TAF Journal of Applied and Physical Sciences JAPS
2018, **4**(2): 47-54

PRIMARY RESEARCH

The Application of tree-based ML algorithm in steel plates Ffaults identification

Jiahui Chen *
College of Materials Science and Engineering, China Northeastern University, Shenyang, China

Keywords
Steel plate
Fault classification
Machine learning
UCI
Decision tree
Adaboostin
Bagging
Random forest

Abstract
This research aims to apply a series of classical machine learning algorithms based on decision trees (Decision Tree, Adaboosting, Bagging, Random Forest) to verify the ten-fold cross-validation of the steel plate fault data. The source of the data set was the Research Center of Sciences of Communication in Italy and has been used two times by M Buscema when it is provided [15, 16]. The data set includes 7 different types of steel plate faults: Pastry, Z_Scratch, K_Scatch, Stains, Dirtiness, Bumps, and Other Faults. It is found that the Bagging algorithm outperforms the other methods and achieves 96.30% and 90% accuracy on the training and testing set, respectively. This will allow us to find abnormalities on the surface of the steel plate timely and reduce losses. Based on these algorithms, we can cooperate with iron and steel practitioners to design more appropriate algorithms to achieve higher recognition accuracy in the future.

Received: 16 April 2018
Accepted: 15 May 2018
Published: 5 June 2018

I. INTRODUCTION

The faults of the steel plate refer to the bubbles, scratches, cracks and shrinkage holes, which are mainly caused by the technology in the production process. The faults of the steel plate not only affect the appearance of the products, but also cause the stress concentration and cracking, which influences the plasticity and toughness of the steel, and reduces the performance and service life of the steel. The production of faulty products will bring serious economic losses to the company. Moreover, it may lead to more serious accidents if the faulty steel flows into the market. Thus, it is essential for producers to control the production of faulty steel strictly and choose the right method based on the previous heat treatment experience. However, it is impossible to avoid the occurrence of various faults under the complicated production environments. It is important to discover and identify faults quickly, change the technology in time and reduce the occurrence of faults. The traditional detection methods include visual inspection, metallographic examination and scanning electron microscope [1, 2, 3, 4]. However, these traditional methods are unstable and inaccurate, since they are relied on the subjective judgement of inspectors.

In recent years, machine learning has been widely applied in all walks of life. Machine learning transforms the process of human thinking and induction into computer learning and modeling. With the development of computational science, machine learning algorithms and techniques can handle tens of thousands of data very efficiently [5, 6], which increase computational efficiency greatly in comparison to traditional methods. Decision tree [7, 8] is a classical and popular machine learning model in this field. There are also a series of model based on decision tree (for example, Adaboosting, Bagging and Random Forest) are widely used for their good performance and strong learning ability.

In order to use the efficiency of machine learning to replace

*Corresponding author: Jiahui Chen
†email: chenjh@stumail.neu.edu.cn

the traditional manual recognition, we use these classical algorithms and the data of steel plate surface to solve the defect classification problem. Previously, some scholars have tried to apply machine algorithms to more industrial applications, such as surface fault detection signal processing of cold-rolled steel plate, rail bottom fault detection [1, 2]. We apply a series of classical machine learning algorithms based on decision trees (Decision Tree, Adaboosting, Bagging, Random Forest) to model the steel plate fault data. It is found that Bagging algorithm outperforms the other methods and achieves 96.30% and 90% accuracy on the training and testing set, respectively. This will allow us to find abnormalities on the surface of the steel plate timely and reduce losses.

A decision tree is a tool to help decision making by using a tree-like graph to model the decision procedure and possible results. It is a way to show an algorithm that only contains conditional control statements. A single decision tree can achieve good results for data with few conditions. However, the effect of a single decision tree is very limited and unstable for classification problems with complexities and large data. Therefore, Kearns and Valiant raised the idea of Boosting: A set of weak learners may create a single strong learner [9]. The definition of a weak learner is a classifier that is only slightly related to the real classification. However, a strong learner is defined to be a classifier that is closely connected with the real classification. Robert Schapire answered the question of Kearns and Valiant in his paper [10], which had a great impact on machine learning and brought great progress to boosting [11]. Adaboosting and Bagging are two typical boosting algorithms, which are widely applied. Adaboosting (Adaptive Boosting) was created by Yoav Freund and Robert Schapire [12]. Adaboosting learn a set of weak classifiers or basic classifiers from training data. The strong classifier is obtained by weighted sum of weak classifiers. In 1994, Breiman [13] proposed the idea of Bagging (Bootstrap aggregating) to improve classification by combining classifications of randomly generated training sets. It helps to reduce variance and avoid overfitting. Random Forest is a typical Bagging algorithm, but it is different from the classic Bagging. It not only extract instance randomly, but also extracts variables randomly, which was proposed by Tin Kam Ho [14] for the first time.

