The use of Machine Learning Methods in User Knowledge Model Analysis and Classification

Carlos Gomes de Oliveira*, Eduardo Terra Morelli†, and Cassius Figueiredo‡
Instituto INFNET
Escola Superior da Tecnologia da Informação
Rio de Janeiro, RJ
Email: *carlos.gomes@al.infnet.edu.br, †eduardo.morelli@prof.infnet.edu.br, ‡cassius.figueiredo@prof.infnet.edu.br

***Abstract*—In most schools and training facilities, exams or tests measure the student's knowledge on a given topic, usually placed after the studying phase. A prediction model would allow us to redirect the study focus and follow the study progression on an individual basis, with the intention to maximize the knowledge on each subject.**

On this paper is to show the use of tree-based machine learning methods such as Decision Trees, Bootstrapped Aggregation, and Random Forests to classify the data in proper classes. These methods were not selected at random, as they allow us to find the most useful attributes and have this usefulness quantified. That allows us to adjust the model to maximize its performance.

Once trained, this model can also predict the future performance of each student. With that result, we use his knowledge and needs to tailor the study contents and exams.

***Index Terms*—classification, knowledge acquisition, Decision Trees, Extreme Gradient Boosting, XGBoost, Random Forests, Logistic Regression, User Knowledge Modeling, data set**

I. Introduction

We obtained this dataset from Kahraman's [1] Ph.D. Thesis, where his goal was to develop an object model to form the domain dependent data of adaptive learning environments. Since the data covers student earlier knowledge and current studying efforts, to use prior information to classify the current knowledge is a valuable resource for an adaptive educational system [2], [3], [4].

The objective of this paper is to show how to prepare a data set, analyze its features, and build prediction models that give us insight on the data. The initial analysis of the features shows each feature cannot alone define the classes for the data, so they must be correlated to design a good prediction model. The main challenge is to train a model with just a few hundred lines of data.

This can be accomplished through machine learning methods.

II. Related Work

In the data age, the use of machine learning algorithms is widely used for prediction, for example, Cornel and Mirela used Decision Trees [5] to predict economic forecasts for a university [6], by the American Rheumatism Association in 1987 to revise criteria for the classification of rheumatoid arthritis [7], and to predict rates of relapse in subgroups of male and female smokers [8].

Random Forests have been used from Gene Classification [9] to Image Classification [10].

Extreme Gradient Boosting [11] have been used to study fish species richness in the oceans surrounding New Zealand [12] and to classify remotely sensed imagery [13].

III. Machine Learning Methods in User Knowledge Model Analysis and Classification

In this paper, we used Decision Trees (ctree), Recursive Partitioning and Regression Trees (rpart), the C4.5 Algorithm (J48), PART, Bootstrapped Aggregation (bagging), Random Forest and C5.0 Algorithm (C5.0).

Decision Trees had been used for its long history and well-documented results as a baseline for comparison to other methods.

Since RF introduction in 2001 [14], it has been widely used in data classification and regression, and more recently XGBoost has shown great results and its acceptance is growing as demonstrated in several Kaggle [15] competitions.

Before training any model the data should be analyzed, cleansed, and imputed. Data cleansing is a process of transforming the original data in a way that keeps its accuracy and improves its usability by programs, especially machine learning algorithms, that are very sensitive to discrepancies in data. One data cleansing procedure is described in (Real-world data is dirty: Data cleansing and the merge/purge problem) [16]. Finally, imputation is a statistical method to fill in missing values [17].

A frequent concern is to be sure if the model is not overfitting. As stated by Douglas M. Hawkins (The problem of overfitting) [18], over-fitting is the use of models that include more terms than are necessary or use more complicated approaches than are necessary.

A. Random Forests

A special mention to Random Forests, this algorithm usually stands out in good results. We used Random Forests to perform classification analysis on the knowledge model data set. RF is a powerful and resilient algorithm in comparison to other top performer algorithms. It is a variation of the basic supervised learning model implementing decision trees, but creating a multitude of trees, hence its name.

