driven Framework[D]. Fairfax Virginia: George Mason University, 1995.

11 Romono J J. A Test-based Risk Identification Method for Requirements Assessment[D]. Fairfax Virginia: George Mason University, 1997.

（上接第 102 页）

参考文献

1 Dietterich T G. Ensemble Learning[M]. The Handbook of Brain Theory and Neural Networks(2nd Edition). Cambridge, MA: MIT Press, 2002.

2 Breiman L. Bagging Predictors[J]. Machine Learning, 1996, 24(2): 123-140.

3 Ho T K. The Random Subspace Method for Constructing Decision Forests[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 1998, 20(8), 832-844.

4 Breiman L. Random Forests[J]. Machine Learning, 2001, 45(1): 5-32.

5 Quinlan J R. C4.5: Programs for Machine Learning[M]. San Mateo, CA: Morgan Kaufmann, 1993.

6 Fan W, Wang H, Yu P S, et al. Is Random Model Better on Its Accuracy and Efficiency[C]. Proceedings of the 3rd IEEE International Conference on Data Mining, Melbourne, 2003: 51-58.

7 Liu F T, Ting K M, Maximizing W F. Tree Diversity by Building Complete-random Decision Trees[C]. Proceedings of the 9th Pacific-asia Conference on Knowledge Discovery and Data Mining, Hanoi, Vietnam, 2005: 605-610.

8 Krogh A, Vedelsby J. Advances in Neural Information Processing Systems 7[M]. Cambridge, MA: MIT Press, 1995: 231-238.

9 Iba W, Langley P. Induction of One-level Decision Trees[C]. Proceedings of the 9th International Conference on Machine Learning, San, Fransisco, CA, 1992: 233-240.

10 Cendrowska J. PRISM: An Algorithm for Inducing Modular Rules[J]. International Journal of Man-machine Studies, 1987, 27(4): 349-370.

11 Blake C L, Keogh E, Merz C J. UCI Repository of Machine Learning Databases[M]. Irvine, CA: University of California, 1998.

12 Bradley A P. The Use of the Area Under the ROC Curve in the Evaluation of Machine Learning Algorithms[J]. Pattern Recognition, 1997, 30(7): 1145-1159.