The rest of the article is organized as follows. In Section 2, we introduce the steel plate fault data briefly and the date preprocessing procedure. In Section 3, we build fault detection models with a series of aforementioned tree-based machine learning models, including Decision tree, Adaboosting, Bagging and Random Forest. The cross-validation results of the four models are compared and analyzed in different perspective. The performances of these models are further discussed in Section4. Finally, we make a brief conclusion about our results and discuss the advantages and disadvantages of our methods.

II. DATA DESCRIPTION

The Steel Plates Faults Data Set are public available at the UCI (University of California Irvine) Machine Learning Repository. The source of the data set was the Research Center of Sciences of Communication in Italy and has been used two times by M Buscema when it is provided [15, 16]. The data set includes 7 different types of steel plate faults: Pastry, Z_Scratch, K_Scatch, Stains, Dirtiness, Bumps and Other Faults. Six of these seven variables represent the type of defect on the steel surface. There are seven common defects on the surface of steel, such as cracks, scratches, folding, scarring, end burrs, etc. These are often caused by accidents in the production process.

The machine vision system is characterized by improving the flexibility and automation of production. In some dangerous working environments that are not suitable for people's work or where artificial vision is difficult to meet the requirements, machine vision is often used to replace artificial vision. At the same time, in high-volume industrial production process, manual visual inspection is used to check product quality with low efficiency and low precision. Machine vision inspection methods can greatly improve production efficiency and automation of production. Moreover, machine vision is easy to realize information integration, and is the basic technology to realize computer integrated manufacturing. The product can be measured, guided, tested, and identified on the fastest production line, and the production tasks can be completed with quality and quantity. We can fix the camera on the production line and analyze the images taken in real time through machine vision. Different defects will present different parameters. The Steel Plates Faults Data Set should be parameters extracted from machine vision.

The 1941 instances consist of 27 variables and one steel plate fault. There are 673 instances of Other Faults, which are not classified clearly. We drop out these instances, since the huge number of uncertain class influence our modeling significantly. Thus, 1268 instances from the other six classes are left in the data set. The 27 attributes and the number of each class are listed in Table 1 and Table 2 respectively. For more information about the data and the complete data set, please refer to https://bit.ly/2LpKvyQ. The 1268 instances are divided into training set and test set randomly. Finally, we arrive at a training set with 918 instances and a testing set with 350 instances.

TABLE 1
ATTRIBUTES OF STEEL PLATES FAULTS DATA SET

Attribute					
Attribute 1	X_Minimum	Numerical	14	Steel_Plate_Thickness	Numerical
2	X_Maximum	Numerical	15	Edges_Index	Numerical
3	Y_Minimum	Numerical	16	Empty_Index	Numerical
4	Y_Maximum	Numerical	17	Square_Index	Numerical
5	Pixels_Areas	Numerical	18	Outside_X_Index	Numerical
6	X_Perimeter	Numerical	19	Edges_X_Index	Numerical
7	Y_Perimeter	Numerical	20	Edges_Y_Index	Numerical
8	Sum_of_Luminosity	Numerical	21	Outside_Global_Index	Numerical
9	Minimum_of_Luminosity	Numerical	22	Log of Areas	Numerical
10	Maximum_of_Luminosity	Numerical	23	Log_X_Index	Numerical
11	Length_of_Conveyer	Numerical	24	Log_Y_Index	Numerical
12	Type of Steel_A300	Qualitative	25	Orientation_Index	Numerical
13	Type of Steel_A400	Qualitative	26	Luminosity_Index	Numerical
			27	Sigmoid of Areas	Numerical

TABLE 2
FAULTS TYPE OF STEEL PLATE

PASTRY	Z_SCRATCH	K_SCATCH	STAINS	DIRTINESS	BUMPS
158	190	391	72	55	402

III. MODELS

A. *Decision Tree*

Decision Tree is a basic classification and regression method [17]. The classification decision tree model is a tree structure describing the classification of instances. A decision tree is composed of nodes and directed edges. Nodes can be divided into two types: internal node and leaf node. An internal node represents a feature or attribute, and a leaf node represents a class. The frequently used algorithms in the decision tree are ID3, C4.5, and CART.

