Effects of Dynamic Subspacing in Random Forest

Md Nasim Adnan(✉) and Md Zahidul Islam

School of Computing and Mathematics, Charles Sturt University, Bathurst, NSW 2795, Australia
{madnan,zislam}@csu.edu.au

Abstract. Due to its simplicity and good performance, Random Forest attains much interest from the research community. The splitting attribute at each node of a decision tree for Random Forest is determined from a predefined number of randomly selected attributes (a subset of the entire attribute set). The size of an attribute subset (subspace) is one of the most important factors that stems multitude of influences over Random Forest. In this paper, we propose a new technique that dynamically determines the size of subspaces based on the relative size of the current data segment to the entire data set. In order to assess the effects of the proposed technique, we conduct experiments involving five widely used data set from the UCI Machine Learning Repository. The experimental results indicate the capability of the proposed technique on improving the ensemble accuracy of Random Forest.

Keywords: Decision tree · Decision forest · Random forest

1 Introduction

Data mining has entered into our day to day life; we now predict the diagnoses of patients, credit approvals/denials and even elections. These predictions are carried out by classifier(s) based on previously known information. Likewise, classifiers are used in business, science, education, security and many other arena. As classifiers enter such influential and sensitive ambit, the importance of improving their efficiency is paramount.

A classifier is a function that maps a set of non-class attributes $\boldsymbol{m} = \{A_1, A_2, ..., A_m\}$ to a predefined class attribute C from an existing data set $\boldsymbol{D}$. A data set generally contains two types of attributes such as numerical (e.g. Age) and categorical (e.g. Gender). Among categorical attributes, one is chosen to be the "class" attribute. All other attributes are termed as "non-class" attributes. A classifier is built from an existing data set (i.e. training data set) where the values of the class attribute are present and then the classifier is applied on unseen/test records to predict their class values.

There are different types of classifiers in literature such as Artificial Neural Networks [25,51,52], Bayesian Classifiers [14,37], Nearest-Neighbor classifiers [26,46], Support Vector Machines [18] and Decision Trees [17,40,41]. Some

G. Cong et al. (Eds.): ADMA 2017, LNAI 10604, pp. 303–312, 2017.
https://doi.org/10.1007/978-3-319-69179-4_21

classifiers such as Artificial Neural Network and Support Vector Machines work similar to a "black box" where they only give predictive results without providing any reasoning for the results [27,32]. On the other hand, a decision tree expresses the patterns that exist in a data set into a flow-chart like representation that closely resembles human reasoning. Each path of the flow-chart represents a logic rule which can be used for knowledge discovery as well as predicting unlabeled records. In this way, decision trees avoid the knowledge discovery bottleneck and thus very popular to the real-world users [38,39].

It is worth to mention that decision trees require no domain knowledge for any parameter setting and therefore more appropriate for exploratory knowledge discovery [21]; and unlike some classifiers (such as Artificial Neural Networks, Nearest-Neighbor Classifiers and Support Vector Machines) decision trees are readily applicable on both categorical and numerical data that further increases their application domain. In addition, decision trees are able to deal with high dimensional, redundant as well as correlated attributes [11,21,33].

Hunt's Concept Learning System (CLS) [23] can be credited as the pioneering work for inducing top-down decision trees. According to CLS, the induction of a decision tree starts by selecting a non-class attribute A_i to split a training data set $\boldsymbol{D}$ into a disjoint set of horizontal partitions [24,40,46]. The purpose of this splitting is to create a purer distribution of class values in the succeeding partitions than the distribution in $\boldsymbol{D}$. The purity of class distribution in succeeding partitions is checked for all contending non-class attributes and the attribute that gives purer class distribution than others is selected as the splitting attribute. The process of selecting the splitting attribute continues recursively in each subsequent partition $\boldsymbol{D}_i$ until either every partition gets the "purest class distribution" or a stopping criterion is satisfied. By "purest class distribution" we mean the presence of a single class value for all records. A stopping criterion can be the minimum number of records that a partition must contain; meaning that if an splitting event creates one or more succeeding partitions with less than the minimum number of records, the splitting is not considered.

