Int. J. Advance Soft Compu. Appl, Vol. 12, No. 1, March 2020
ISSN 2074-8523

Improving Learning Style Prediction using Tree-based Algorithm with Hyperparameter Optimization

Haziqah Shamsudin[1], Umi Kalsom Yusof[2*], and Maziani Sabudin[3]

[1,2,3] School of Computer Science, 11800 Universiti Sains Malaysia, Pulau Pinang, Malaysia.
e-mail: haziqah.ucom12@student.usm.my, umiyusof@usm.my

Abstract

Learning style of specific users in an online learning system is determined based on their interaction and behavior towards the system. The most common online learning theory used in determining the learning style is the Felder-Silvermans Theory. Many researchers have proposed machine learning algorithms to establish learning style by using log file attributes. However, they did not optimize the parameters selections which also contribute to low performance matrices. In this paper, tree-based algorithm is being used to detect the learning style of the user. The tree-based algorithms used in this paper are the Decision Tree (CART), Random Forest (RF), and Extreme Gradient Boosting (Xgb). In order to optimize the results of the performance matrices, the parameters of the tree-based algorithm classifiers are optimized by using the grid search hyper-parameter optimization. From the experiments, RF had proven to be the most effective algorithm, with the accuracy improving from 89% to 93%.

Keywords: *Hyperparameter Optimization, Learning Style, Online Learning, Tree-based Algorithm.*

1 Introduction

Learning style is known as way of learning or inclination by the student on how materials are introduced, how to work with it and how to internalize information [1]. Identifying a student's learning style can help students aware of their strength and weaknesses when it comes to learning. It is also meant to be used in determining the learning preferences of each student either in a traditional classroom or through an online learning based system [2].

An online learning based system can be defined as an online system where there is an interaction between students and system [3]. Initially, in an online learning based system, the learning style of the user is determined by using available learning style questionnaires based on the selected learning style model. The most commonly used learning style model is the FSLSM. It also incorporates different elements from different learning style models such as Kolb, Pask and Myers-Briggs [4]. However, when students are asked to fill in the questionnaire, the students take longer time to fill it as the questions are long. Furthermore, students tend to refuse spending too much time on the questionnaire. This causes them to put in random answers [5].

So with that, researchers came out with a new alternative where they determine the learning style automatically [4]. This is done by collecting log files of the interactive behavior of the user with the system. This consists of the number of mouse clicks, the time taken to do the task, number of views towards certain materials, and others. These attributes were then matched with the learning style model that they had chosen. From there, the results obtained are analyzed further and the learning style of the user is revealed.

In this paper, the used of tree-based algorithms is being highlighted as a classifier in predicting the learning style. The chosen tree-based algorithms in this paper are the CART, Random Forest (RF) and Extreme Gradient Boosting (Xgb). The use of CART acts as a base model of the tree-based algorithm followed by RF which highlights the used of bagging method and lastly, the used of Xgb highlights the used of boosting method. To further enhance the tree-based algorithm, hyperparameter optimization is being incorporated in the algorithm. Hyperparameter optimization is needed to find the best combination of parameters that will improve the performance of the tree-based algorithms.

2 Related Work

2.1 Learning Style

Determining an accurate learning style of the user will lead to better adaptivity of the online learning system which will then increase the user's performance. There are many learning style models available in this area as mentioned by H.M. Truong [4] in the last 30 years, where over 70 theories were developed [4]. One of the most commonly used models is the Felder-Silvermans model, which differentiates learning styles through 4 different dimensions which are Perception, Input, Processing and Understanding. This theory is by far the most widely used in adaptive learning systems (accounted for 70.6% of all papers in the survey conducted by H.M. Truong), [4].

FSLSM can describe the student's learning style in great detail. Perception dimension which consists of a sensing and intuitive learning style describes a preference for processing information. In this dimension, sensing learning styles learners prefer to learn concrete materials and facts. On the other hand, intuitive learners prefer underlying materials and theories which is known as abstract materials where it contains a general principles. The Input dimension consists of a visual and verbal learning style. Visual learners prefer graphical materials such as videos, graphs or charts, while verbal learners prefer spoken or written words [7].

The processing dimension consists of an active and reflective learning style. Active learners prefer to learn by collaboration and experimentation which is by doing while reflective learners prefer to think about the information and absorb it alone or in small groups. Lastly, for the understanding dimension it consists of a sequential and global learning style, where sequential learners prefer to learn in a step-by-step sequence and had the tendency to take small steps through the learning material while global learners tend to make larger steps of understanding by seeing the larger picture [8].