The depth of the tree is an important parameter in the Decision Tree. It reflects the tradeoff between the ability to fit the underlying structure of the data and the predictive performance. If the depth of tree is too deep, overfitting can occur namely, it would fit the training data well, but it may perform terrible in testing data. We can prune the tree from bottom to top to make the tree simpler and have better generalization ability. Cross Validation (CV) [18] is commonly used to select the complexity parameter (here, the depth of tree) in machine learning fields. In K-fold cross validation, the original samples are randomly divided into subsamples of equal size K. A single subsample is reserved as the data for the validation model, and the other K-1 samples are used for training. The cross-validation is repeated K times. Each sub-sample is verified once and the K-averaged results or other combinations are used to get a single estimate of model prediction performance. We use ten-fold cross validation in this paper to select the complexity parameter (here, the depth of tree).

The cross validation error (CV Error) for trees with different depth is computed based on the training data and the results are presented in Figure 1. We also evaluate the training error with the training data and the testing error with the testing set, respectively. The results are also shown in Figure 1. With the increase of tree depth, the training error decreases gradually and overfitting will occur when the depth continues to increase. The testing error and training error are both high when the depth is small, which implies the underfitting of tree model. When the depth is equal to 7, we see that the cross validation error reaches the minimum, which conforms to the bias-variance dilemma [19]. When training is insufficient, the learner's ability to fit is not strong enough, and the bias dominates the generalized error rate. As training degree deepens, the fitting ability of the classifier increases gradually, and the variance gradually leads to the generalized error rate. Thus, we choose the decision tree with depth equal to 7 as our final classifier. Then we fit a decision tree with a depth of 7 with the 918 training data and use the remaining 350 testing data to evaluate the performance. The accuracy on the training set and the testing set are 93.57% and 85.43%, respectively, as shown in Table 3.

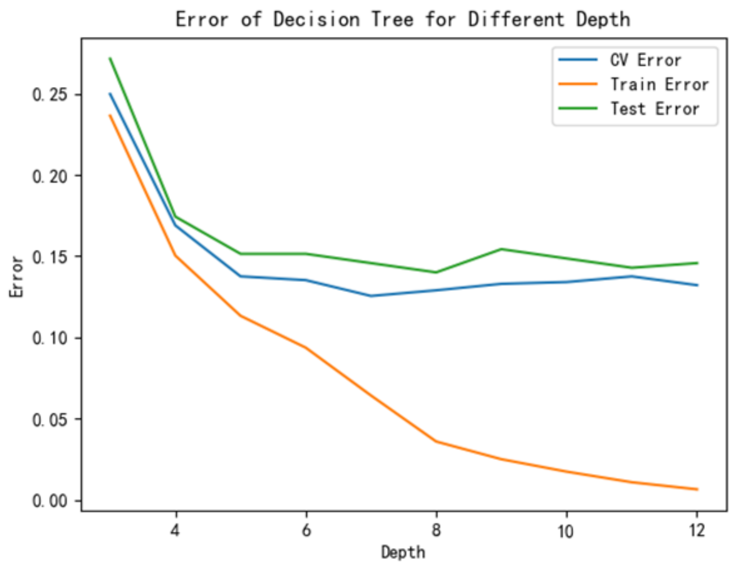

Fig. 1. Error of decision tree for different depth

B. *Adaboosting*

Adaboosting (Adaptive Boosting) is a machine learning meta-algorithm formulated by Yoav Freund and Robert Schapire. Its core idea is to train different weak classifiers for the same training set, then assemble these weak classifiers to form a stronger classifier. In each round, the weight of the samples misclassified by the previous weak classifier is increased [20]. Adaboosting makes good use of weak classifiers for cascading and has high accuracy.

We use decision trees as the weak classifiers in Adaboosting method. In order to keep consistent with the decision tree above, the depth of each weak classifier is set as 7. Similarly, we compute the ten-fold cross validation error of Adaboosting models with different number of classifiers (decision trees). The results are shown in Figure 2. The training error and testing error are also computed and plotted in Figure 2. With the increasing number of weak classifiers, the training set error approaches 0% quickly. The accuracy of the training set is improved significantly, compared with a single decision tree. Cross validation error first decreases and then rises, resulting from the bias-variance dilemma. The testing error of Adaboosting is lower than that of a single decision tree, which implies the improvement of the prediction accuracy. However, the code runs much longer than the decision tree model. We choose the number of weak classifiers as 15 according to the cross validation error, and the corresponding training error and setting error is 100.00% and 88.57%, respectively, as shown in Table 3.

