International Journal of Assessment Tools in Education
2019, Vol. 6, No. 5-Special Issue, 20–36
https://dx.doi.org/10.21449/ijate.627361
Published at http://www.ijate.net http://dergipark.gov.tr *Research Article*

Educational data mining: A tutorial for the *rattle* package in R

Okan Bulut [1,*], **Hatice Cigdem Yavuz** [2]

[1] Centre for Research in Applied Measurement and Evaluation, University of Alberta, Edmonton, AB, Canada
[2] Cukurova University, Faculty of Education, Sarıçam/Adana, Turkey

ARTICLE HISTORY
Received: 01 October 2018
Accepted: 05 December 2019

KEYWORDS
Educational data mining,
rattle,
Decision tree,
Random forest,
Support vector machines

Abstract: Educational data mining (EDM) has been a rapidly growing research field over the last decade and enabled researchers to discover patterns and trends in education with more sophisticated methods. EDM offers promising solutions to complex educational problems. Given the rapid increase in the availability of big data in education and software programs to analyze big data, the demand for user-friendly, free software programs to implement EDM methods also continues to increase. The R programming language has become a popular environment for data mining due to its availability and flexibility. The *rattle* package in R contains a set of functions to implement data mining with a graphical user interface. This study demonstrates three widely used data mining algorithms (classification and regression tree, random forest, and support vector machine) in EDM using real data from the 2015 administration of the Programme for International Student Assessment (PISA). First, a brief introduction to EDM is provided along with the description of the selected data mining algorithms. Then, how to perform data mining analysis using the *rattle*'s graphical user interface is demonstrated. The study concludes by comparing the results of the selected data mining algorithms and highlighting how those algorithms can be utilized in the context of educational research.

1. INTRODUCTION

As an interdisciplinary field, educational data mining (EDM) refers to the development and use of advanced statistical methods to explore and identify patterns and relationships in data derived from educational settings. EDM aims to implement advanced machine learning and data mining algorithms (1) to exploit unprocessed data from educational settings (e.g., large-scale assessments, records of students' academic progress in school, and log data from e-learning systems), (2) to discover relations, patterns, and trends in education, and (3) to use the discovered information in order guide and improve the decision-making process in educational practices. The increasing availability and popularity of big data in education has created new pathways for educational researchers who are interested in applying EDM methods to find solutions for various problems in education – such as enhancing the quality of online learning environments (Ducange, Pecori, Sarti, & Vecchio, 2016), early prediction of student dropouts

CONTACT: Okan Bulut ✉ bulut@ualberta.ca Centre for Research in Applied Measurement and Evaluation, University of Alberta, Edmonton/Alberta, Canada

ISSN-e: 2148-7456 /

(Aulck, Velagapudi, Blumenstock, & West, 2016), and forecasting students' academic performance and identifying students who might be at risk of academic failure (Hussain, Zhu, Zhang, Abidi, & Ali, 2019).

Educational researchers who intend to use the EDM methods typically follow a deductive reasoning approach in which they first collect or get access to large volumes of data, explore the data visually and statistically, and then do further investigations in order to discover hidden patterns and relationships in the data. Unlike theory-driven educational research that usually aims to obtain evidence supporting a priori hypothesis, the primary goal of a typical EDM process is to find and extract new knowledge from the data without a particular priori hypothesis and to use the discovered information for the purpose of building new theory, if possible. In the context of EDM, educational researchers' interests are mainly focused on several dimensions, such as learning, predictive, behavioral, and visual analytics (Aldowah, Al-Samarraie, & Fauzy, 2019). Recent systematic review studies have also highlighted a vast and growing body of research on EDM and its applications in various areas of education (e.g., Aldowah et al., 2019; Baker, Martin, & Rossi, 2017; Dutt, Ismail, & Herawan, 2016; Peña-Ayala, 2014). The findings of these review studies reveal that educational researchers will continue to harness the power of EDM for solving complex problems in education with the availability of big data in education.

From the methodological point of view, EDM methods are the same as the data mining methods utilized in other scientific fields (e.g., business, finance, medicine, and agriculture). The current data mining methods can be categorized into two main types according to the availability of a target (i.e., dependent) variable in the data: supervised (also known as predictive) and unsupervised (also known as descriptive). Supervised data mining methods are appropriate when the researcher wants to predict a specific target variable that is already available in the data. Typical examples of supervised data mining applications include regression and classification tasks where the researcher wants to predict either a categorical (classification) or continuous (regression) variable using a set of predictors (i.e., features) available in the data. Unsupervised data mining methods are appropriate when the goal is to find hidden structures or relations in the data instead of predicting a target variable. Common examples of unsupervised data mining applications include clustering, association rule mining, and dimensionality reduction. A detailed review of data mining methods commonly used in educational research can be found in Aldowah et al. (2019) and Peña-Ayala (2014).

In education, researchers and practitioners are often interested in research problems in which the primary goal is the prediction of an outcome (i.e., dependent) variable from a set of predictors (Berland, Baker, & Blikstein, 2014; Sinharay, 2016). Therefore, EDM applications mostly involve supervised data mining methods, instead of unsupervised data mining methods. Previous research indicated that the supervised data mining methods often provide higher prediction accuracy than traditional methods, such as multiple linear and logistic regression (e.g., Fernández-Delgado, Cernadas, Barro, & Amorim, 2014; Koon & Petscher, 2015, 2016; Spikol, Ruffaldi, Dabisias, & Cukurova, 2018). This study focuses on three data mining algorithms that can be used for both classification and regression problems: classification and regression trees (CART; Breiman, Friedman, Olshen, & Stone, 1984), random forest (RF; Breiman, 2001), and support vector machines (SVM; Cortes & Vapnik, 1995). These algorithms have been widely used in previous EDM research due to their relatively lower complexity and ease of implementation and interpretation (e.g., Guruler, Istanbullu, & Karahasan, 2010; Ivancevic, Celikovic, & Lukovic, 2011; Mccuaig & Baldwin, 2012; Pardos, Wang, & Trivedi, 2012).

The CART algorithm relies on stratifying a large dataset into a number of smaller subsets in which separate regression models can be built for either continuous or categorical outcome

variables. Then, the model provides a set of classification or regression rules in a decision tree based on the nodes generated from the utilized predictors (Agarwal, Pandey, & Tiwari, 2012). As a nonparametric approach, CART does not make explicit assumptions about the distributions of variables, and thus it can produce relatively more accurate predictions (e.g., Strobl, 2013). The RF algorithm is similar to the CART algorithm in terms of relying on a regression or classification tree model for prediction. However, unlike CART, the RF algorithm generates many decision trees and combines all of them for making a final prediction (Breiman, 2001). Therefore, the RF algorithm can overcome many estimation issues (e.g., instability, high bias, and under-representation of classifications) in the CART algorithm because predictions are made based on the combination of many tree models that are generated differently using bootstrap samples, instead of a single decision tree model based on the entire sample (Sinharay, 2016; Williams, 2011). Differently from the previous two algorithms, the SVM algorithm relies on creating a separating hyperplane in an *N*-dimensional prediction space where *N* refers to the number of available predictors in the data. A hyperplane can be considered as a decision boundary that helps separate or classify the data points. If the outcome variable is categorical, then the hyperplane aims to create classes having the maximum distance between each other (Williams, 2011). If, however, the outcome variable is continuous, then the hyperplane creates a regression line (or plane) that can minimize the difference between the predicted and original values of the outcome variable.

Currently, there are many software programs that are capable of implementing the data mining algorithms mentioned above – such as RapidMiner, Weka, KEEL, KNIME, Orange, Python, R, and IBM SPSS Modeler (see Slater, Joksimović, Kovanovic, Baker, and Gasevic [2017] for a detailed review). Some of these programs (e.g., RapidMiner, Weka, and IBM SPSS Modeler) provide a graphical user interface (GUI) for users to easily select an algorithm along with the type of data mining analysis that they want to perform. Compared to these software programs, advanced programming languages such as Python and R (R Core Team, 2019) can provide users with more sophisticated tools to explore, organize, visualize, and model the data within the same computing environment. However, the amount of time that it takes to learn a new programming language and to achieve expertise in it can be very long for novice users who do not have any previous experience in programming. An exception in this situation is the *rattle* package (Williams, 2011) that provides a user-friendly GUI to perform data mining analysis within the R statistical computing environment (R Core Team, 2019). The *rattle* package can perform data mining analysis using a variety of advanced algorithms. The purpose of this study is to demonstrate how to use the *rattle* package for performing data mining analysis. Using a real dataset from a large-scale international assessment, the implementation of the CART, RF, and SVM algorithms using the *rattle* package is demonstrated. The steps for building and evaluating a predictive model in the *rattle* are also described in detail.