2.2 Tree-based Algorithm

Tree based algorithms like decision CART, RF and Xgb are used in all kinds of data science problems. From previous research, there are two papers in learning style prediction that use decision tree algorithms [8], [9]. Both of these papers manage to increase the percentage of accuracy in learning style prediction compared to previous papers. With that, in this paper we focus more on the use of tree-based algorithm in improving the learning style prediction.

2.3 Hyperparameter Optimization

Machine learning systems are abounding with hyperparameters. Choosing the best hyperparameter is essential in improving the machine learning model. After complete training of the model parameters, hyperparameters are chosen to optimize the validation loss. [10]. In machine learning, determining the optimal hyperparameter for a learning algorithm, and identifying good value of the parameter is called hyperparameter optimization. Hyperparameter optimization is the minimization of parameter over a subset of parameter. Different datasets, tasks, and learning algorithm families give rise to different sets of parameters and functions [11].

The critical step in hyperparameter optimization is to choose the set of parameters. The most commonly used technique in hyperparameter optimization is a grid search technique. Grid search requires choosing a set of values for each variable. It is easy to execute and parallelization is trivial. Other than that, it is also is reliable in low dimensional spaces [12].

3 Methodology

3.1 Data Selection

The data used in this paper taken from a research done by Renato [9]. The data is collected from the year 2012-2016. It contains a record of 507 students enrolled in, Computer Technology courses which have successfully completed the Computer Programming 1 subject. This dataset consists of 15 different attributes. Table 1 shows the different attributes involved which is then divided into the respective dimensions of learning style according to the FSLSM theory. These attributes are then matched to a learning style model specified by the researcher. In this case, FSLSM is used which consists of four different dimensions. The dimensions involved are Input, Processing, Perception and Understanding.

Table 1: Attributes match to Dimension of FSLSM

Attribute Name	Description of Attribute	Dimension
Forum Post Forum View Self-assessment	Post more often in dimension forum Reading post but rarely posting themselves Perform more self-assessment test	PROCESSING
Concrete Materials Examples Exercise rev	Prefers concrete learning materials (facts, data) Prefer examples Prefers to review answers in graded exercise tests	PERCEPTION
Visual Materials Video Materials	Prefers learning materials supplemented with pictures, diagrams, graphs Prefers learning material presented in text or audio	INPUT
Course overview Nav Euclidean distance Nav Euclidean distance	Prefers overviews, outlines Prefers to go through the course step by step (linear) Prefers to skipping the material (non-linear way)	UNDERSTANDING

3.2 Tree-based Algorithm

Tree-based algorithm that is involve in this paper is the CART, RF and Xgb. CART is the based model for the tree-based algorithms. Based on the attributes values, classification trees are used to classify an object or an instance into a predefined set of classes [14]. The splitting value of CART is determined by the value of Gini index. At the same time, the *maxdepth* will ensure no further splitting even if the leaves have the required min samples.

RF is a tree-based algorithms that used the bagging method concept [15]. It build a subset tree from the original training data. In RF, the final RF decision is decided from the calculation of majority votes. Two user-defined parameters involved in RF are the number of attributes used in splitting the nodes, *mtry* and the number of trees, *ntree*. In RF, the higher number of trees is better as it uses the square root of the number of variables for the value of *mtry* [15].

Lastly, the tree based algorithm involved is the Xgb. The Xgb classifier uses extreme gradient boosting [18], which has been shown to be effective in a wide variety of tasks. This algorithm first builds a tree from the training dataset. Then, moving forward it will update the weights and build a second tree by improving the first tree. The process is repeated until the stopping criteria is reached. In this algorithm the stopping criteria is determined when the algorithm manage to reduce the iteration error. This algorithm runs in a sequential manner. The whole processes that cover the first stage and the second stage is shown in Fig 1.

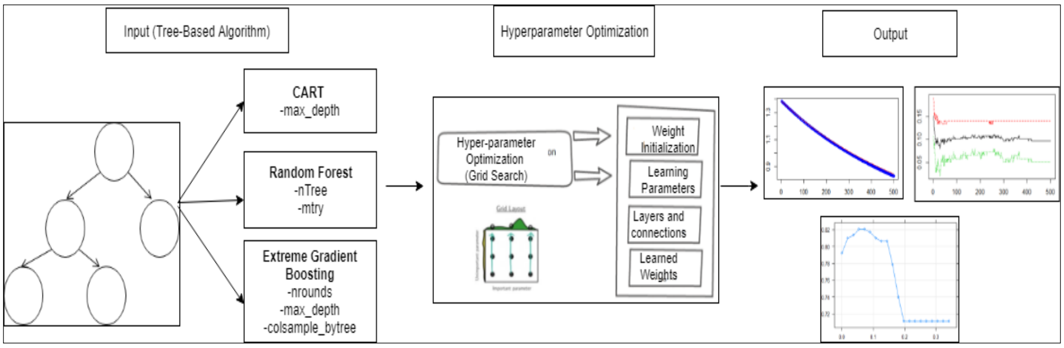


Fig. 1: Working Flow of the Tree based Algorithm in Learning Style Prediction with and without Hyperparameter Optimization