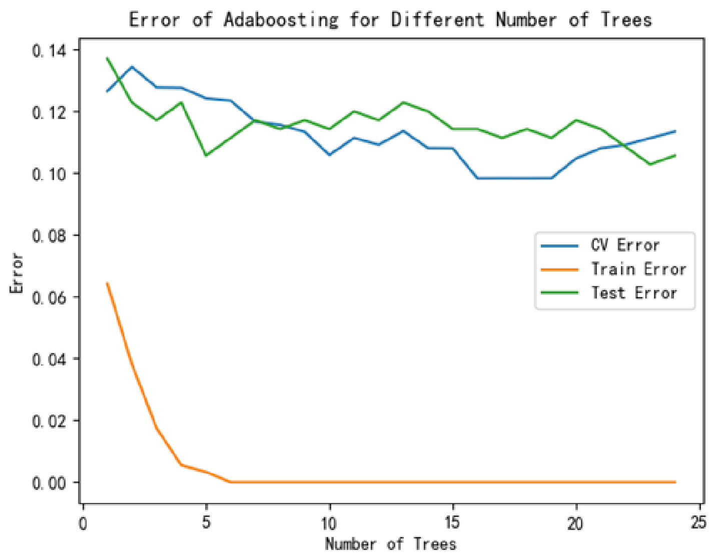

Fig. 2. Error of adaboosting for different number of trees

C. Bagging

Bagging is another machine learning meta-algorithm applied to statistical classification and regression. It aims at improving the stability and accuracy of machine learning algorithms by constructing a series of predictive functions and combining them into a prediction function. Bagging and Boosting algorithm looks similar, but the training method of base classifier is totally different [7]. Given a data set containing m instances, we first take an instance into the data set randomly. After m random resampling operations, we get a data set with m instances. Some instances may appear several times in the data set, while others may never appear. The training set of Bagging algorithm is obtained from the original data set with put back sampling. Each base classifier is independent and parallel. Since the training method of each base classifier is independent and identical, equal-weighted strategy is used to vote by the classifier.

In order to make a fair comparison of the characteristics of each model, we also use a decision tree with a depth of 7 as our base classifier. Similarly, the ten-fold cross validation error, training error and testing error for different number of base classifiers are evaluated, and the results are presented in Figure 3. The error rate in the training set also drops very rapidly. The training error of Bagging is about 4% when the number of base classifiers is large, while the training error is near to 0%. This suggests that Bagging seems to be a better choice to prevent overfitting in the learning process. The cross validation error and testing error decrease quickly and then stabilizes gradually. According to the cross validation error, we select the number of classifier as 18. The corresponding Bagging models are fitted, and the accuracy on the training set and the testing set is 96.30% and 90.00%, respectively, as shown in Table 2. Compared with Adaboosting, Bagging achieves higher accuracy on the testing data set.