2. METHOD

2.1. Study Group

The sample of this study comes from the 2015 administration of the Organisation for Economic Co-operation and Development's (OECD) Programme for International Student Assessment (PISA). PISA is a large-scale, international assessment program that assesses the extent to which 15-year-old students have acquired adequate competency in various subject areas such as reading, mathematics, and science (OECD, 2018). The 2015 administration of PISA involved approximately 540,000 15-year-old students from 72 participating countries and economies. The sample of this study consists of 5896 students (49.83 % female) who participated in PISA 2015 from Turkey. This study uses the PISA dataset for the demonstration of the *rattle* package because the dataset is publicly available through the OECD website

(http://www.oecd.org/pisa/) and it consists of many categorical and continuous variables from students and schools – which creates a large-size database suitable for an EDM research study.

2.2. Measures

2.2.1. *Scientific Literacy Test in PISA 2015*

The primary focus of PISA 2015 was to assess students' scientific literacy as well as their attitudes and preferences regarding learning experiences in science. The results of PISA 2015 suggest that there is a large variation in students' competency levels in science and that this variation can be explained by many factors, such as demographic variables, socioeconomic status, students' participation in science-related activities, and the opportunity to learn science at school (Mostafa, Echazarra & Guillou, 2018; OECD, 2018). In this study, students' performance levels in scientific literacy were obtained from the PISA 2015 Scientific Literacy Test. The scientific literacy test was designed to assess three major competencies: explaining phenomena scientifically, evaluating and designing scientific inquiry, and interpreting data and evidence scientifically (OECD, 2017). Moreover, 36% of the items in the test were in physical, 36% in living, 28% in earth and space context. Students' scores obtained from the test were scaled with a mean of 500 and a standard deviation of 100. The average scientific literacy score in PISA 2015 was 493 across all participating countries. Using this score as a cutoff value, a categorical variable (science_perf) was created. For students whose scores were equal or higher than 493, science_perf was labeled as "High". If, however, students' scores were less than 493, then the label of "Low" was assigned to science_perf. The resulting categorical variable was used as the outcome variable in the data mining analysis.

2.2.2. *The student questionnaire*

The other variables regarding students (i.e., predictors) were obtained from the student questionnaire of PISA 2015. Table 1 shows the complete list of the variables used in this study.

Table 1. The list of the variables used in this study

Variable	Data type	Description
gender	Categorical	Female=1, Male=0
computer	Categorical	Owning a computer at home; Yes=1, No=0
software	Categorical	Owning software at home; Yes=1, No=0
internet	Categorical	Owning internet at home; Yes=1, No=0
desk	Categorical	Owning a desk at home; Yes=1, No=0
own.room	Categorical	Owning a room at home; Yes=1, No=0
quiet.study	Categorical	Owning a quiet study area at home; Yes=1, No=0
ANXTEST	Numeric	Test anxiety
COOPERATE	Numeric	Enjoying cooperation
EMOSUPS	Numeric	Parents emotional support
PARED	Numeric	Highest education of parents in years
TMINS	Numeric	Learning time in total
ESCS	Numeric	Index of economic, social and cultural status
TEACHSUP	Numeric	Teacher support in a science class
TDTEACH	Numeric	Teacher-directed science instruction
IBTEACH	Numeric	Inquiry-based science teaching and learning practices
SCIEEFF	Numeric	Science self-efficacy
science_perf	Categorical	If science scores >= 493, *High*; *Low* otherwise

2.3. Procedure

To use the *rattle* package, readers first need to download and install the R software program into their computers. Readers who have no experience regarding downloading, installing, and using R are recommended to check the program manual on the CRAN website (https://cran.r-project.org/doc/manuals/r-release/R-intro.pdf) prepared by Venables, Smith, and the R Core Team (2019). The *rattle* package contains a set of functions to implement data mining with a GUI. The latest installation instructions can be found at http://rattle.togaware.com. In this study, Rattle version 5.2.0 was used for data mining analysis. To download and install the *rattle* (Williams, 2011) and its required extension *RGtk2* (Lawrence & Lang, 2010), the following codes must be executed in the R console (note that this step requires Internet connection):

```
#Installing the packages
install.packages("rattle")
install.packages("RGtk2")
```

Once the packages have been installed successfully, both packages must be activated using the **library** command in R:

```
#Activating the packages
library("rattle")
library("RGtk2")
```

The next step is to the **rattle** command, which will open the *rattle* GUI as demonstrated in Figure 1.

```
#Opening the rattle GUI
rattle()
```

The *rattle* GUI can read several data formats, such as text files with .txt, .dat, or .csv extensions, RData files, and Open Database Connectivity (ODBC) files. This study uses "pisa_turkey.csv", which consists of the variables listed in Table 1. To open the pisa_turkey.csv in the *rattle*, the first step is to click "Filename" under the "Data" tab and look for the data file in the computer. Once the data file is found, the "Open" and "Execute" buttons should be clicked, respectively. This process will open the pisa_turkey.csv file in the *rattle* and load the dataset into the program (see Figure 2). For other types of data formats, the same procedure can be followed by selecting a specific file format available under "Source". Once a dataset is properly read and loaded into the *rattle*, a summary screen of the dataset becomes available (see Figure 3). The summary menu shows all the variables in the dataset, types of variables (numeric or categorical), and the role of the variables (e.g., input, target, and identity). Furthermore, the "Comment" column in the summary screen can help users identify potential issues in the variables (e.g., extreme missingness). Using the summary menu, users can change the default preferences regarding the variables. For example, the outcome variable must be labeled as "Target" so that this variable can be used as the outcome variable in the modeling stage. If the user wants to exclude some variables from the dataset, these variables should be labeled as "Ignore". Note that changes made on the summary screen will be saved only after the user clicks the "Execute" button. Otherwise, changes made on the variables will be lost.