The resulting class is collected from these trees and the most frequent classification of regression is a step in the right direction of obtaining a pure (correct) result. Since it was branded and introduced by Breiman (Breiman, 2001), it has proved its usefulness, especially on its strengths as outlined in his original paper [14]:

- Its accuracy is considered as good as or better than Adaboost.
- It is quite robust to outliers points and noise in data.
- I also return extremely useful information about the model, such as the variable importance, internal estimates of error, strength, and correlation.
- It is considered faster than traditional bagging or boosting.
- It can be easily parallelized and it is simple to use.
- As a result of the consolidation of the parallel trees, it does not overfit.
- And finally, it performs similarly on both continuous and categorical features in the dataset. All these characteristics are mentioned by those who increasingly use it in fields from image analysis and genetics to application log and business data classification and regression analysis.

IV. Experiments and Results

The objective of this experiment is to analyze data collected and find patterns in the data. This dataset was obtained from the UCI Machine Learning Repository [19]. We used the UNS - the knowledge level of user attribute as a prediction class and built a prediction model on a selection of the other features to predict this class. We expect to find in attribute relevance to the model accuracy an important insight into the data and the problem of predicting this class.

This database was obtained from a Ph.D. thesis [1] and donated to the UCI repository. It has a relatively small number of records, this represents an additional difficulty in our model.

According to [2], the data represent knowledge of students about the domain dependent data, this dynamic data in user model might be also called the user knowledge model [20]. The data was obtained from the interaction of the students with the web-environment by the user modeling system (UMS) [20]. In [2], the generic object model developed by Kahraman in the Adaptive Educational Electric Course (AEEC) [1] were used to form application domain (domain model) and knowledge model of the students.

For example, when looking to a specific educational objective that has specific intrinsic features such as the study time interval or duration, the repetition number of study sessions, the difficulty level, and the questions being studied. This objective also has other objectives as prerequisites, and some features related to these prerequisites as the interval or duration, the knowledge level to be learned, and the questions [2].

Furthermore, let's explain the parameters. UMS classifies the knowledge levels (UNS) of users depending on the real values of these features. There are five different features (STG, SCG, PEG, STR, LPR). STG, SCG, and PEG are describe the learning objects and the others describe the prerequisite objects used to classify the current knowledge of a user about the learning objects in AEEC [2]. The definition of users' features; the degree of study time (STG), the degree of repetition number (SCG) and the user performance in exams (PEG) for the learning object, the degree of study time (STR) and the learning percentage (LPR) of users for prerequisites objects [2]. The current knowledge of students (UNS) is determined using real values of (STG, SCG, PEG, STR, LPR) as input parameters of the user modeling algorithm in AEEC [2].

In [2] the author explains that these features are obtained from the UMS model as it tracks and collects the users' data such as learning activities/feedbacks/answers/navigation paths about the learning objects and prerequisite objects. Reading texts, solving problems/exercises/tests, navigation in the different pages of a learning environment are several examples of available on-line data. In further steps, this data is converted in our studied features.

The dataset is already split into train and test data. There are 258 observations in train data, 145 observations in test data, and 403 observations in total. The proportion is 64% of all observations for train data and 36% of all observations for test data.

A. Feature Analysis and Engineering

The first step is to check the need for some feature engineering had to be performed. The first phase is data cleansing. To do that we have to get an overview of the data-set.

In Fig. 1, for example, we can see a barplot with all the features, in descending order of the number of distinct values for each feature. At the end, separated by a vertical line, is our dependent variable. We checked but there are no missing values.

The next step is to analyze the data distribution. In Table I we have some basic statistics on the numerical independent features. The minimum value is 0 and the maximum value is near 1. The first quarter seems balanced among all features, but we notice discrepancies in the median, mean and the 3rd quarter. The categorical feature in Table II does not have a higher number of values on the ”Middle” value but on the ”Low” value. This means that our features do not have similar distributions, so we have to dig deeper into these features.

Next, we looked into the data distribution of all features, in Fig. 2 we have boxplots for all numerical features. The

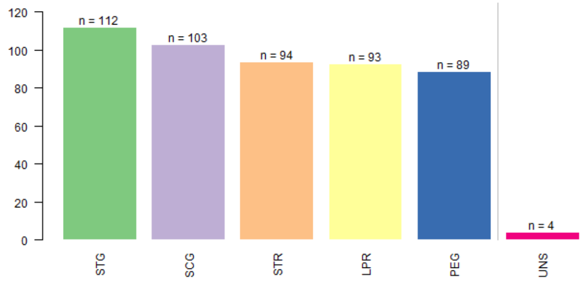

Fig. 1. All Features

TABLE I
Data Statistical Summary - Numerical Features

Feat	Min	1st Q	Median	Mean	3rd Q	Max
STG	0.0	0.2	0.3	0.3531	0.48	0.99
SCG	0.0	0.2	0.3	0.3559	0.51	0.90
STR	0.0	0.265	0.44	0.4577	0.68	0.95
LPR	0.0	0.25	0.33	0.4313	0.65	0.99
PEG	0.0	0.25	0.4	0.4564	0.66	0.99