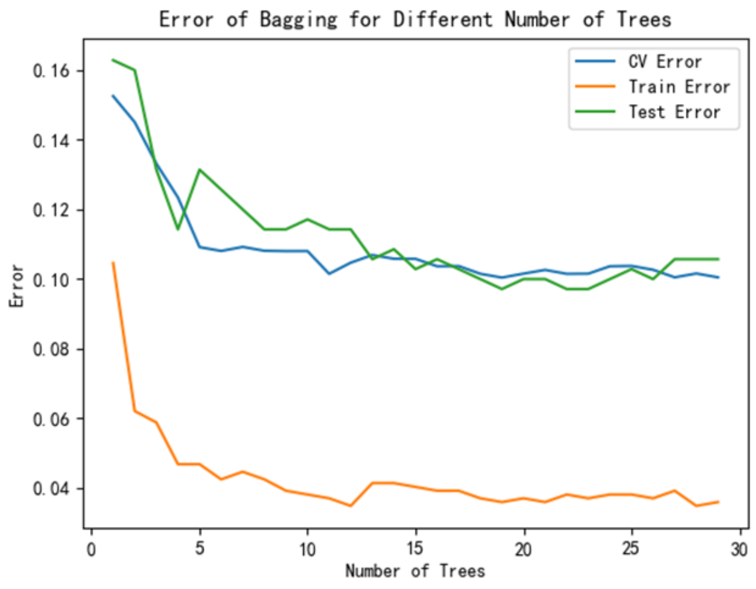

Fig. 3. Error of Bagging for different number of trees

D. Random Forest

Random Forest is an extended variant of Bagging [21]. Random Forest is based on the decision tree to build Bagging, and further introduces random attribute selection during the training process. In the progress of Random Forest training, we not only sample instances with put back sampling, but also select variables randomly in each round. Random Forest is simple, easy to implement and has low computational complexity, which show superior performance in many cases. Because the diversity of the base classifier in the Random Forest not only comes from sample disturbances but also from attribute disturbances, it further improves the generalization performance of Random Forest. The Random Forest with different number of trees are fitted, and the corresponding cross validation error, training error and testing error are evaluated. The results are shown in Figure 4. Compared with the other three models, we see that the cross validation error and the test error of Random Forest decrease faster. As the number of trees continues to increase, it still shows a downward trend. Different from Bagging, the training error decreases to 0% when the number of trees is large enough. When the number of classifier is 18, the accuracy of the training set and the testing set is 100.00% and 90.29%, respectively, as shown in Table 2. Because Random Forest samples both instances and attributes, it reduces the amount of learning tasks significantly. In terms of running time, it is obvious that Random Forest is faster than Adaboosting and Bagging.

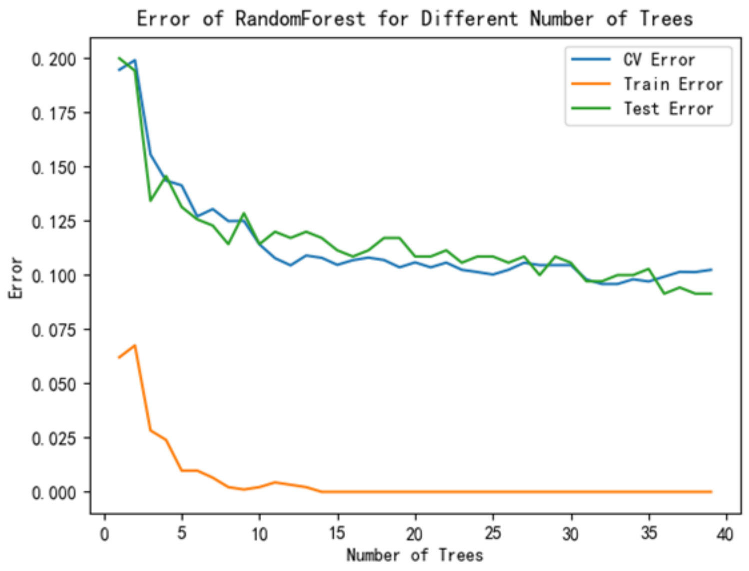

Fig. 4. Error of random forest for different number of trees

IV. MODEL COMPARISON

A. Results

The train accuracy, test accuracy, training time and number of trees of aforementioned four methods are listed in Table 3. As we need to monitor the surface of steel plate constantly in production process, our model needs to be both accurate and efficient. In terms of training time, a single decision tree is most time-saving. However, the accuracy of a single tree is lowest among all four models. The training time of Random Forest is significantly lower than Adaboosting and Bagging, although it has the largest number of trees. Compared with Adaboosting, Bagging has a shorter running time and higher accuracy (when the number of decision trees is similar). In terms of test accuracy, Bagging and Random Forest performs comparably and better than Adaboosting. Generally speaking, Bagging performs better in the steel plate fault classification data in consideration of both accuracy and efficiency.

TABLE 3
COMPARISON OF ACCURACY, RUNNING TIME, AND NUMBER OF DECISION TREES

	Train Accuracy	Test Accuracy	Time (s)	N tree
Decision Tree	93.57%	85.43%	0.014	1
Adaboosting	100.00%	88.57%	0.241	15
Bagging	96.30%	90.00%	0.166	18
Random Forest	100.00%	90.29%	0.138	31

B. Advantages and Disadvantages

Comparing the advantages and disadvantages of each model can help us choose the correct algorithm when facing a new problem. In the following article, we will describe the main advantages and disadvantages of the four models briefly.