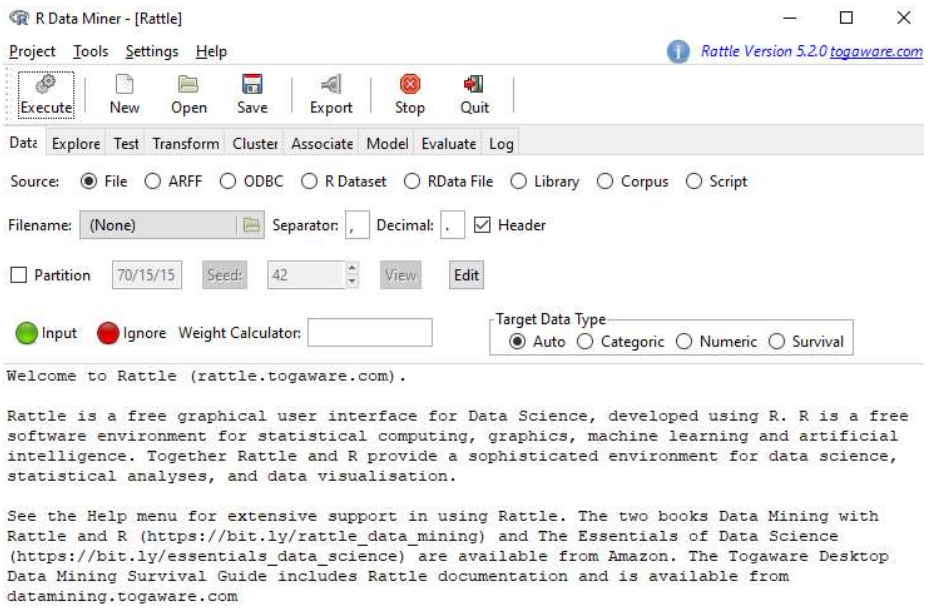

Figure 1. The graphical user interface (GUI) of the *rattle*

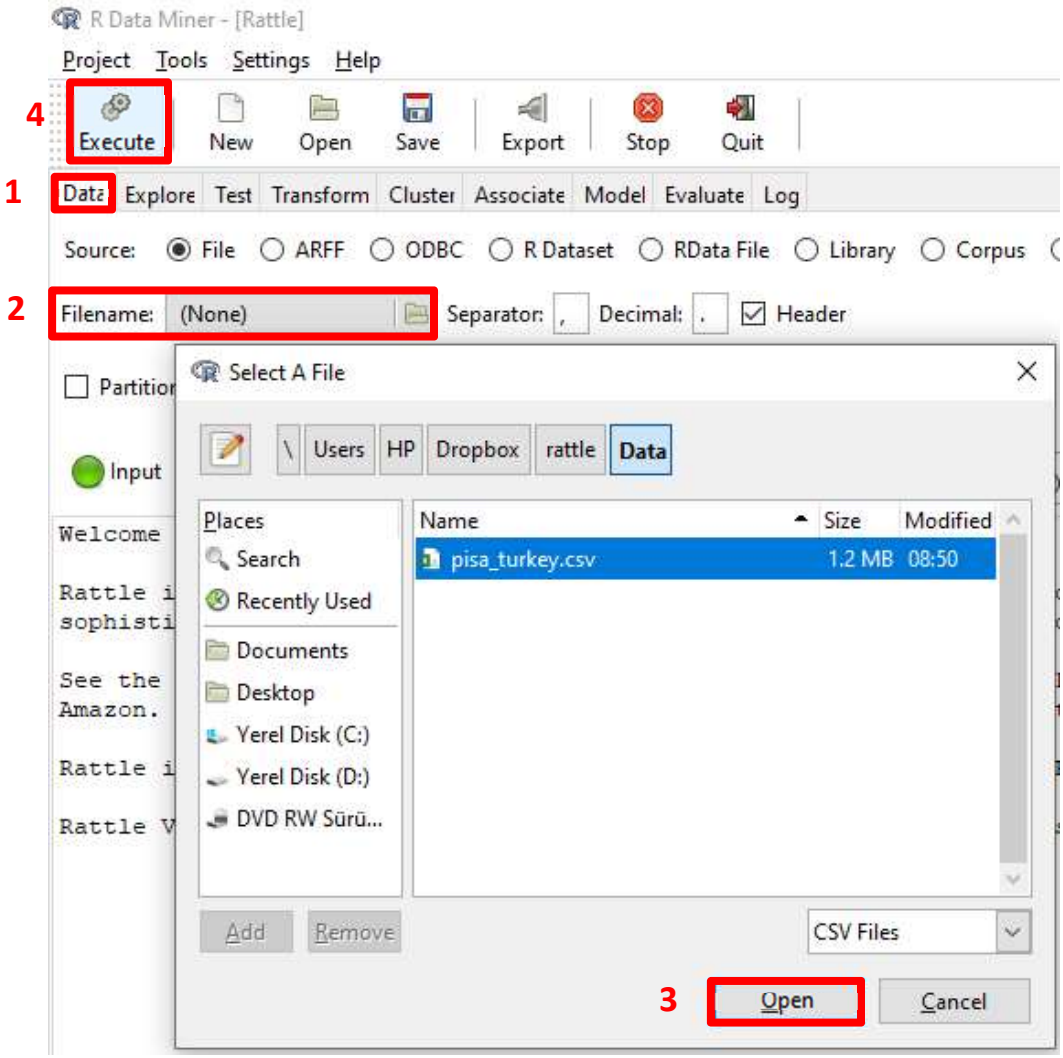

Figure 2. Loading the data

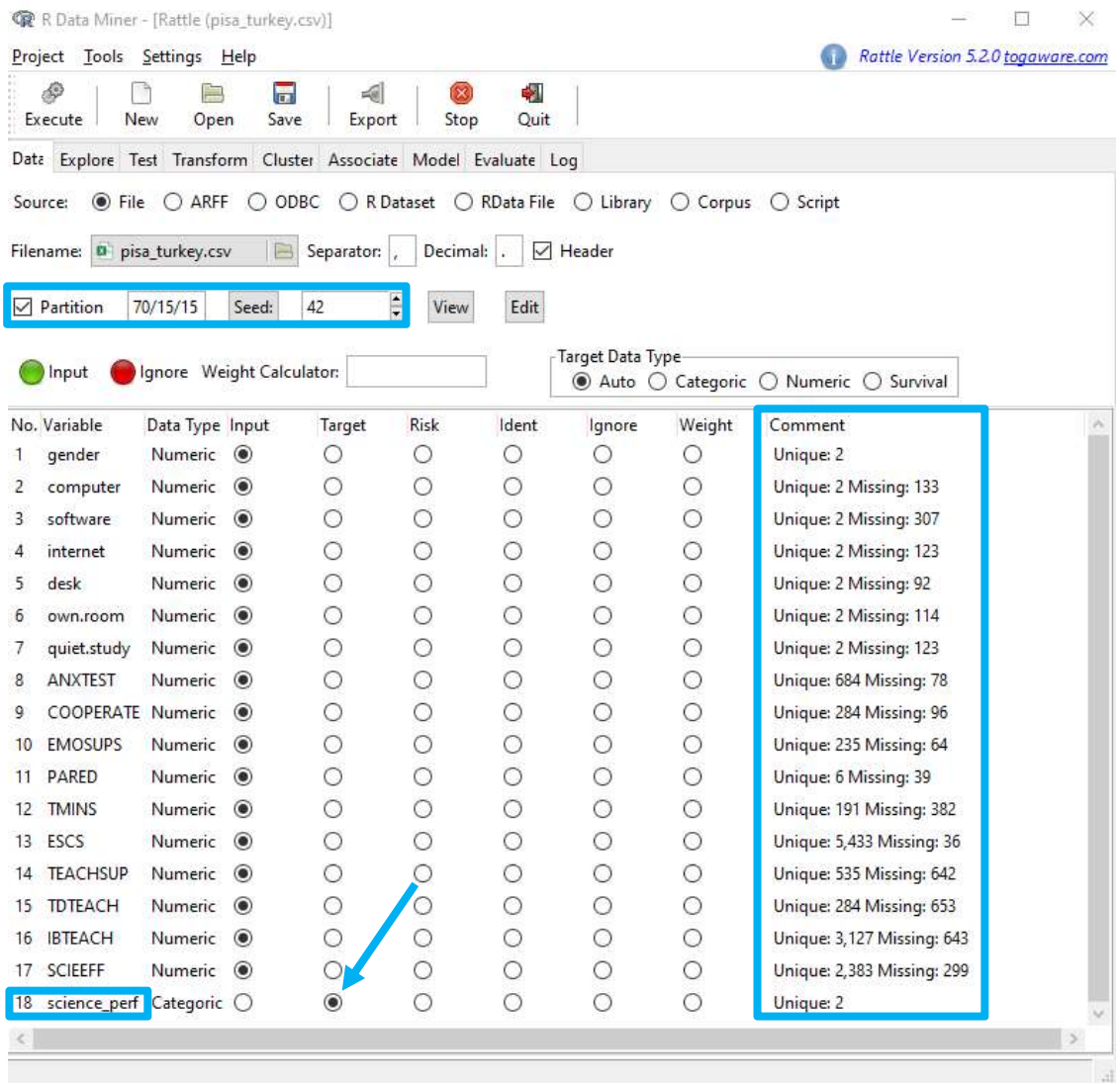

Figure 3. The view of the "pisa_turkey.csv" dataset

3. RESULTS/FINDINGS

This section demonstrates how to implement the CART, RF, and SVM algorithms for the prediction of students' proficiency status in the scientific literacy test. For each algorithm, the *rattle* will require users to download and install the required packages for the first-time implementation. Therefore, users should accept and install the suggested packages if the *rattle* shows any warning messages about downloading and installing such packages. By default, the *rattle* should be able to recognize "science_perf" as the target variable. However, if this is not the case, it must be specified as "Target" under the Data option before running the subsequent analyses (see Figure 3). Once the "Partition" option is checked, the *rattle* splits the dataset into three parts: training dataset (70% of the dataset), test dataset (15% of the dataset), and the validation dataset (the remaining 15% of the dataset). These partitions are created using random sampling based on the seed value (default = 42) under the Data tab. Using the same seed ensures that the user can get the same randomly drawn training, test, and validation datasets every time the *rattle* is used for the same dataset. The training dataset is used for model building and the other two datasets are used for evaluating the accuracy of predictions made from the model. Alternatively, two datasets (e.g., training with 70% and validation with 30%) can be created by typing 70/30 inside the "Partition" box.

3.1. Classification and regression trees (CART)

The first two steps to build a decision tree using the CART approach are to switch to the "Model" tab in the *rattle* and to select the "Tree" option (see Figure 4). Then, the third step is to set the model parameters. "Min Split" is the minimum number of observations that must exist in a node (default = 20); "Min Bucket" is the minimum number of observations in any terminal node (default is Min Split/3); "Max Depth" is the maximum depth of any node of the final tree (default = 3); and "Complexity" is the complexity parameter to prune the subtrees that do not improve the overall model fit (default = 0.01). If this parameter is set to zero, then the CART algorithm keeps all the estimated nodes and typically creates a highly complex model that might be hard to interpret. However, a large value for the complexity parameter might also be detrimental to the model because it would remove many useful nodes from the model and leave a simple model with a very low predictive accuracy. Therefore, users are recommended to build several models by tuning the model parameters based on resulting model evaluation indices (e.g., accuracy, sensitivity, and recall). The fourth step is to click on the "Execute" button – which runs the CART algorithm based on the requested settings. The CART algorithm uses all the variables selected as "input" under the Data tab to predict the target variable (science_perf). Once the estimation is complete, the results can be printed on the screen by clicking on the "Rules" button. Furthermore, visualizations can be drawn for the final decision tree model by clicking the "Draw" button. Figure 4 illustrates the steps to be followed to implement the CART approach and the output returned from the *rattle*.