TABLE II
Data Statistical Summary - Categorical Feature

Very Low	Low	Middle	High
50	129	122	102

first information to stand out are outliers, dots on the upper side of the STG feature boxplot. Remember from the Fig. 1 that the STG feature had the greatest number of distinct values among all numeric independent features. One possible course of action is the discretization of the data. We noticed that the features STG and SCG are very concentrated below the 0.50 line, while the other features are more evenly distributed. Besides, the median, the thick line in the middle of the box, is near the mean, the red dot, only in the STR feature, all other features have the median far from the center of the boxplot.

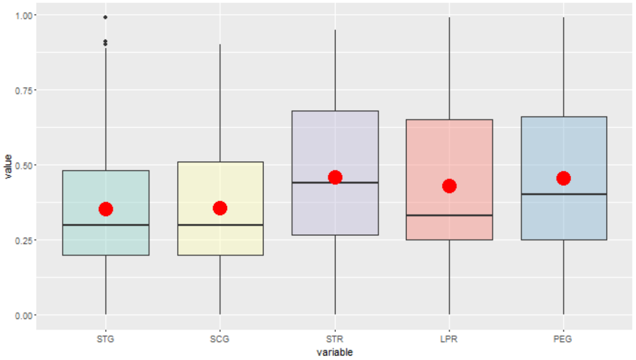

Fig. 2. All Features Boxplot

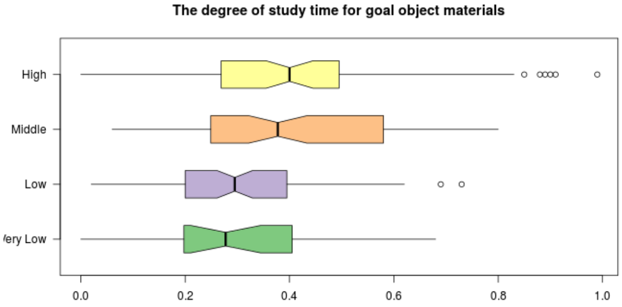

Fig. 3. STG Boxplot

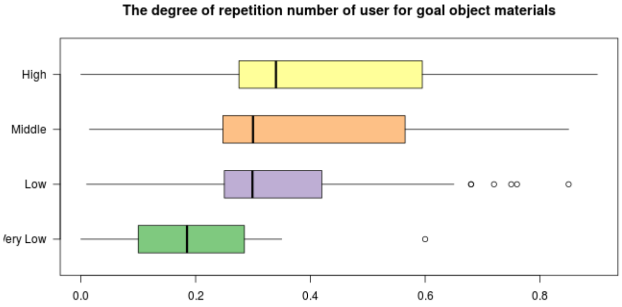

Fig. 4. SCG Boxplot

Also, we look into the STG feature in more detail. We have in Fig. 3 a detailed boxplot with the STG feature values broken into the values for the UNS class. The data distribution is clearly overlapped for the "Very Low" and "Low" class values. In addition, the other values' distributions also overlap significantly. This feature seems to has little value to help make the class predictions.

Then we look into the SCG feature, its boxplot is in Fig. 4. The data distribution for the "Very Low" feature is mostly distinct from the other values, but these other values overlap significantly. This feature have very little value to the model.

Afterward, we analyze the STR feature that, with its boxplot is in Fig. 5 shows a distribution that overlaps for all values of the prediction class. The "Very Low" and "Low" class values' distributions show a median somewhat apart from each other and the other two values, that overlap. This feature seems to carry very little value to the model and can even get in the way of the prediction.

Next to last, we investigate the LPR feature with its boxplot is in Fig. 6. The data distribution for the prediction class values is the most mixed of all, even the distribution of the "Middle" class values is concentrated below the "Very Low" feature value. However, the medians seem somewhat apart from each other, this feature seems to carry some value to the model.