A decision tree consists of nodes (denoted by rectangles) and leaves (denoted by ovals) as shown in Fig. 1. The node of a decision tree symbolizes a splitting event where the splitting attribute (label of the node) partitions a data set according to its domain values. As a result, a disjoint set of horizontal segments of the data set are generated and each segment contains one set of domain values of the splitting attribute. For example, in Fig. 1 "Trouble Remembering" is selected as the splitting attribute in the root node. "Trouble Remembering" has two domain values: "Y" and "N" and thus it splits the data set into two disjoint horizontal segments in such as way that the records of one segment contain "Y" value for "Trouble Remembering" attribute and the records of another segment contain "N" value. The domain values of the splitting attribute designated for the respective horizontal segments are represented by the labels of edges leaving the node.

The use of an ensemble of classifiers is a comparatively newer area of research [3–7,20,42]. Interestingly, an ensemble of classifiers is found to be more effective

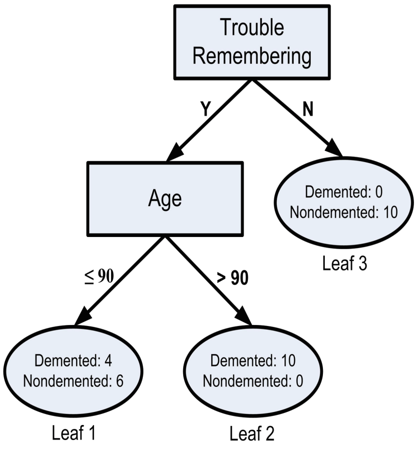

Fig. 1. Decision Tree

for unstable classifiers such as decision trees [46,49]. Decision trees are considered to be an unstable classifier as slight change(s) in a training data set can cause significant differences between the resulting decision trees obtained from the original and modified data sets. A decision forest is an ensemble of decision trees where an individual decision tree acts as a base classifier. The ensemble classification is performed by taking a vote based on the predictions made by each decision tree of the forest [46].

In order to achieve better ensemble accuracy, decision trees to be voted should be as different/diverse as possible; otherwise if they were identical, there could be little or no improvement from the ensemble [44]. Hence, diversity is considered as the cornerstone of ensemble systems and this is the reason why unstable classifiers are good for ensembles. However, to establish the scope of generating too diverse decision trees may be the cause of generating less accurate decision trees as optimization on the two conflicting objectives can be difficult to attain simultaneously [22]. In literature, we find a considerable study on the accuracy-diversity trade-off and find that individual accuracy should not be ignored for diversity [8,28,47].

Several decision forest algorithms exist aiming to generate more diverse as well as accurate decision trees by manipulating the training data set. We explain some of the renowned algorithms as follows.

Bagging: Bagging [15] generates a new training data set $\boldsymbol{D}_i$ where the records of $\boldsymbol{D}_i$ are selected randomly from the original training data set $\boldsymbol{D}$. A new training data set $\boldsymbol{D}_i$ contains the same number of records as in $\boldsymbol{D}$. Thus, some records of $\boldsymbol{D}$ can be selected multiple times and some records may not be selected at all. This approach of generating a new training data set is known as bootstrap sampling. Approximately, 63.2% of the original records are selected in a bootstrap sample and the remaining 36.8% records are repeated [21]. In Bagging, a predefined number ($|T|$) of bootstrap samples $\boldsymbol{D}_1, \boldsymbol{D}_2, ..., \boldsymbol{D}_{|T|}$ are generated using the above mentioned approach. A decision tree induction algorithm is then applied on each bootstrap sample $\boldsymbol{D}_i$ $(i = 1, 2, \ldots, |T|)$ in order to generate $|T|$ number of trees for the forest.

Random Subspace: The Random Subspace algorithm [22] algorithm randomly draws a subset of attributes (subspace) $\boldsymbol{f}$ from the entire attribute space $\boldsymbol{m}$. $\boldsymbol{f}$ can be drawn either at the tree level or at the node level. When selected at the tree level, attributes in $\boldsymbol{f}$ remains the same for each node of a tree; on the other hand attributes in $\boldsymbol{f}$ may differ from one node to another in a tree when selected at the node level. The best attribute in $\boldsymbol{f}$ is determined to be the splitting attribute for the associated node. The Random Subspace algorithm is applied on the original training data set (not on bootstrap samples) for building decision trees.