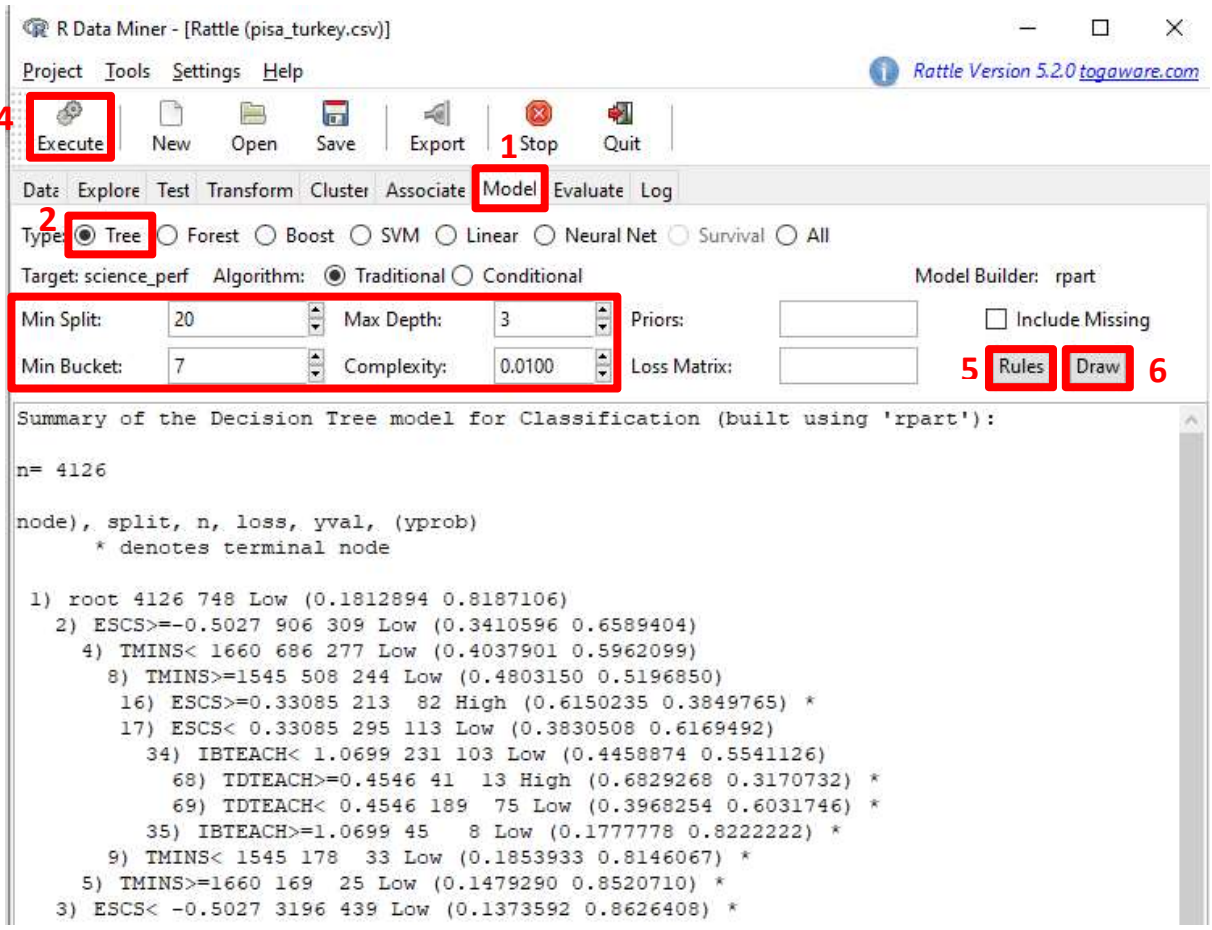

Figure 4. Building a predictive model with the CART algorithm

The output returned from the CART model shows the rules that were used to create the nodes in the decision tree. The output shows the decision nodes and the terminal nodes that were specified with *. For example, a decision node was created based on ESCS (index of economic, social and cultural status) at the beginning of the tree. Based on whether students' ESCS index values were equal or larger than -0.5027, two branches were created in the decision tree model. Then, the group of students who meet the ESCS condition is split into two additional branches depending on whether their total learning time (TMINS) is less than 1660 minutes. The remaining nodes can be interpreted in a similar manner. A relatively easier way to see all the nodes in the model is to draw a decision tree plot. The "Draw" option under the Model tab generates a decision tree plot based on the nodes summarized in the output. Figure 5 shows the decision tree plot returned from the *rattle* for the prediction of the proficiency status in scientific literacy (i.e., science_perf).

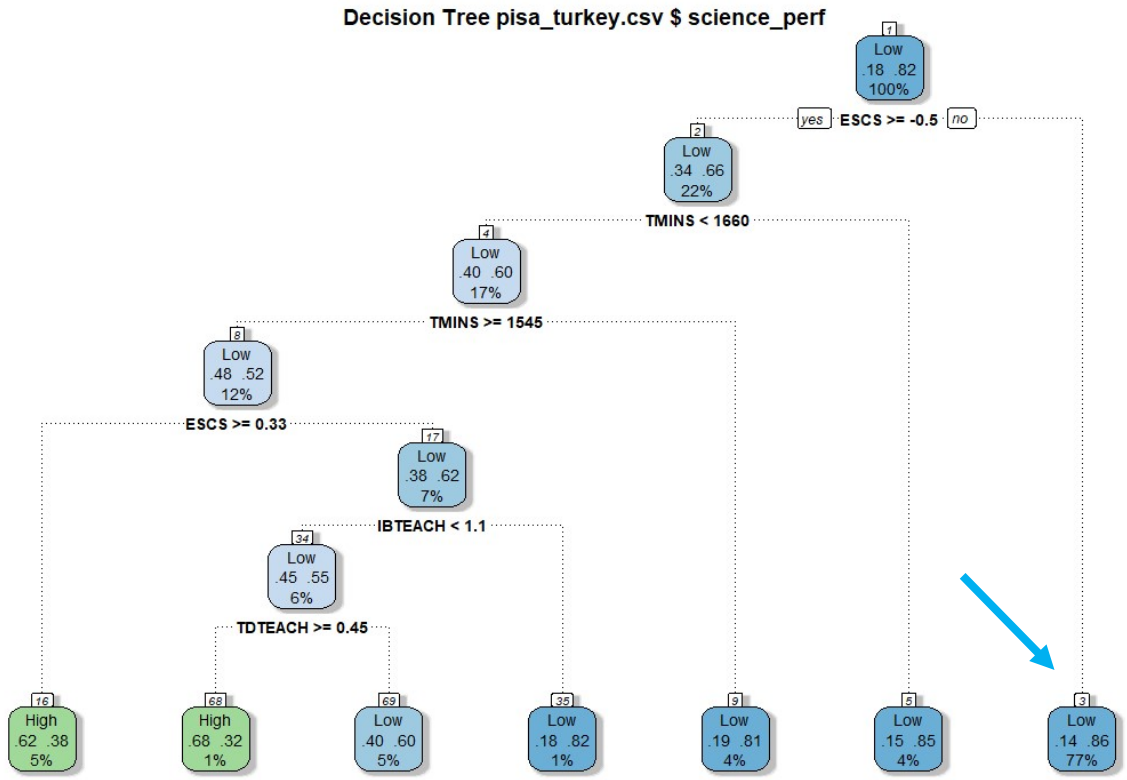

Figure 5. The decision tree plot for the prediction of science_perf

In Figure 5, the categories of science_perf are color-coded where the blue color boxes represent the "Low" category and the green color boxes represent the "High" category. Within each box, the two values in the middle represent the probabilities of the first and second categories. For example, the first terminal node on the right-hand side of the plot shows that students who have ESCS index values smaller than -0.5 have the probabilities of 14% of being in the "Low" category and 86% of being in the "High" category. The number at the bottom of each box represents the percentage of observations in the node. Focusing on the same blue box from the previous example, 77% of the students in the training dataset fall into the node where ESCS is smaller than -0.5. Figure 5 also shows that only four of the input variables (ESCS, IBTEACH, TDTEACH, and TMINS) were used as the predictors. This is because the CART algorithm keeps the predictors that can significantly contribute to the prediction, depending on the selected model parameters (e.g., complexity, min split, and max depth).