Finally, we get to the last independent feature, PEG, its boxplot is in Fig. 7. The data distribution for the

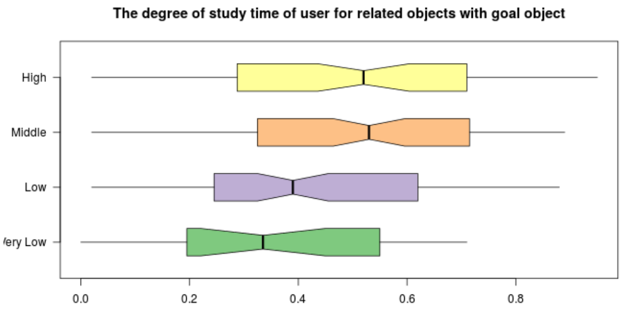

Fig. 5. STR Boxplot

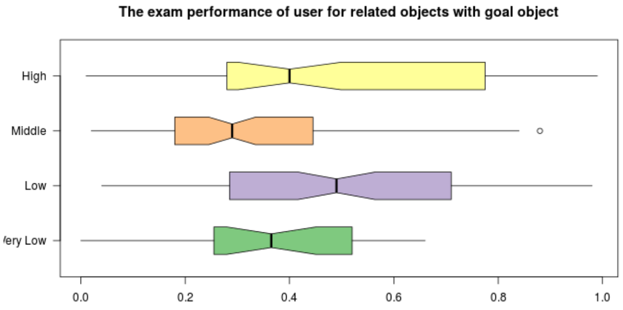

Fig. 6. LPR Boxplot

prediction class values is the most distinctive of all, with all distributions showing very little superposition. Furthermore, between the first and third quartiles, we have no superposition and there is a good distance among them. Besides, the data is mostly concentrated around the median, and the distance between the first and last quartiles for each value of the prediction class is significantly smaller than the other features. Here we have data with enough value to make an outlier analysis. The outliers that could be trimmed are those near the beginning and the end of all distributions. We have only one outlier for the feature ”Low” that could be trimmed near the 0 value. This is the most valuable feature for this model.

Another graph that shows these characteristics of the features is in Fig. 8. It is a scatterplot of all features with the prediction class values represented by the different circle colors. We can see how all the values are mixed with all features, except for the PEG feature, that shows a good separation among the values.

B. Model Execution

At Last, we run our prediction models. We ran Decision Trees (ctree), Recursive Partitioning and Regression Trees (rpart), the C4.5 Algorithm (J48), PART, Bootstrapped Aggregation (bagging), Random Forest and C5.0 Algorithm (C5.0). The initial formula for prediction defines UNS as the dependent variable or prediction class. STG, SCG, STR, LPR, and PEG are our independent variables or prediction features.

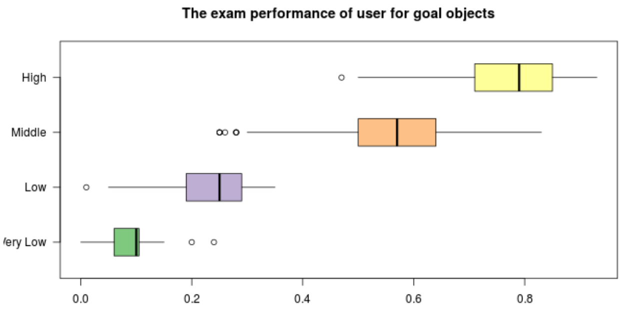

Fig. 7. PEG Boxplot

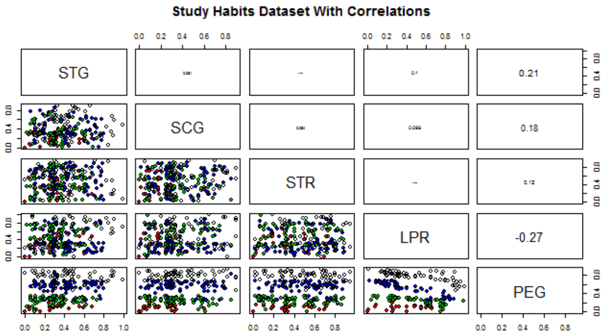

Fig. 8. Independent Variables Scatterplot

TABLE III
Decision Trees

Class Value	Precision	Recall	F1
High	0.9750000	1.0000	0.9873
Middle	0.8823529	0.8824	0.8824
Low	0.8400000	0.9130	0.8750
Very Low	1.0000000	0.8077	0.8936

TABLE IV
Recursive Partitioning and Regression

Class Value	Precision	Recall	F1
High	0.9750000	1.0000	0.9873
Middle	0.8823529	0.8824	0.8824
Low	0.8400000	0.9130	0.8750
Very Low	1.0000000	0.8077	0.8936

To begin with, Decision Trees had an accuracy of 0.9103 and the other performance metrics can be seen in Table III.