Random Forest: Random Forest [16] is regarded as a state-of-the-art decision forest building algorithm [12,13] which is technically a combination of Bagging and Random Subspace algorithms. In the simplest form of Random Forest, attributes in $\boldsymbol{f}$ is randomly selected at the node level and the size of $\boldsymbol{f}$ is chosen to be $int(log_2|\boldsymbol{m}|) + 1$ [16] (popularly known as the hyperparameter [13]).

Since its inception in 2001, Random Forest attains much interest from the research community and thereby numerous enhancements have been proposed in recent years [2–5,9,12,45,53]. In particular, the selection of more suitable subspace ($\boldsymbol{f}$) invites much attention [13,19,20,36,43,48,50]. In Forest-RK [13], the authors proposed for random selection of $|\boldsymbol{f}|$ between 1 to $|\boldsymbol{m}|$ while in Extremely Randomized Trees [20], $|\boldsymbol{f}|$ is chosen to be $\sqrt{|\boldsymbol{m}|}$ for the classification problem. The Extremely Randomized Trees algorithm improvises more randomness for numerical attributes by selecting the cut-points fully at random while ensuring a minimum number of records in either sides of a cut-point. Another algorithm [19] suggested setting the cut point midway between two training records that had been picked randomly.

In [50], the authors applied the stratified sampling of attributes for Random Forest to deal with high dimensional data set. The key idea behind the stratified sampling is to divide the attributes $\boldsymbol{m}$ into two groups. One group will contain the good attributes $\boldsymbol{m}_G$ and the other group will contain the bad attributes $\boldsymbol{m}_B$. The attributes having the informativeness capacity higher than the average informativeness capacity are placed in the group of good attributes $\boldsymbol{m}_G$ and all other attributes are placed in the group of bad attributes $\boldsymbol{m}_B$. Then $int(log_2|\boldsymbol{m}|) + 1$ number of attributes are selected randomly from each group in proportion to the size of the groups.

We understand that with the traversal from the root node down the tree, the number of records in data segments become smaller and smaller as a result of the recursive partitioning [1,24,46]. A data segment is partitioned according to its splitting attributes domain values and thus the resultant partitions tend to have very weak relationship with the same attribute (specially when the attribute is a categorical attribute). As shown earlier, in Fig. 1 "Trouble Remembering" is selected as the splitting attribute in the root node. "Trouble Remembering" has two domain values: "Y" and "N" and thus it splits the data set into two disjoint horizontal segments in such as way that the records of one segment contain “Y” value for “Trouble Remembering” attribute and the records of another segment contain “N” value. Hence, any of the two disjoint horizontal segments can not be split further by the “Trouble Remembering” attribute. As a consequence, fewer attributes become relevant for a data segment that has been generated through a number of partitioning involving different splitting attributes and hence the probability of drawing relevant attributes in $\boldsymbol{f}$ becomes progressively lower down the tree.

We now understand that if this problem of Random Forest is not addressed accordingly, poor-quality splitting attributes may be selected down the tree from which poor-quality partitions may generated. This may decrease individual accuracies of trees in a forest which in turn may affect the ensemble accuracy negatively. Yet, none of the above mentioned variants on subspacing of Random Forest addresses this issue. In this paper, we propose a new technique that dynamically increases the size of $\boldsymbol{f}$ down the tree based on the relative size of the current data segment to the entire training data set so that the probability of drawing relevant attributes in $\boldsymbol{f}$ does not get decreased.

The remainder of this paper is organized as follows: In Sect. 2 we explain the proposed technique. Section 3 discusses the experimental results in detail. Finally, we offer some concluding remarks in Sect. 4.