3.2. Random forest (RF)

To implement the RF algorithm for the same classification task (i.e., predicting science_perf) in the *rattle*, the "Forest" option must be selected under the "Model" tab. Then, the model parameters need to be determined. "Trees" refer to the numbers of decision (or regression) trees to be built (default = 500); "Variables" is the number of predictors randomly sampled as candidates at each split (default = square root of the number of predictors for classification and the number of predictors / 3 for regression); and "Sample Size" is the sizes of sample to draw (default = 0.632 * the number of observations in the training dataset). Once the model parameters are determined, the next step is to click on the "Execute" button to perform the analysis. Like the CART algorithm, the RF algorithm also uses all of the input variables to predict the target variable (science_perf). Once the estimation is complete, the results can be printed on the screen by clicking the "Rules" and "Importance" buttons. Figure 6 shows the steps to be followed to implement the RF algorithm and to view the output in the *rattle*.

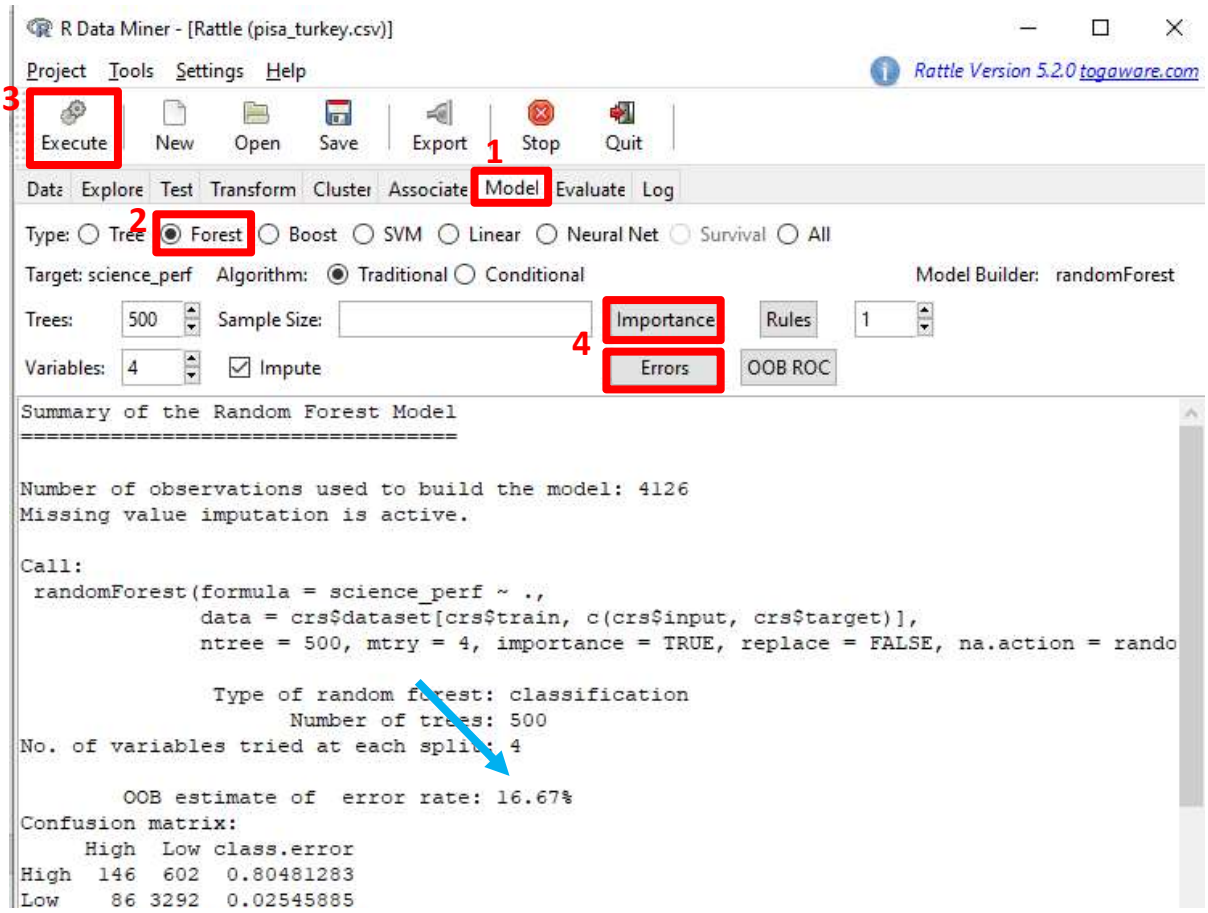

Figure 6. Building a predictive model with the RF algorithm

The output returned from the RF algorithm shows the number of observations used for building the model, the formula used to build the predictive model, and the selected model parameters. The output also shows additional information, such as the out-of-bag (OOB) estimate of the error rate and the confusion matrix. OBB is a method of measuring the prediction error of a predictive model estimated with the RF algorithm. In this example, the OOB estimate of error rate is 16.67 %, suggesting that 83.33 % of the predictions made for science_perf is correct within the training dataset. Additionally, the visual output returned from the "Importance" and "Errors" indicates the importance of the predictors in the prediction process (Figure 7) and error

rates across all of the decision trees built for the model (Figure 8). Based on the variable importance measures shown in Figure 7, ESCS, IBTEACH, TMINS, and computer appear to be the strongest predictors in the estimated model since they have higher importance values, compared to the other variables. Although there is no particular cut-off value to determine which predictors are more important, the predictive power of these variables appears to be relatively higher than the other variables (see the "High" category of the top-left corner of Figure 7). The findings also suggest that the model error rates did not change after 100 trees. That is, the same model could be estimated with only 100 trees to obtain the final model more efficiently.