3.3 Incorporating Hyperparameter Optimization in the Tree-based Algorithms

3.3.1 Classifiction and Regression Tree (CART)

In this paper the type of DT used is CART. In decision tree, the most crucial part in the construction of the algorithm is the assignment of the considered node to its attribute. Impurity measure is responsible in deciding the choice of the attribute. In this paper, gini impurity measure is being used. This is because gini index favors larger partitions and it uses squared proportional of classes which is suitable for binary class of dataset.

For binary class, the node is split into two child nodes. Then, the best splitting attribute is determined. It is determined by using the gini index. Gini index is responsible in optimizing the control parameter involved in this paper which is the complexity value $_{\chi}(cp)$, where *cp* = *complexity parameter*. The complexity parameter is used to decide the optimal tree size and control the size of the decision tree. The tree building stops when the cost of adding another variable to the decision tree is above the *cp* value. The formula of gini index is shown in Equation 1 and Equation 2.

$$Gini = 1 - \sum_{i=1}^{C}(P_i)^2 \quad (1)$$

Where the algorithms works as follows:-

$$1 - (P(1 - (P(class1)^2) + P(class2)^2) + \cdots + P(classN)^2)) \quad (2)$$

The working flow of this algorithm is shown in Algorithm 1. It shows the working flow of applying this algorithm in predicting the learning style of the user. The algorithm is then applied to all the dimension of FSLSM which consists of Input, Processing, Perception and Understanding dimension. The parameter setup for this algorithm begins with *cp* value of 0.0001 and tune for 20 times.

Algorithm 1 Decision Tree (CART) Algorithm with Hyperparameter Optimization

```
procedure CART(cp)
    for each class, C_i ∈ D, do
        Specify the trControl with 5-fold cross-validation and grid search
        Find the prior probabilities D(C_i)
    end for
    Create the root node for each split predicate and label
    for each branch in split do
        Calculate the probability of class in the given branch
        square the class probabilities
        subtract the sum from 1
        weight each branch based on the baseline probability
        sum the weighted gini index for each split
    end for
    apply the optimized model with optimal cp value to the testing data C_i ∈ E
    The process is repeated for all four dimension respectively
    Print the result
    return
end procedure
```

3.3.2 Random Forest

RF brings extra randomness into the model when it is growing the trees. The parameters in RF are either used to increase the predictive power of the model or to make the model faster. The advantages of using the RF algorithm is that it can avoid over fitting problems, handle missing values and can also be modeled for categorical values.

For this paper, the parameter that was selected to undergo hyperparameter optimization to improve the percentage of accuracy is the *ntree* and *mtry*. The RF is developed by aggregating trees. It can be used for classification. The function of *mtry* is to randomly samples variables at each split. This algorithm works in three steps. First, the bootstrap samples are draw by *ntree*. For each bootstrap sample, grow un-pruned tree by choosing the best split based on the random sample of *mtry* predictors at each node. Lastly, predict new data using majority votes for classification. The pseudocode of the algorithm is shown in Algorithm 2.

In RF, the whole dataset is split into a subsample of the tree. For each set of the tree, the result of the predictor model and OOB error is calculated. At the same time, the parameter is adjusted by following the concept of grid search hyperparameter optimization. Next, the most optimized parameter is then compared within each subsample tree and the majority vote of the best model is chosen as the final predictor model of the random forest. The parameter value for random forest consists of nTree = (0...500) and a constant parameter value for stepFactor and improve of 0.5 and 0.05 respectively.

Algorithm 2 Random Forest Algorithm with Hyperparameter Optimization

procedure RF(*Ntree, mtry*)
    for each class, $C_i \in D$, do
        Specify the trControl with 5-fold cross-validation and grid search
    end for
    for RF functions **do**
        Draw *ntree* bootstrap sample,
        For each bootstrap sample, grow un-pruned tree by choosing best split based of random sample of *mtry* prediction at each node
        Predict new data using majority votes based on *ntree* trees
        Plot RF
    end for
    t: **optimize *mtry* to reduce OOB error**
    for dot use (D),
        specify the supporting parameter of model t and *ntree*
        Determine best *mtry* value, from the optimize and insert the value back to step 7
        Predict the result of new data
    end for
    Print the result
    return
end procedure

3.3.3 Extreme Gradient Boosting

In the Xgb algorithm several parameters are responsible in improving the prediction accuracy of the model. The parameter *nrounds* is used to determine the number of iteration of the sequential process of Xgb. Next, the max depth parameter is responsible in building the split of the tree in Xgb. In the Xgb algorithm the build of the next tree depends heavily on the previous tree build. This is because Xgb improves its performance by increasing the weight of the tree based on the previous tree. It runs in a sequential manner. Several other parameters that are involved in building the model of Xgb is *eta, gamma* and *colsample_bytree*. The pseudocode of the algorithm is shown in Algorithm 3.