2 Our Technique

In the proposed technique, the number of attributes in $\boldsymbol{f}$ is dynamically increased with the decrease of records in the current data segment according to Eq. 1:

$$int(log_2(|\boldsymbol{m}| \times \frac{|\boldsymbol{D}|}{|\boldsymbol{D}_i|})) + 1 \tag{1}$$

Here, $|\boldsymbol{D}_i|$ denotes the number of records present in the current data segment and $|\boldsymbol{D}|$ denotes the number of records present in the entire training data set. The term $(\frac{|\boldsymbol{D}|}{|\boldsymbol{D}_i|})$ dynamically increases the number of attributes in $\boldsymbol{f}$ with the relative decrease of records in $\boldsymbol{D}_i$ to $\boldsymbol{D}$. For example, let $\boldsymbol{D} = 1000$ with $\boldsymbol{m} = 16$ and let $\boldsymbol{D}_i = 50$. Conventionally, $|\boldsymbol{f}|$ will remain as 5 for the $\boldsymbol{D}_i$ with 50 records in Random Forest even if the $\boldsymbol{D}_i$ is a result of a number of partitioning involving several splitting attributes. However, according to the proposed technique $|\boldsymbol{f}|$ is increased to 9 in this case.

Random Forest is modified through the proposed technique and its new form is presented in the following.

```
for ($i = 1$ to $|T|$) do /* $|T|$ is the number of trees in Random Forest */
    Step 1: Generate a bootstrap sample from the training data set.
    Step 2: Generate a decision tree from the bootstrap sample with subspaces generated from the proposed technique.
end for.
```

3 Experimental Results

We conduct the experimentation on five well known data sets that are publicly available from the UCI Machine Learning Repository [31]. We already know, when a data segment is partitioned by the splitting attribute, the resultant partitions tend to have very weak relationship with the splitting attribute specially when the attribute is categorical. Hence, in order to exhibit the effectiveness of the proposed technique we select all five data sets with all-categorical attributes. The data sets are listed in Table 1.

Table 1. Description of the data sets

Data Set name (DS)	Non-Class attributes	Records	Distinct class Values
Car Evaluation (CE)	06	1728	4
Chess (CHS)	36	3196	2
Hayes-Roth (HR)	04	132	3
Nursery (NUR)	08	12960	5
Tic-Tac-Toe Endgame (TTT)	09	958	2

We implement Random Forest (RF) and the Modified Random Forest (MRF) maintaining the following settings. We use Gini Index as a measure of classification capacity in accordance with RF. The minimum Gini Index value is set to 0.01 for any attribute to qualify for splitting a node. Each leaf node of a tree requires at least two records and no further post-pruning is applied. We apply majority voting in order to aggregate results for the forests. The entire experimentation is conducted by a single machine with Intel(R) 3.4 GHz processor and 4 GB Main Memory (RAM) running under 64-bit Windows 8.1 Operating System. All the results reported in this paper are obtained using 10-fold-cross-validation (10-CV) [10,29,30] for every data set.

In 10-CV, at first a data set is randomly divided into 10 horizontal segments/partitions. The segments are mutually exclusive meaning that they do not have any overlapping records. Each segment in turn is considered to be the testing data set (out of bag samples) while for the same turn the remaining nine segments are considered to be the training data set. Thus, we get 10 training data sets and 10 corresponding testing data sets. A classifier is then built from each training data set and its performance indicator (such as prediction accuracy) is tested on each corresponding testing data set. The average result obtained from all 10 testing data sets is termed to be 10-CV. The best results reported in this chapter are stressed through **bold-face**.

Ensemble Accuracy (EA) is one of the most important performance indicators for any decision forest algorithm [2,3,6]. In Table 2, we present the EA (in percent) of RF and MRF for all data sets considered.

Table 2. EA

DS	RF	MRF
CE	92.8	**93.9**
CHS	94.9	**97.0**
HR	71.1	**74.2**
NUR	95.0	**97.4**
TTT	84.5	**90.6**
Avg.	87.7	**90.6**

From Table 2, we see that MRF provides the best EA for all data sets considered. We know, EA of a decision forest mainly depends on two factors: individual tree accuracy and diversity among the trees. Hence, in order to explain reasons behind the improvement, we first compute individual accuracy (in percent) of each tree in a forest to compute the Average Individual Accuracy (AIA) for the forest as was done in literature [3–5,9].