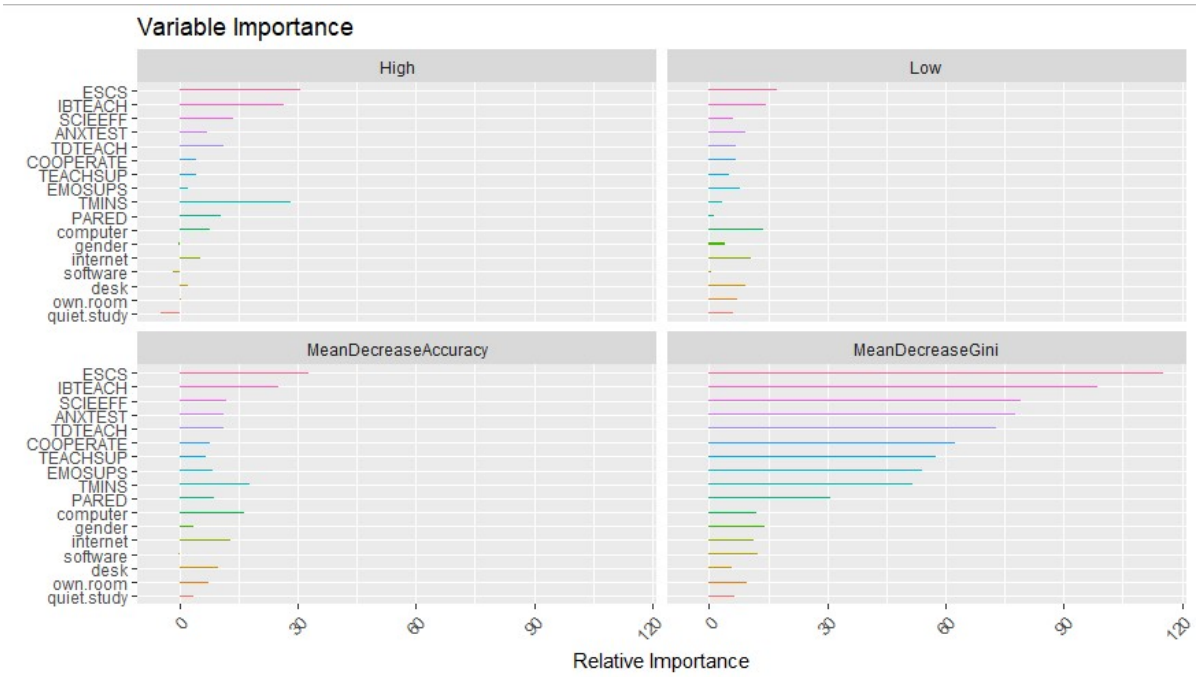

Figure 7. A plot of variable importance for the RF algorithm

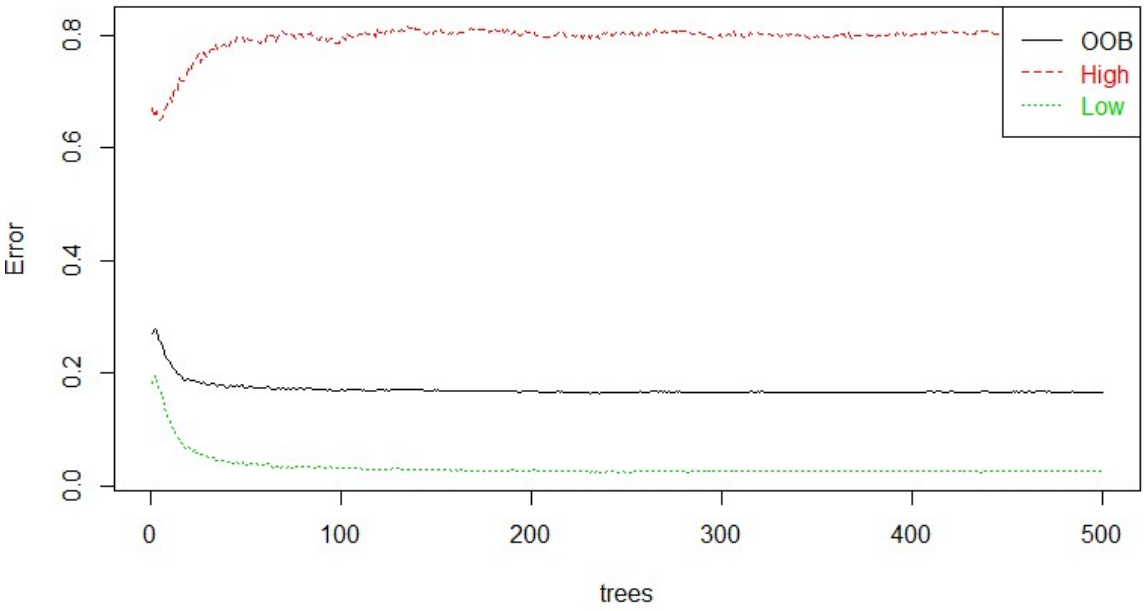

Figure 8. A plot of error rates for the RF algorithm

3.3. Support vector machine (SVM)

To implement the SVM algorithm for predicting science_perf, the "SVM" option must be selected under the "Model" tab. Unlike the CART and RF algorithms, there are not many model parameters to choose for the SVM algorithm. Instead, the most important decision that users must make is the selection of a kernel function. The default kernel function in the *rattle* is "rbfdot", which refers to the Gaussian radial basis function. The "rbfdot" function is a general-purpose kernel suitable for cases where there is no prior knowledge about the data. There are also other popular kernel functions available for the SVM algorithm, such as "polydot" for the polynomial kernel function and "vanilladot" for the linear kernel function. Non-linear kernels often provide a better model-data fit than linear kernels at the expense of high computational complexity and estimation time. Once a kernel is selected, the next step is to click on the "Execute" button to perform the analysis. Like the previous algorithms, the SVM algorithm also utilizes all of the selected input variables to predict the target variable (science_perf). Once the estimation is complete, the results are printed on the screen. Figure 9 shows the steps to be followed to implement the SVM algorithm in the *rattle*.

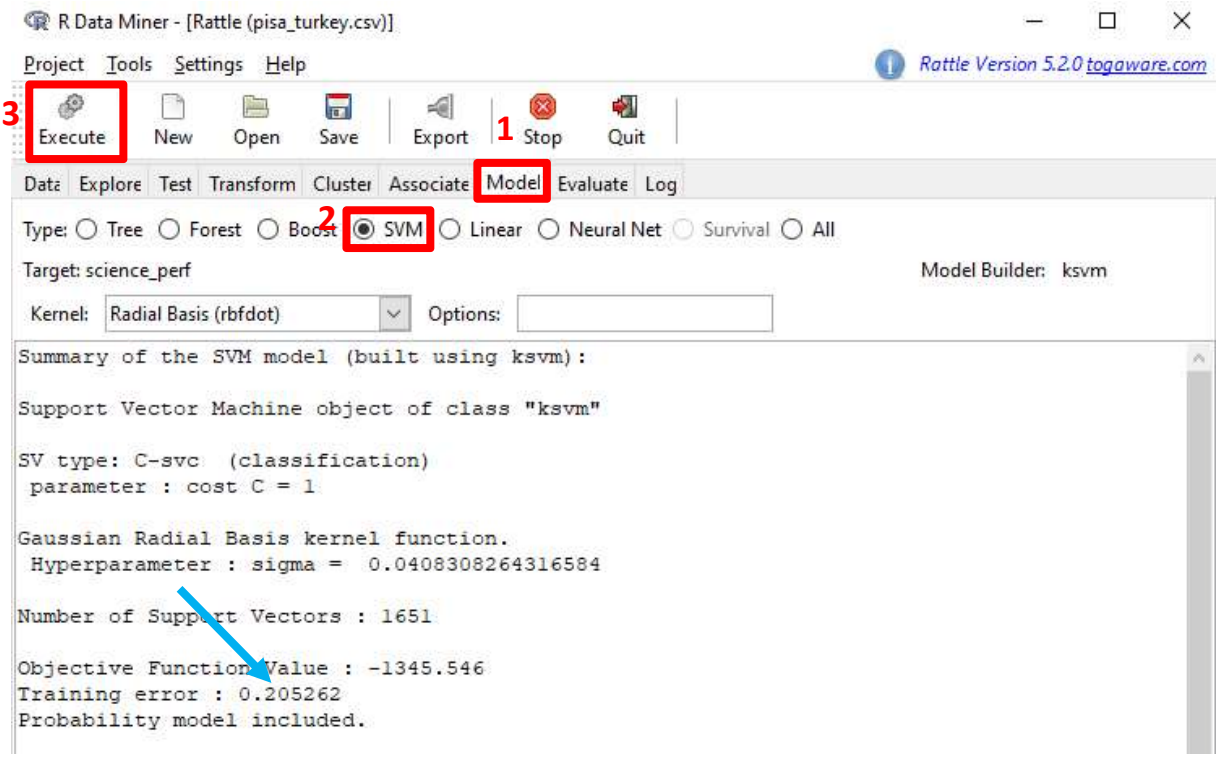

Figure 9. Building the support vector machine classification model

The output returned from the SVM algorithm shows the default settings used in the estimation process (the cost parameter of C as 1 and the hyperparameter of sigma as 0.0408). In addition, the output shows the number of support vectors created in the model (1651). An important section of the output is "Training error". The results show that the overall prediction error in the training dataset was around 20.53%. That is, roughly 80% of the predictions made for science_perf in the training dataset are accurate. It should be noted that although the output resulted from the SVM algorithm is quite concise in the *rattle* compared to those from the CART and RF algorithms, it is often much more difficult to interpret the content of this output given the complex hyperparameters used in the SVM algorithm.

3.4. Evaluating models

Unlike traditional statistical methods, the data mining methods require researchers to build several models, evaluate outcomes from each model, adjust the models accordingly, and continue to tune the models until an acceptable level of accuracy is reached. As demonstrated in this tutorial, several algorithms can also be used for the same classification or regression task. Therefore, researchers must not only tune their models but also select the most suitable algorithm based on the model evaluation measures. In the *rattle*, model evaluation can be performed using the options under the "Evaluate" tab. For model evaluation, either validation or test datasets should be used because these datasets consist of the observations that the algorithms have not seen when building the prediction model. Figure 10 shows the steps to view the error matrix from the SVM algorithm, although the same steps can be followed to see the same output for other algorithms as well. This tutorial focused on the prediction of a binary outcome variable (science_perf), and thus the error matrix returns a two-by-two matrix of predicted and actual values and proportions of the two categories (i.e., "High" and "Low" proficiency in scientific literacy). The overall prediction error for the SVM-based model is 21% and the average classes (i.e., category) error is 49.4%. The error matrix shows that the prediction accuracy of the "Low" category was precise, whereas the prediction accuracy of the "High" category was quite poor.