In extreme gradient boosting, the algorithm works in a sequential manner compared to random forest which works in a parallel manner. In Xgb, the original dataset first builds a tree. Then, the tree undergoes a series of iterations and updates the weights of subsequent tree builds. This is obtained with the aid of parameter optimizing from the tree. The parameter value of extreme gradient boosting is specified as *nrounds* = (100, 200), followed by *max_depth* = (10, 15, 20, 25), *colsample_bytree* = (0.5, 0.9) and lastly, the fixed parameter is *eta* and *gamma* with a value of 0.1 and 0 respectively.

Algorithm 3 Extreme Gradient Boosting Algorithm with Hyperparameter Optimization

procedure Xgb(*error rate*, *nrounds*, *max_depth*)
  for each class, $C_i \in D$, do
    Specify the trControl with 5-fold cross-validation and grid search
  end for
  for each tree **do**
    Repeat from 1 to add up to the number of trees
    Based on the steps run previously, update the targets weights (higher for the ones mis-classified)
    For the selected subsample of data, fit the model
    On the full set observations, make
    Update the output with current results taking into account the learning rate
    Return the final output
  end for
  t: **optimize *nrounds, max_depth* to reduce error rate**
  for dot use (D),
    Predict the result from the new optimized model
  end for
  Print the result
  return
end procedure

3.4 Performance Measure

Confusion matrix is used to measure the performance of the proposed model. In classification problem, confusion matrix able to illustrates the accuracy of the solution. The confusion matrix contains information about actual and predicted classification done by a classification system. Using data in the matrix, the performance is evaluated. The predicted TP and TN classifications are calculated based on Equation 3 and Equation 4 as follows:

$$TP_{rate} = \frac{TP}{TP+FN} \tag{3}$$

$$TN_{rate} = \frac{TN}{TN+FP} \tag{4}$$

Where,

- *TP* (True Positive) = Number of records classified as true while they were actually true.
- *TN* (True Negative) = Number of records classified as false while they were actually false.
- *FP* (False Positive) = It denotes the number of records classified as true while they were actually false.
- *FN* (False Negative) = It denotes the number of records classified as false while they were actually true.
- In this paper the objectives function is needed to maximize the accuracy value of the learning style prediction and user performance. The formula to calculate the accuracy is given in Eq. 5. Accuracy is needed to determine how often the classifier is correct.

$$Accuracy = \frac{TN+FP}{TP+FN+FP+TP} \quad (5)$$

The other performance measure used in this paper is the sensitivity and specificity. Sensitivity, measures correctly identified proportion of actual positives which is also called the true positive rate. Specificity, measures correctly identified proportion of actual negatives, which is also known as the true negative rate. The equation for sensitivity and specificity is both shown is Equation 6 and Equation 7 respectively.

$$Sensitivity = \frac{TP}{TP+FN} * 100 \quad (6)$$

$$Specificity = \frac{TN}{TN+FP} * 100 \quad (7)$$

4. Result and Evaluation

This subsection presents and discusses the results of the tree based algorithms with and without hyperparameter optimization. Subsequently, the results are compared with previous work from literature based on its accuracy.

First, the analysis begins with the CART algorithm. The optimal cp value obtained after undergoing the hyperparameter optimization is shown in Figure 2a. The most optimal *cp* value selected is 0.099 with the percentage of accuracy being 0.858. The rule of selecting the best *cp* value is by selecting the value which has the highest percentage of accuracy. Next, the process is repeated to find the optimal cp value for processing dimension. In this dimension, the two class involves are active and reflective. The positive class obtained is the active class.

For this dimension, the prediction of learning style shows a rise of 0.01%. Even though the rise in percentage is small, the increasing in percentage of accuracy will lead to more accurate prediction of learning style, which then increases the adaptivity of the online learning system as mention in [19]. The most optimal cp value obtained for this dimension is 0.099 with the value of 0.858 and 0.716 for accuracy and kappa respectively. The result is shown in Figure 2b. Proceed further, the CART is used to evaluate the perception dimension. In this dimension the positive class obtained after undergoing hyperparameter optimization is the intuitive class. The three attributes involved in this dimension are concrete materials, examples and exercise review. For this dimension, the prediction of learning style also increased by 0.01%. The most optimal *cp* value obtain for this dimension is 0.144 with the value of 0.794 and 0.355 for accuracy and kappa respectively. It is shown in Figure 2c.