Margineantu and Dietterich [34] proposed a visualization method for classifier ensembles called Kappa that has been used extensively as a measure of diversity in literature [9,29,35,42,53]. Kappa typically estimates the diversity between *two trees* T_i and T_j. Diversity among *more than two trees* is computed by first computing the Kappa (K) value of a single tree T_i with the ensemble of trees except the tree in consideration (i.e. with $T - T_i$ where T is the set of all trees in the forest) [9]. The combined prediction of the forest (computed through the majority voting) can be regarded as a single tree T_j. Then Kappa is computed between T_i and T_j as shown in Eq. 2, where $Pr(a)$ is the probability of the observed agreement between two classifiers T_i and T_j, and $Pr(e)$ is the probability of the random agreement between T_i and T_j. Once the Kappa for every single tree T_i of a decision forest is computed we then compute the Average Individual Kappa (AIK) for the forest.

$$K = \frac{Pr(a) - Pr(e)}{1 - Pr(e)} \tag{2}$$

According to Eq. 2, when $K = 1$ two trees agree on every example. When $K = 0$ they disagree on every examples except agreements by chance. Rarely, K can be negative when two trees disagree in most examples suppressing agreements by chance. Thus, the lower the Kappa (K)/AIK value, the higher the diversity. The results related to AIA and AIK are reported in Table 3.

From Table 3, we observe that MRF achieves higher AIA for all data sets considered. This indicates that trees generated by MRF are individually more accurate; and we understand that this is due to the fact that MRF can effort better-quality splitting down the tree.

Table 3. AIA and AIK

	AIA		AIK	
DS	RF	MRF	RF	MRF
CE	83.5	**88.9**	**0.68**	0.82
CHS	68.0	**71.9**	**0.49**	0.56
HR	55.3	**56.2**	**0.33**	0.34
NUR	71.6	**80.8**	**0.65**	0.79
TTT	54.9	**59.4**	**0.32**	0.38
Avg.	66.6	**71.4**	**0.49**	0.58

4 Conclusion

In this paper, we propose a novel technique that dynamically increases the size of subspaces down the tree based on the relative size of the current data segment to the entire training data set so that the probability of drawing relevant attributes in subspaces does not get decreased. Aided by the proposed technique, Random Forest is able to generate better-quality trees (individually more accurate). This in turn helps Random Forest to attain higher ensemble accuracy. In future, we plan to apply the proposed technique on other decision forest algorithms.

References

1. Adnan, M.N., Islam, M.Z.: ComboSplit: Combining various splitting criteria for building a single decision tree. In: Proceedings of the International Conference on Artificial Intelligence and Pattern Recognition, pp. 1–8 (2014)
2. Adnan, M.N., Islam, M.Z.: A comprehensive method for attribute space extension for random forest. In: Proceedings of 17th International Conference on Computer and Information Technology, Dec (2014)
3. Adnan, M.N., Islam, M.Z.: Complement random forest. In: Proceedings of the 13th Australasian Data Mining Conference (AusDM), pp. 89–97 (2015)
4. Adnan, M.N., Islam, M.Z.: Improving the random forest algorithm by randomly varying the size of the bootstrap samples for low dimensional data sets. In: Proceedings of the European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning, pp. 391–396 (2015)
5. Adnan, M.N., Islam, M.Z.: One-vs-all binarization technique in the context of random forest. In: Proceedings of the European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning, pp. 385–390 (2015)
6. Adnan, M.N., Islam, M.Z.: Forest CERN: A New Decision Forest Building Technique. In: Bailey, J., Khan, L., Washio, T., Dobbie, G., Huang, J.Z., Wang, R. (eds.) PAKDD 2016. LNCS (LNAI), vol. 9651, pp. 304–315. Springer, Cham (2016). doi:10.1007/978-3-319-31753-3_25
7. Adnan, M.N., Islam, M.Z.: Knowledge discovery from a data set on dementia through decision forest. In: Proceedings of the 14th Australasian Data Mining Conference (AusDM) (2016). Accepted