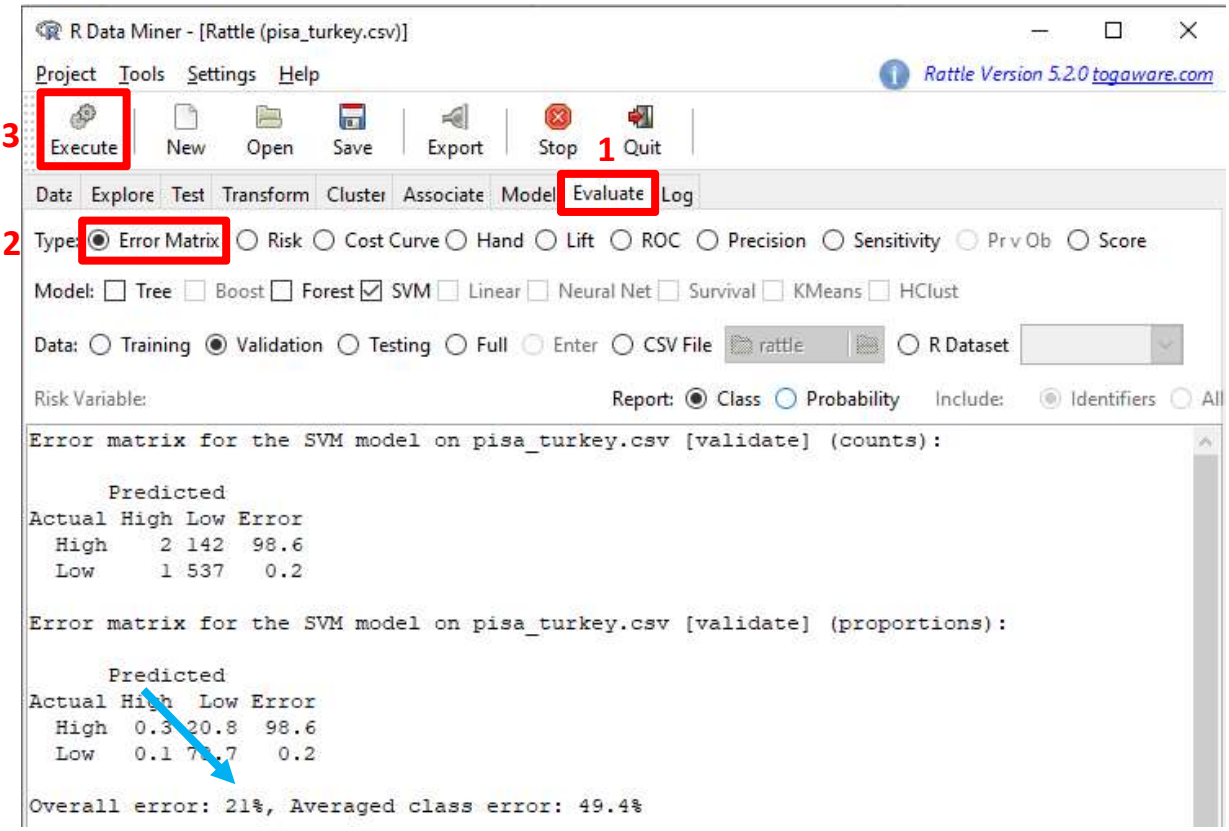

Figure 10. The view of the "Evaluate" tab in *rattle*

The "Evaluate" tab offers many useful measures for model evaluation. For example, the sensitivity, specificity, precision, and recall plots can be created using the "Sensitivity" and "Precision" options under the "Evaluate" tab. Users must select one of these evaluation options and click "Execute" to draw the plots. Table 2 shows the calculation of the evaluation measures available in the *rattle*.

Table 2. Evaluation measures for the classification of "Low" and "High" groups in science_perf

Predicted Classification	Actual Classification	
	Low proficiency in science	**High** proficiency in science
Low proficiency in science	True Positive (TP)	False Positive (FP)
High proficiency in science	False Negative (FN)	True Negative (TN)

Note: Sensitivity = TP/TP+FN; Specificity = TN/TN+FP; Precision = TP/(TP+FP); Recall = TP/(TP+FN)

To compare the results from different data mining algorithms, the models must be estimated with each algorithm first so that the evaluation measures under the "Evaluation" tab can draw the plots by including the results from all algorithms. Figures 11 and 12 show the plots of sensitivity/specificity and precision/recall across the three data mining algorithms (i.e., CART, RF, and SVM). Figure 11 shows that there is a significant trade-off between sensitivity and specificity for all the algorithms. As the specificity level (i.e., detecting "High") increases, the sensitivity level (i.e., detecting "Low") decreases. Among the three algorithms, the performance of the RF algorithm appears to be the best in terms of balancing sensitivity and specificity. Figure 12 shows the precision and recall levels across the three algorithms. The results suggest that all the algorithms indicate high precision and recall values in predicting the "High" and "Low" values of science_perf. Given the similar precision and recall values across the three algorithms, sensitivity and specificity can be more decisive evaluation measures for the example presented in this study.

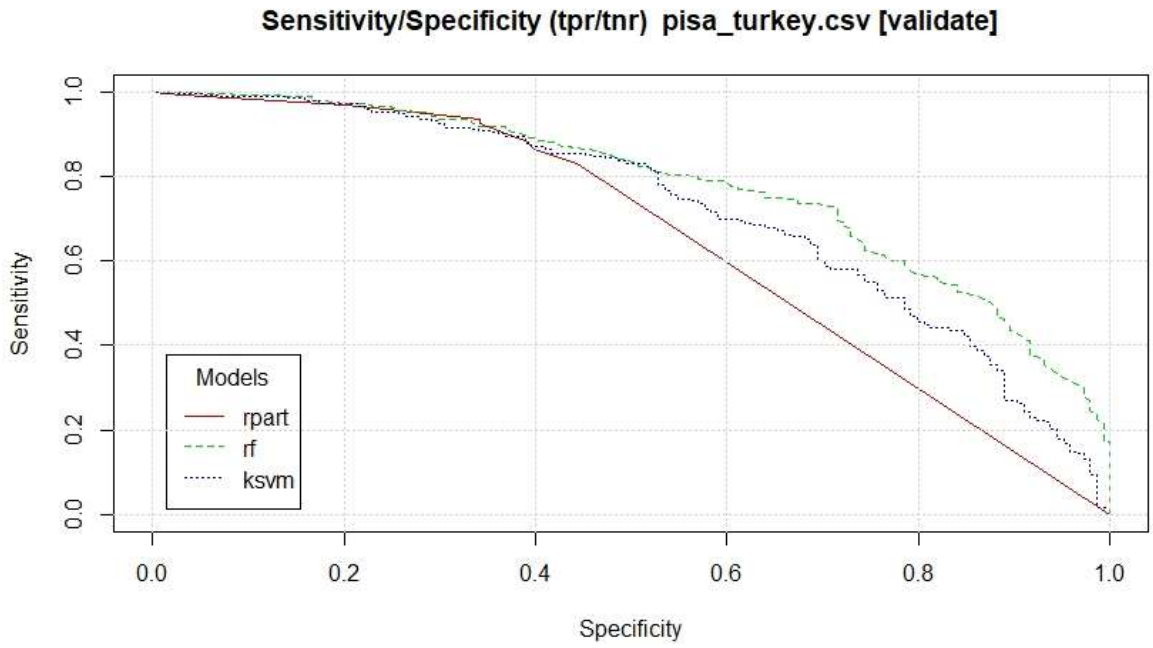

Figure 11. The sensitivity and specificity plots of the three data mining algorithms (**Note**: rpart refers to CART, rf refers to RF, and ksvm refers to SVM).

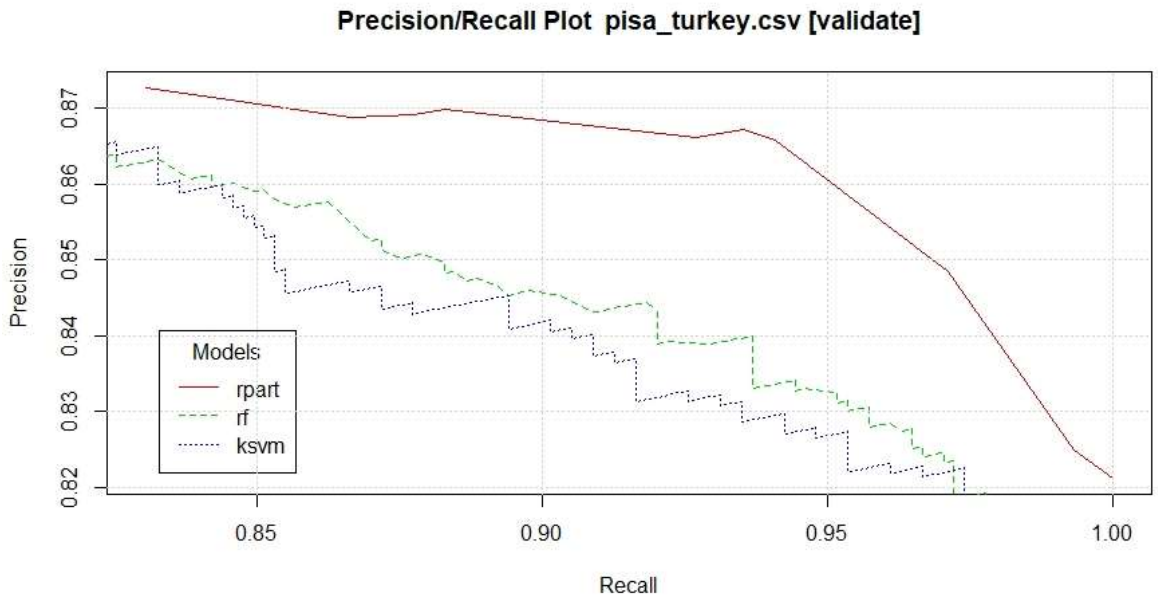

Figure 12. The precision and recall plots of the three data mining algorithms (**Note**: rpart refers to CART, rf refers to RF, and ksvm refers to SVM).