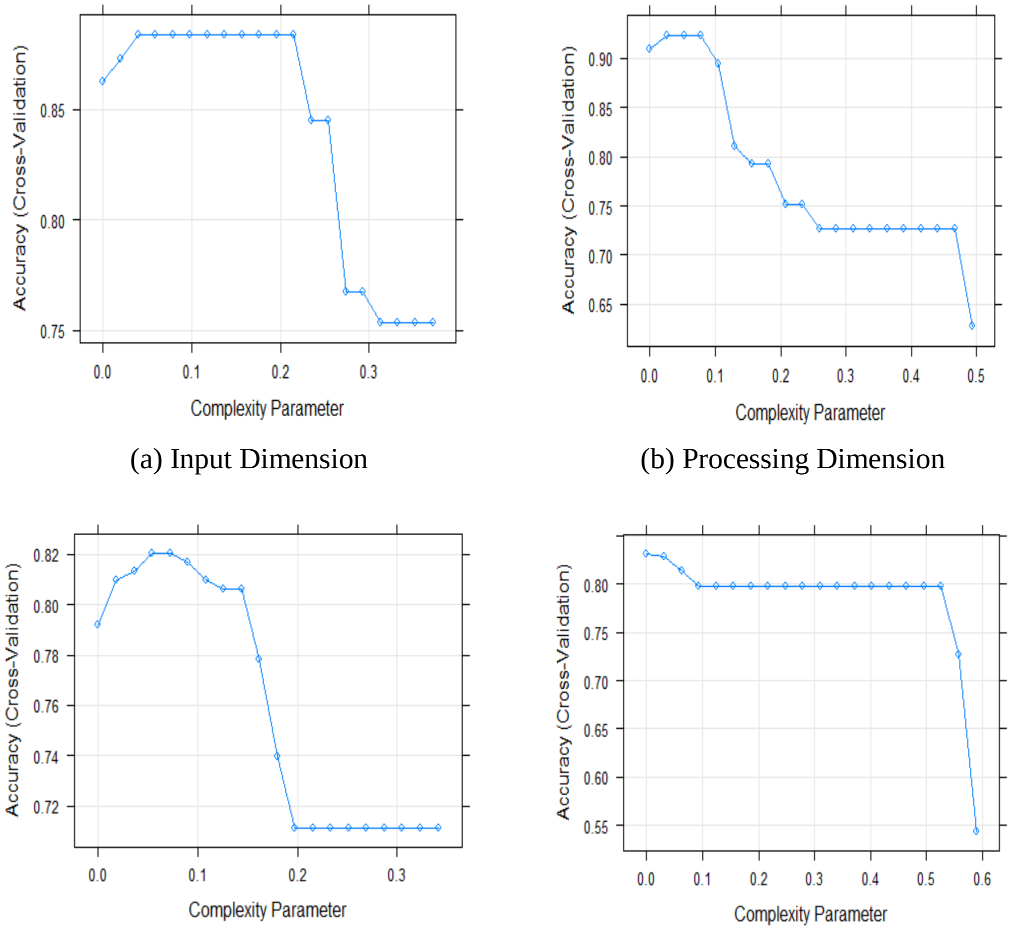
Lastly, the CART is used to evaluate the understanding dimension. In this dimension the positive class obtained after undergoing hyperparameter optimization is the global class. The two attributes involved in this dimension are course overview and nav euclidean distance. As shown in Figure 2d, the prediction of learning style also increased by 0.02%. The most optimal cp value obtained for this dimension is 0.594 with the value of 0.815 and 0.635 for accuracy and kappa respectively.

Next, the RF algorithm is applied to the same dataset. For Input dimension, the procedure begins by determining the most optimized *ntree* for the dimension. From Figure 3a, 3b, 3c and 3d, it can be observed that the optimization parameter for the dimensions remain constant from the value (300...500). This is because, according to [20], the concept of random forest, the higher the *ntree* value, the more subsample trees will be created which will allow a better prediction model. In this paper, from the optimization of the *ntree* it turns out that all the dimension have a constant value at 500.

The optimized value of the ntree is input back to the random forest model and the best value of mtry with OOB error remains consistent at either value 0 and 1. The mtry value is calculated by the formula of *sqrtp= mtry*. For the input dimension the OOB error obtained is 12.15%, followed by 4.07% for the processing dimension. Next, the OOB error for the perception dimension is 12.36% and lastly, the OOB error obtained for the understanding dimension is 19.23%. The value of the OOB error varies between all the dimensions. This is because, several factors are taken into consideration such as the number of instances in the class and the number of attributes involved.