8. Adnan, M.N., Islam, M.Z.: Optimizing the number of trees in a decision forest to discover a subforest with high ensemble accuracy using a genetic algorithm. Knowl. Based Syst. **110**, 86–97 (2016)
9. Amasyali, M.F., Ersoy, O.K.: Classifier ensembles with the extended space forest. IEEE Trans. Knowl. Data Eng. **16**, 145–153 (2014)
10. Arlot, S.: A survey of cross-validation procedures for model selection. Stat. Surv. **4**, 40–79 (2010)
11. Barros, R.C., Basgalupp, M.P., de Carvalho, A.C.P.L.F., Freitas, A.A.: A survey of evolutionary algorithm for decision tree induction. IEEE Trans. Syst. Man Cybern. Part C Appl. Rev. **42**(3), 291–312 (2012)
12. Bernard, S., Adam, S., Heutte, L.: Dynamic random forests. Pattern Recognit. Lett. **33**, 1580–1586 (2012)
13. Bernard, S., Heutte, L., Adam, S.: Forest-RK: A New Random Forest Induction Method. In: Huang, D.-S., Wunsch, D.C., Levine, D.S., Jo, K.-H. (eds.) ICIC 2008. LNCS, vol. 5227, pp. 430–437. Springer, Heidelberg (2008). doi:10.1007/978-3-540-85984-0_52
14. Bishop, C.M.: Pattern Recognition and Machine Learning. Springer, New York (2008)
15. Breiman, L.: Bagging predictors. Mach. Learn. **24**, 123–140 (1996)
16. Breiman, L.: Random forests. Mach. Learn. **45**, 5–32 (2001)
17. Breiman, L., Friedman, J., Olshen, R., Stone, C.: Classification and Regression Trees. Wadsworth International Group. San Diego (1985)
18. Burges, C.J.C.: A tutorial on support vector machines for pattern recognition. Data Min. Knowl. Discov. **2**, 121–167 (1998)
19. Cutler, A., Zhao, G.: Pert: perfect random tree ensembles. Comput. Sci. Stat. **33**, 204–497 (2001)
20. Geurts, P., Ernst, D., Wehenkel, L.: Extremely randomized trees. Mach. Learn. **63**, 3–42 (2006)
21. Han, J., Kamber, M.: Data Mining Concepts and Techniques. Morgan Kaufmann Publishers (2006)
22. Ho, T.K.: The random subspace method for constructing decision forests. IEEE Trans. Pattern Anal. Mach. Intell. **20**, 832–844 (1998)
23. Hunt, E., Marin, J., Stone, P.: Experiments in Induction. Academic Press, New York (1966)
24. Islam, M.Z., Giggins, H.: Knowledge discovery through sysfor - a systematically developed forest of multiple decision trees. In: Proceedings of the 9th Australlasian Data Mining Conference (2011)
25. Jain, A.K., Mao, J.: Artificial neural network: a tutorial. Computer **29**(3), 31–44 (1996)
26. Kataria, A., Singh, M.D.: A review of data classification using k-nearest neighbour algorithm. Int. J. Emerg. Technol. Adv. Eng. **3**(6), 354–360 (2013)
27. Kotsiantis, S.B.: Decision trees: a recent overview. Artif. Intell. Rev. **39**, 261–283 (2013)
28. Kuncheva, L.I., Whitaker, C.J.: Measures of diversity in classifier ensembles and their relationship with ensemble accuracy. Mach. Learn. **51**, 181–207 (2003)
29. Kurgan, L.A., Cios, K.J.: Caim discretization algorithm. IEEE Trans. Knowl. Data Eng. **16**, 145–153 (2004)
30. Li, J., Liu, H.: Ensembles of cascading trees. In: Proceedings of the third IEEE International Conference on Data Mining, pp. 585–588. (2003)
31. Lichman, M.: UCI machine learning repository. http://archive.ics.uci.edu/ml/datasets.html. Accessed 15 Mar 2016