4. DISCUSSION and CONCLUSION

The purpose of this study was to demonstrate the implementation of the three data mining algorithms (i.e., CART, RF, and SVM) using the *rattle* package (Williams, 2011) in R (R Core Team, 2019). The selected algorithms are widely used methods in EDM research for both classification and regression tasks. The example used in this study demonstrated how to build classification models using the CART, RF, and SVM algorithms for predicting students' proficiency levels (low or high) in the scientific literacy test of PISA 2015. In addition, the model evaluation stages were also described.

Based on the results of this study, the RF algorithm appeared to be the best performing algorithm for predicting students' proficiency levels in the scientific literacy test of PISA 2015. This is not a surprising finding because the RF algorithm often provides accurate prediction results in datasets that contain both numerical and categorical predictors (i.e., features). These findings tie well with a previous study wherein Fernández-Delgado et al. (2014) compared the performances of 179 classification algorithms using 121 real datasets. The researchers found that the RF algorithm was the best algorithm for most real world classification problems, followed by the SVM algorithm. A similar pattern of results was obtained in the current study.

The results from the three algorithms were somewhat different in this study mainly because each algorithm handles different types of variables and their relationships in the pisa_turkey dataset. For example, the CART algorithm yielded sensitivity and specificity values similar to those from the other two algorithms, but it used fewer predictors in the estimation. Depending on what complexity parameter has been selected, the decision tree model can either retain or eliminate the subtrees created based on relatively less important predictors in the dataset. Furthermore, when some predictors are highly correlated, the CART algorithm may choose only one of those predictors and ignore the others. Therefore, researchers are recommended to choose an algorithm and tune its parameters after careful consideration and review of their data.

As a free software program, the *rattle* uses many powerful packages available within the R computing environment for conducting data mining analysis. Unlike the R software program that requires users to type and execute their codes, the *rattle* provides a user-friendly GUI that enables users to import their data files easily, select an algorithm from a variety of options, and evaluate the results using model evaluation measures. The output returned from the *rattle*

involves both statistical and visual outcomes to facilitate users' evaluation and fine-tuning of their models. Although it is not demonstrated in this study, the *rattle* is also capable of providing users with the opportunity to explore their datasets descriptively and to transform variables (e.g., rescaling, recoding, and normalizing) before performing further analysis. In addition, the *rattle* is capable of performing unsupervised data mining, including clustering with *k*-means and hierarchical clustering methods and association rule analysis. For advanced R users who might prefer to keep the R codes for their analysis, the *rattle* provides a script that presents the underlying R codes for all analyses conducted in the program under the "Log" tab. For a comprehensive review of the *rattle*, readers are recommended to check out *Data Mining with Rattle and R* by Williams (2011) who is also the author of the *rattle*.

Acknowledgements

This study used some materials from the workshop of "Exploring, Visualizing, and Modeling Big Data with R" held at the 2019 annual meeting of the National Council on Measurement in Education, Toronto, ON, Canada.

ORCID

Okan Bulut https://orcid.org/0000-0001-5853-1267
Hatice Cigdem Yavuz https://orcid.org/0000-0003-2585-3686

5. REFERENCES

Agarwal, S., Pandey, G. N., & Tiwari, M. D. (2012). Data mining in education: Data classification and decision tree approach. *International Journal of e-Education, e-Business, e-Management and e-Learning, 2*(2), 140.

Aldowah, H., Al-Samarraie, H., & Fauzy, W. M. (2019). Educational Data Mining and Learning Analytics for 21st century higher education: A Review and Synthesis. *Telematics and Informatics, 37*, 13-49.

Aulck, L., Velagapudi, N., Blumenstock, J., & West, J. (2016). Predicting student dropout in higher education. *arXiv preprint arXiv:1606.06364*.

Baker, R. S., Martin, T., & Rossi, L. M. (2017). Educational data mining and learning analytics. In A. A. Rupp & J. P. Leighton (Eds.), *The handbook of cognition and assessment: Frameworks, methodologies, and applications* (pp. 379-396). Oxford, UK: John Wiley & Sons, Inc.

Berland, M., Baker, R. S., & Blikstein, P. (2014). Educational data mining and learning analytics: Applications to constructionist research. *Technology, Knowledge and Learning, 19*(1-2), 205-220.

Breiman, L. (2001). Random forest. *Machine Learning, 45*(1), 5–32.

Breiman, L., Friedman, J. H., Olshen, R. A., & Stone, C. J. (1984). *Classification and regression trees*. Belmont, CA: Wadsworth.

Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine learning, 20*(3), 273-297.

Ducange, P., Pecori, R., Sarti, L., & Vecchio, M. (2016, October). Educational big data mining: how to enhance virtual learning environments. In *International Joint Conference SOCO'16-CISIS'16-ICEUTE'16* (pp. 681-690). Springer, Cham.

Dutt, A., Ismail, M. A., & Herawan, T. (2017). A systematic review on educational data mining. *IEEE Access*, 5, 15991-16005.

Fernández-Delgado, M., Cernadas, E., Barro, S., & Amorim, D. (2014). Do we need hundreds of classifiers to solve real world classification problems? *The Journal of Machine Learning Research*, *15*(1), 3133-3181.

Guruler, H., Istanbullu, A., & Karahasan, M. (2010). A new student performance analysing system using knowledge discovery in higher educational databases. *Computers & Education, 55*(1), 247-254.

Hussain, M., Zhu, W., Zhang, W., Abidi, S. M. R., & Ali, S. (2019). Using machine learning to predict student difficulties from learning session data. *Artificial Intelligence Review*, *52*(1), 381-407.

Ivancevic, V., Celikovic, M., & Lukovic, I. (2011). Analyzing student spatial deployment in a computer laboratory. In *Proceedings of the 4th international conference on educational data mining* (pp. 265–270).

Koon, S., & Petscher, Y. (2015). *Comparing methodologies for developing an early warning system: Classification and regression tree model versus logistic regression. REL 2015-077*. Regional Educational Laboratory Southeast.

Koon, S., & Petscher, Y. (2016). *Can scores on an interim high school reading assessment accurately predict low performance on college readiness exams? REL 2016-124*. Regional Educational Laboratory Southeast.

Lawrence, M., & Lang, D. T. (2010). RGtk2: A ghraphical user interface toolkit for R. *Journal of Statistical Software, 37*(8), 1-52.

Mccuaig, J., & Baldwin, J. (2012). Identifying successful learners from interaction behaviour. In *Proceedings of the 5th international conference on educational data mining* (pp. 160–163).

Mostafa, T., Echazarra, A., & Guillou, H. (2018). *The science of teaching science: An exploration of science teaching practices in PISA 2015*. OECD Education Working Papers, No. 188. Paris, France: OECD Publishing.

OECD (2017). PISA 2015 *Assessment and Analytical Framework: Science, Reading, Mathematic, Financial Literacy and Collaborative Problem Solving*. PISA, OECD Publishing, Paris, https://doi.org/10.1787/9789264281820-en

OECD (2018). PISA 2015 results in focus. Retrieved from https://www.oecd.org/pisa/pisa-2015-results-in-focus.pdf

Pardos, Z. A., Wang, Q. Y., & Trivedi, S. (2012). The real world significance of performance prediction. In *Proceedings of the 5th international conference on educational data mining* (pp. 192–195).

Peña-Ayala, A. (2014). Educational data mining: A survey and a data mining-based analysis of recent works. *Expert System with Applications, 41*(4), 1432-1462. http://dx.doi.org/10.1016/j.eswa.2013.08.042

R Core Team (2019). *R: A language and environment for statistical computing*. Vienna, Austria: R Foundation for Statistical Computing.

Sinharay, S. (2016). An NCME instructional module on data mining methods for classification and regression. *Educational Measurement: Issues and Practice, 35*(3), 38–54. http://dx.doi.org/10.1111/emip.12088

Slater, S., Joksimović, S., Kovanovic, V., Baker, R. S., & Gasevic, D. (2017). Tools for Educational Data Mining: A Review. *Journal of Educational and Behavioral Statistics, 42*(1), 85–106. https://doi.org/10.3102/1076998616666808

Spikol, D., Ruffaldi, E., Dabisias, G., & Cukurova, M. (2018). Supervised machine learning in multimodal learning analytics for estimating success in project-based learning. *Journal of Computer Assisted Learning*, *34*(4), 366-377.

Strobl, C. (2013). Data mining. In T. Little (Ed.), *The Oxford handbook of quantitative methods in psychology* (Vol. 2, pp. 678–700). New York, NY: Oxford University Press.

Venables, W. N., Smith, D. N., & the R Core Team (2019). An introduction to R. Retrieved from https://cran.r-project.org/doc/manuals/r-release/R-intro.pdf

Williams, G. J. (2011). *Data mining with Rattle and R: The art of excavating data for knowledge discovery*. New York: Springer-Verlag.