1) Decision tree: First, the Decision Tree is simple and easy to understand. It is time-saving even with large data amount. In Figure 1, we can see that as the depth increases, the decision tree is also easy to overfit. This can be solved by pruning. The combination algorithm of decision tree, such as Bagging and Random Forest, can solve the overfitting problem.

2) Adaboosting: Adaboosting makes a good use of weak classifiers for cascading, which uses different classification algorithms as weak classifiers. Adaboosting can achieve high accuracy. Compared with the Bagging algorithm and Random Forest algorithm, Adaboosting considers the weight of each classifier fully. The number of Adaboosting weak classifiers is difficult to choose, but we can use cross validation to determine the optimal number of weak classifiers. Adaboosting is more time-consuming, as shown in Table 3.

3) Bagging: It's the same degree of complexity when training a Bagging integration and using the base learning algorithm to train a learner [22, 23]. This shows that Bagging is a very efficient ensemble learning algorithm. Both Bagging and Boosting are effective methods to improve the accuracy of the classification. In most data sets, the accuracy of Bagging is lower than Boosting. However, Bagging can save a lot of time by parallel training, and can avoid overfitting effectively. These two points can be seen from Table 3 and Figure 3.

4) Random forest: In order to solve the problem of limited number of instances, Random Forest is extracted twice randomly, once for training instances, and the other is the random extraction of variables. Being able to deal with continuous and discrete variables, Random Forest can prevent overfitting and increase stability. What's more, it is insensitive to noise and suitable for multi-classification problems, which can handle high-dimensional data (a lot of features) with high accuracy and high speed.

V. DISCUSSION

Although the accuracy of each model is different, it is very high for the traditional model. In Mahmoud Fakhr's and Alaa M. Elsayad's " Steel Plates Faults Diagnosis with Data Mining Models" paper, data mining model is used to achieve 98.09% accuracy. Although they have achieved high accuracy, we have omitted some samples which have not been well classified in the data description. It is still uncertain what kind of defects such samples are, which may cause some minor troubles in actual production. Some scholars even apply more methods to defect classification to analyze which methods have better efficiency and accuracy [24]. Despite there may be problems in practice, these machine learning methods must be more efficient than manual methods. More suitable algorithms can be studied in the future to improve accuracy and applicability. These ideas can also be extended to other industrial production areas.

VI. CONCLUSION AND RECOMMENDATIONS

inspection methods are mainly used to identify faults. In the era of machine learning, we can use computers instead of manpower to identify faults more fast and accurate, and improve the quality and efficiency of production. We use some classical algorithms in machine learning to deal with the data of steel plate faults which are public available at the UCI. The accuracy, time consumption, advantages and disadvantages of each model are compared in this article. In data analysis, we omit some samples that can not be classified. If we let machine learning samples like this, some defects may not be found in actual production. However, there is no design for a more appropriate algorithm. In the future, we can communicate with the steel factory technicians in details to design a more suitable algorithm for the classification of the faults of the steel plate, which can certainly improve the accuracy of classification. Since each model has its own advantages, we can combine the classic models with each other, give each model a weight, and integrate the advantages of each model. The combined models may achieve better classification performance.

REFERENCES

[1] H. J. Lv and Z. Y. Luo, ``Signal processing of surface fault detection for cold rolled steel plate,'' *Chinese Journal of Quantum Electronics*, vol. 1, no. 1, pp. 101-105, 1991.

[2] Q. T. Xu, ``Inspection and analysis of rail bottom faults,'' *Ansteel Technology*, vol. 3, no. 4, pp. 37-42., 1997.

[3] A. H. Al-Saeedi and O. Altun, ``Binary Mean-Variance Mapping Optimization Algorithm (BMVMO),'' *Journal of Applied and Physical Sciences*, vol. 2, no. 2, pp. 42-47, 2016. doi: https://doi.org/10.20474/japs-2.2.4

[4] T. A. Reda, N. M., M. A. Marzok, and S. M. Khamis, ``Range-suffrage algorithm for grid task scheduling,'' *International Journal of Applied and Physical Sciences*, vol. 1, no. 2, pp. 42-50, 2015. doi: https://doi.org/10.20469/ijaps.50004-2

[5] I. H. Witten, E. Frank, M. A. Hall, and C. J. Pal, *Data Mining: Practical Machine Learning Tools and Techniques*. Burlington, MA: Morgan Kaufmann, 2016.