32. Liu, S., Patel, R.Y., Daga, P.R., Liu, H., Fu, G., Doerksen, R.J., Chen, Y., Wilkins, D.E.: Combined rule extraction and feature elimination in supervised classification. IEEE Trans. NanoBioscience **11**(3), 228–236 (2012)
33. Maimon, O., Rokach, L. (eds.): The Data Mining and Knowledge Discovery Handbook. Springer, New York (2005)
34. Margineantu, D.D., Dietterich, T.G.: Pruning adaptive boosting. In: Proceedings of the 14th International Conference on Machine Learning, pp. 211–218 (1997)
35. Maudes, J., Rodriguez, J.J., Osorio, C.G., Pedrajas, N.G.: Random feature weights for decision tree ensemble construction. Inf. Fusion **13**, 20–30 (2012)
36. Menze, B., Petrich, W., Hamprecht, F.: Multivariate feature selection and hierarchical classification for infrared spectroscopy: serum-based detection of bovine spongiform encephalopathy. Anal. Bioanal. Chem. **387**, 1801–1807 (2007)
37. Mitchell, T.M.: Machine Learning. McGraw-Hill, New York (1997)
38. Murthy, S.K.: On growing better decision trees from data. Ph.D. thesis, The Johns Hopkins University, Baltimore, Maryland (1997)
39. Murthy, S.K.: Automatic construction of decision trees from data: a multi-disciplinary survey. Data Min. Knowl. Discov. **2**, 345–389 (1998)
40. Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers, San Mateo (1993)
41. Quinlan, J.R.: Improved use of continuous attributes in c4.5. J. Artif. Intell. Res. **4**, 77–90 (1996)
42. Rodriguez, J.J., Kuncheva, L.I., Alonso, C.J.: Rotation forest: a new classifier ensemble method. IEEE Trans. Pattern Anal. Mach. Intell. **28**, 1619–1630 (2006)
43. Saeys, Y., Abeel, T., Van de Peer, Y.: Robust feature selection using ensemble feature selection techniques. In: Daelemans, W., Goethals, B., Morik, K. (eds.) ECML PKDD 2008. LNCS, vol. 5212, pp. 313–325. Springer, Heidelberg (2008). doi:10.1007/978-3-540-87481-2_21
44. Shipp, C.A., Kuncheva, L.I.: Relationships between combination methods and measures of diversity in combining classifiers. Inf. Fusion **3**, 135–148 (2002)
45. Robnik-Šikonja, M.: Improving Random Forests. In: Boulicaut, J.-F., Esposito, F., Giannotti, F., Pedreschi, D. (eds.) ECML 2004. LNCS (LNAI), vol. 3201, pp. 359–370. Springer, Heidelberg (2004). doi:10.1007/978-3-540-30115-8_34
46. Tan, P.N., Steinbach, M., Kumar, V.: Introduction to Data Mining. Pearson Education, Boston(2006)
47. Tang, E.K., Suganthan, P.N., Yao, X.: An analysis of diversity measures. Mach. Learn. **65**, 247–271 (2006)
48. Tuv, E., Borisov, A., Runger, G., Torkkola, K.: Feature selection with ensembles, artificial variables, and redundancy elimination. J. Mach. Learn. Res. **10**, 1341–1366 (2009)
49. Williams, G.J.: Combining decision trees: Initial results from the MIL algorithm. In: Proceedings of the First Australian Joint Artificial Intelligence Conference, pp. 273–289, Sydney, Australia, 2–4 Nov 1988, 1987
50. Ye, Y., Wu, Q., Huang, J.Z., Ng, M.K., Li, X.: Stratified sampling of feature subspace selection in random forests for high dimensional data. Pattern Recognit. **46**, 769–787 (2014)
51. Zhang, G., Patuwo, B.E., Hu, M.Y.: Forecasting with artificial neural networks: the state of the art. Int. J. Forecast. **14**, 35–62 (1998)
52. Zhang, G.P.: Neural networks for classification: a survey. IEEE Trans. Syst. Man Cybern. **30**, 451–462 (2000)
53. Zhang, L., Suganthan, P.N.: Random forests with ensemble of feature spaces. Pattern Recognit. **47**, 3429–3437 (2014)