Lastly, the hyperparameter optimization for extreme gradient boosting algorithm shows a constant curve line for the training and testing of the model in terms of the nlog value. It is shown in Figure 4a, 4c, 4b and 4d.

To compare the result of accuracy before and after incorporating hyperparameter optimization, the result is tabulated in Table 2. From the table the comparison values included the sensitivity, specificity, accuracy and Kappa value. In terms of percentage of accuracy, it can be observed from the Table that for all the dimensions the percentage of accuracy shows a positive improvement except for Xgb in the input dimension and DT in the perception dimension.



(a) Input Dimension

(b) Processing Dimension

(c) Perception Dimension

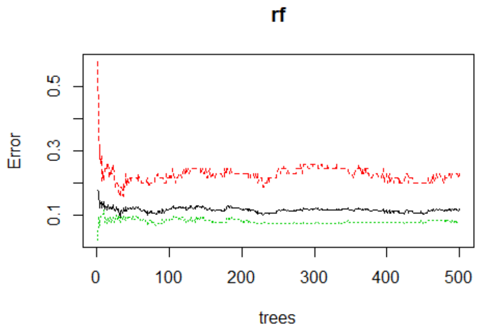
(d) Understanding Dimension

Fig. 2: Optimized *cp* for Learning Style FSLSM Dimension of Decision Tree (CART) with Hyperparameter Optimization

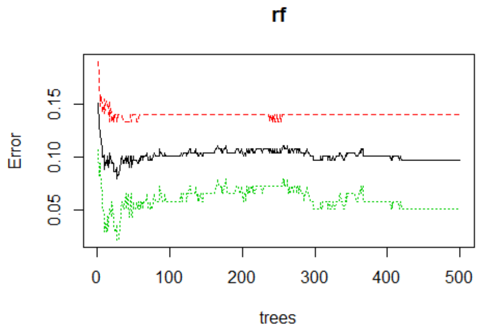
The percentage of accuracy for both dimension decrease by 0.03% and 0.07% respectively. Even though the percentage of accuracy decreased, the other evaluation matrices show positive increments.

From the results obtained, even slightest improvements for the accuracy value will provide a significant difference for the students. Being able to provide accurate learning style will help in improving the recommendations and advice to the students. On the other hand, it may also reduce the possibilities of providing a misleading advice and giving a mismatch items in adaptive learning system.

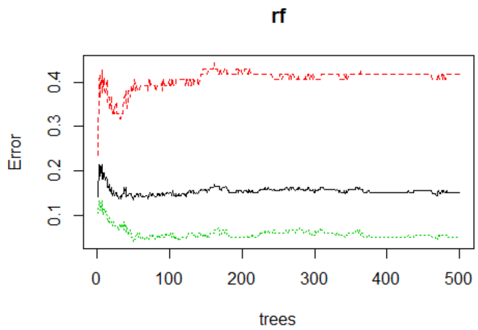
When comparing the results from Table 2, the percentage of accuracy shows a positive increment after incorporating hyperparameter optimization. This positive increment shows that choosing the right parameter and the value within is also important in increasing the prediction accuracy of the learning style, which will then lead to a better adaptivity of the system. Overall, among the tree based algorithms, random forest shows the highest percentage of accuracy in predicting the learning style of a user within the entire dimension.



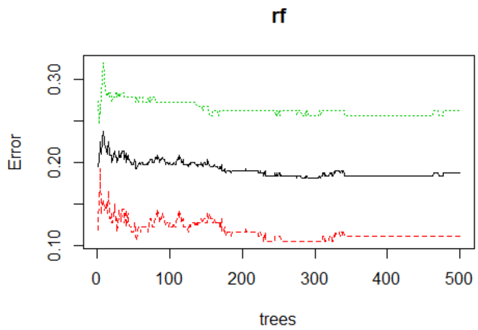
(a) Input Dimension



(b) Processing Dimension



(c) Perception Dimension

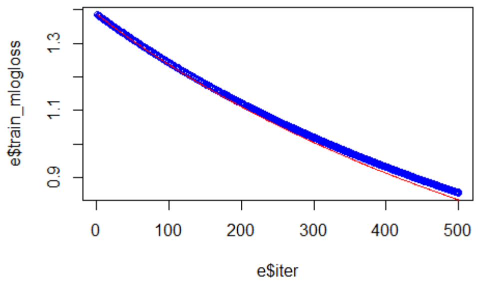


(d) Understanding Dimension

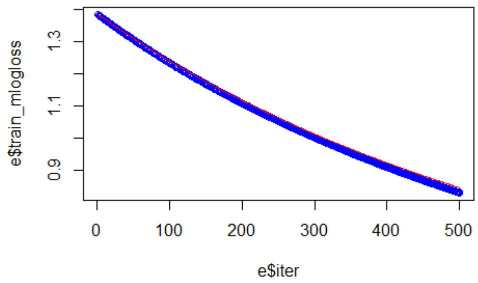
Fig. 3: Optimized *ntree* for Learning Style FSLSM Dimension of Random Forest with hyperparameter Optimization