[6] M. M. Hasan and O. Altun, ``Two enhanced differential evaluation algorithms for expensive optimization,'' *Journal of Applied and Physical Sciences*, vol. 2, no. 2, pp. 48-53, 2016. doi: https://doi.org/10.20474/japs-2.2.3

[7] J. R. Quinlan, ``Simplifying decision trees,'' *International Journal of Man-Machine Studies*, vol. 27, no. 3, pp. 221-234, 1987. doi: https://doi.org/10.1016/s0020-7373(87)80053-6

[8] H. S. Bahrudin, Alam, and T. Haiyunnisa, ``Computational fluid dynamic simulation of pipeline irrigation system based

on ansys,'' *International Journal of Technology and Engineering Studies*, vol. 2, no. 6, pp. 189-193, 2016. doi: https://doi.org/10.20469/ijtes.2.40005-6

[9] M. Kearns and L. Valiant, ``Cryptographic limitations on learning boolean formulae and finite automata,'' *Journal of the ACM*, vol. 41, no. 1, pp. 67-95, 1994. doi: https://doi.org/10.1145/73007.73049

[10] R. E. Schapire, ``The strength of weak learnability,'' *Machine Learning*, vol. 5, no. 2, pp. 197-227, 1990. doi: https://doi.org/10.1007/bf00116037

[11] L. Breiman, ``Bagging predictors,'' *Machine Learning*, vol. 24, no. 2, pp. 123-140, 1996. doi: https://doi.org/10.1007/bf00058655

[12] Y. Freund and R. E. Schapire, ``A decision-theoretic generalization of on-line learning and an application to boosting,'' *Journal of Computer and System Sciences*, vol. 55, no. 1, pp. 119-139, 1997. doi: https://doi.org/10.1007/3-540-59119-2_166

[13] L. Breiman *et al.*, ``Arcing classifier (with discussion and a rejoinder by the author),'' *The Annals of Statistics*, vol. 26, no. 3, pp. 801-849, 1998. doi: https://doi.org/10.1214/aos/1024691079

[14] T. K. Ho, ``Random decision forests,'' in *Proceedings of the Third International Conference on Document Analysis and Recognition,* Montreal, Canada. IEEE, 1995.

[15] M. Buscema, ``Metanet: The theory of independent judges,'' *Substance Use & Misuse*, vol. 33, no. 2, pp. 439-461, 1998. doi: https://doi.org/10.3109/10826089809115875

[16] M. Buscema, S. Terzi, and W. Tastle, ``A new meta-classifier,'' in *2010 Annual Meeting of the North American Fuzzy Information Processing Society,* Toronto, Canada, 2010.

[17] J. Friedman, R. A. Olshen, and C. Stone, *Classification and Regression Trees*. Monterey, CA: Wadsworth & Brooks/Cole Advanced Books & Software., 1984.

[18] R. Kohavi, ``A study of cross-validation and bootstrap for accuracy estimation and model selection,'' in *International Joint Conference on Artificial Intelligence, Montreal, Canada*, 1995.

[19] S. Geman, E. Bienenstock, and R. Doursat, ``Neural networks and the bias/variance dilemma,'' *Neural Computation*, vol. 4, no. 1, pp. 1-58, 1992. doi: https://doi.org/10.1162/neco.1992.4.1.1

[20] H. Li, *Statistical learning method*. Beijing, China: Tsinghua University Press, 2012.

[21] C. Strobl, A.-L. Boulesteix, A. Zeileis, and T. Hothorn, ``Bias in random forest variable importance measures: Illustrations, sources and a solution,'' *BMC Bioinformatics*, vol. 8, no. 1, pp. 25-67, 2007. doi: https://doi.org/10.1186/1471-2105-9-307

[22] T. G. Dietterich, ``An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization,'' *Machine Learning*, vol. 40, no. 2, pp. 139-157, 2000. doi: https://doi.org/10.1023/a:1007607513941

[23] M. W. Ahmad, J. Reynolds, and Y. Rezgui, ``Predictive modelling for solar thermal energy systems: A comparison of support vector regression, random forest, extra trees and regression trees,'' *Journal of cleaner production*, vol. 203, pp. 810--821, 2018. doi: https://doi.org/10.1016/j.promfg.2017.07.092

[24] D. Simić, V. Svirčević, and S. Simić, ``An approach of steel plates fault diagnosis in multiple classes decision making,'' in *International Conference on Hybrid Artificial Intelligence Systems,* Seville, Spain, 2014.