Other than that, in this paper, the measurement analysis on the value of sensitivity and the specificity is also included. This is because both analysis measurements are needed when measuring how good the predictive model is. Sensitivity will determine the positive prediction while specificity will measure how much the model is able to detect false positive value. Being able to detect both of these values will lead to a better prediction of learning style, which then help in improving the adaptiveness of the system.

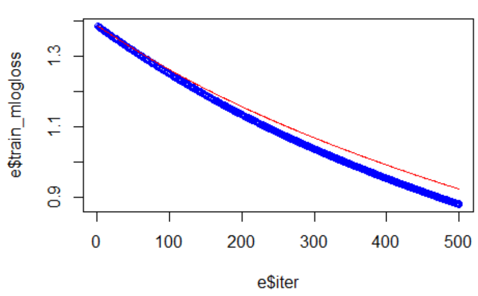
The value of accuracy for each of the dimension is calculated and compared with literature. From Table 3, the percentage of accuracy when including the hyperparameter optimization shows a consistent improvement. It shows that hyperparameter optimization able to improve the accuracy of predicting the learning style. The parameter for all the classifiers used in this paper is optimized accordingly using a grid search hyperparameter optimization.



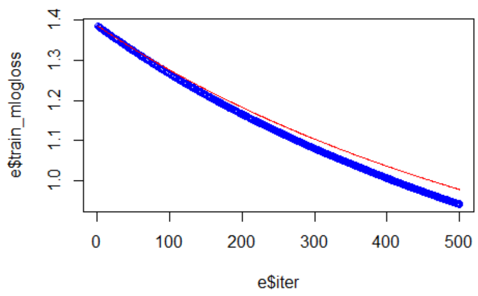
(a) Input Dimension



(b) Processing Dimension



(c) Perception Dimension



(d) Understanding Dimension

Fig. 4: Optimized *errorrate* for Learning Style FSLSM Dimension of Extreme Gradient Boosting with Hyperparameter Optimization

This is because the main objective of this paper is to see the increment value of accuracy when incorporating the hyperparameter optimization. The comparison of the classification accuracy is tabulated in Table 3. The result is compared with previous work from literature which use the same learning theory FSLSM, but without any hyperparameter optimization.

Even though, the increase of accuracy as can be observed from a previous research by [9], is small by only 0.04 compared to Random Forest. However, this small improvement in the accuracy can give a meaningful contribution to the students in online learning system. It means a more accurate information, interventions and correct identification of a students learning style [19]. This will also lead to a better adaptivity of the learning systems.

Table 2: Overall result of comparison for tree-based algorithms with and without hyperparameter optimization

Method	Without hyperparameter			With Hyperparameter			Dimension
	RF	CART	Xgb	RF	CART	Xgb	
Accuracy	0.88	0.89	0.90	0.94	0.93	0.87	INPUT
Sensitivity	0.75	0.55	0.85	0.89	0.87	1.00	
Specificity	0.92	1.00	0.92	0.96	0.95	0.83	
Accuracy	0.88	0.88	0.83	0.95	0.90	0.92	PROCESSING
Sensitivity	0.85	0.75	0.77	0.91	0.79	0.84	
Specificity	0.90	1.00	0.88	1.00	1.00	1.00	
Accuracy	0.86	0.88	0.80	0.95	0.81	0.83	PERCEPTION
Sensitivity	0.74	0.57	0.74	0.84	0.34	0.48	
Specificity	0.91	1.00	0.82	0.99	1.00	1.00	
Accuracy	0.74	0.80	0.79	0.86	0.82	0.79	UNDERSTANDING
Sensitivity	0.74	0.80	0.80	0.93	0.84	0.95	
Specificity	0.74	0.80	0.78	0.79	0.79	0.63	

Table 3: Comparison of average accuracy results with literature

Method	Average Accuracy
Random Forest	0.93 (1)
CART	0.86 (3)
Xgb	0.86 (3)
LSID-ANN [19]	0.81 (4)
J48 [9]	0.89 (2)
DeLeS [21]	0.79 (5)

5. Conclusion

This paper used three different tree based algorithms methods which are CART, RF, and Xgb in predicting the learning style. Then, this tree based algorithm is incorporated with the hyperparameter optimization to further improve the learning style prediction. From there, it is found that when incorporating hyperparameter optimization, the result in predicting the learning style increased with the leading algorithm being RF.

The results of the method were compared with existing approaches using the accuracy value, which is commonly used in the domain area of learning style detection [19],[9], [21]. Based on the accuracy value, the best solution on the averaging value of the FSLSM dimension always come from the tree based algorithm with hyperparameter optimization. By identifying students learning styles with a higher accuracy value, more accurate personalization for adaptive learning systems can be provided.

In conclusion, choosing the right value of parameter will affect the performance of classification model. The possible future work is outlined as follows:-

1. To use a hybrid method and observe the increase of classification accuracy.
2. To test the model in an online learning system and prove the effectiveness in a real case study.

ACKNOWLEDGEMENTS

The authors wish to thank Universiti Sains Malaysia (USM) for the support it has extended in the completion of the present research through the Research University Grant (RUI) (1001/PKOMP/8014084). The authors also wish to thanks Asst. Prof. Dr. Renato Racelis Maaliw III from Southern Luzon State University, Philippines for dataset sharing.

References

[1]. R. M. Felder, L. K. Silverman (1988). Learning and Teaching Styles in Engineering Education. *Engineering education*, 78 (7), 674–681.

[2]. F. Romanelli, E. Bird, M. Ryan (2009). Learning styles: A Review of Theory, Application, and Best Practices. *American Journal of Pharmaceutical Education*, 73 (1), 1-9.

[3]. Su, B., Bonk, C. J., Magjuka, R. J., Liu, X., & Lee, S. H. (2005). The importance of interaction in web-based education: A program-level case study of online MBA courses. Journal of Interactive Online Learning, 4(1), 1-19.

[4]. Truong, H. M. (2016). Integrating learning styles and adaptive e-learning system: Current developments, problems and opportunities. *Computers in human behavior,* 55 (1), 1185-1193.

[5]. Mokhtar, R., Zin, N. A. M., & Abdullah, S. N. H. S. (2010). Rule-based knowledge representation for modality learning style in AIWBES. In *Knowledge Management International Conference.*

[6]. Kim, Yong Seong, W. Nick Street, and Filippo Menczer. (2003) "Feature selection in data mining." In *Data mining: opportunities and challenges.* (pp. 80-105). IGI Global.

[7]. S. Graf, S. R. Viola, T. Leo, Kinshuk. (2007). In-depth analysis of the Felder-Silverman Learning Style Dimensions. *Journal of Research on Technology in Education*, 40 (1) , 79–93.

[8]. E. ¨Ozpolat, G. B. Akar. (2009). Automatic Detection of Learning Styles for an E-learning System, *Computers & Education*, 53(2), 355–367.

[9]. R. R. Maaliw III. (2015). Classification of Learning Styles in Virtual Learning Environment Using Data Mining: A basis for Adaptive Course Design. *IRJET Journal.*

[10]. D. Maclaurin, D. Duvenaud, R. Adams. (2015, June). Gradient-based Hyperparameter Optimization through Reversible Learning, In: *International Conference on Machine Learning* on (pp. 2113–2122).

[11]. J. Bergstra, D. Yamins, D. D. Cox. (2013) Making a Science of Model Search : Hyperparameter Optimization In *Hundreds of Dimensions for Vision Architecture*s. JMLR.

[12]. J. Bergstra, Y. Bengio. (2012, Feb). Random Search for Hyperparameter Optimization. *Journal of Machine Learning Research* (pp. 281–305).

[13]. S. Kale, R. Kumar, S. Vassilvitskii. (2011). Cross-validation and Mean-Square Stability, In *Proceedings of the Second Symposium on Innovations in Computer Science (ICS2011*). Citeseer.

[14]. D. Pregibon, L. A. Clark. (2017). Tree-based models, in: Statistical models in S, Routledge, (pp. 377–419).

[15]. L. Breiman. (2001). Random forests, *Machine learning*, 45 (1), 5–32.

[16]. Liaw, A., & Wiener, M. (2002). Classification and Regression by Random Forest. *R news*, 2(3), 18-22.

[17]. J. H. Friedman. (2002). Stochastic Gradient Boosting, *Computational Statistics & Data Analysis,* 38 (4) 367–378.

[18]. T. Chen, C. Guestrin. (2016). Xgboost: A Scalable Tree Boosting System, In *Proceedings of the 22nd ACM International Conference on Knowledge Discovery and Data Mining* (pp. 785-794). ACM.