For Recursive Partitioning and Regression Trees, we also had an accuracy of 0.9103 and the other performance metrics can be seen in Table IV.

Additionally, the C4.5 (J48), that creates the tree in such a way it maximizes the information gain, we also had an accuracy of 0.9103 and the other performance metrics

TABLE V
C4.5 (J48)

Class Value	Precision	Recall	F1
High	0.9750000	1.0000	0.9873
Middle	0.8823529	0.8824	0.8824
Low	0.8695652	0.8696	0.8696
Very Low	0.9200000	0.8846	0.9020

TABLE VI
PART

Class Value	Precision	Recall	F1
High	0.9750000	1.0000	0.9873
Middle	0.8823529	0.8824	0.8824
Low	0.8809524	0.8043	0.8409
Very Low	0.8275862	0.9231	0.8727

TABLE VII
Bootstrapped Aggregation

Class Value	Precision	Recall	F1
High	0.0000000	0.0000	NaN
Middle	0.8108108	0.8824	0.8451
Low	0.8600000	0.9348	0.8958
Very Low	0.0000000	0.0000	NaN

TABLE VIII
Random Forests

Class Value	Precision	Recall	F1
High	1.0000000	1.0000	1.0000
Middle	0.9687500	0.9118	0.9394
Low	0.8823529	0.9783	0.9278
Very Low	1.0000000	0.8846	0.9388

TABLE IX
C5.0

Class Value	Precision	Recall	F1
High	0.9750000	1.0000	0.9873
Middle	0.8571429	0.8824	0.8696
Low	0.8888889	0.8696	0.8791
Very Low	0.9600000	0.9231	0.9412

can be seen in Table V.

Next, the PART Algorithm, that creates trimmed trees and then extracts the rules, we had an accuracy of 0.8966 and the other performance metrics can be seen in Table VI.

Also, the Bootstrapped Aggregation (Bagging) Algorithm, that uses ensembles to create the trees, we had an accuracy of 0.5034 and the other performance metrics can be seen in Table VII.

In the same way, the Random Forests Algorithm, that uses ensembles to create one decision tree at a time, we had an accuracy of 0.9517 and the other performance metrics can be seen in Table VIII.

Finally, the C5.0 Algorithm, a recent development, had an accuracy of 0.9172 and the other performance metrics can be seen in Table IX.

TABLE X
Overall Accuracies

Model Names	Accuracies
Decision Trees	0.9103
RPART	0.9103
C4.5(J48)	0.9103
PART	0.8966
Bootstrapped Aggregation	0.5034
Random Forests	0.9517
C5.0	0.9172

TABLE XI
Confusion Matrix

Class Value	Very Low	Low	Middle	High	Class Error
High	0	0	1	62	0.01587302
Middle	0	6	81	1	0.07954545
Low	0	79	4	0	0.04819277
Very Low	18	6	0	0	0.25000000

TABLE XII
Feature Importance Matrix

Feat	Very Low	Low	Middle	High	Mean Decr. Accur.	Mean Decr. Gini
STG	0.0119	0.0328	0.0107	0.0178	0.0196	14.5602
SCG	0.0427	0.0031	-0.0010	-0.0032	0.0033	12.8192
STR	-0.0093	-0.001	0.0017	-0.0023	-0.0013	10.4610
LPR	0.0812	0.1109	0.1592	0.1057	0.1236	31.8488
PEG	0.4812	0.4758	0.4312	0.549	0.4757	113.1254

The overall comparison on Accuracy can be seen in Table X.

The best algorithm regarding Accuracy is Random Forests, followed by C5.0, Decision Trees, RPART, C4.5, PART and Bagging at the end. Random Forests ran with 500 trees and 2 variables tried at each split.

The OOB (Out Of Bag) estimate of error rate is 6.98%, as this technique involves sampling the input data with replacement (bootstrap sampling). In this sampling, about one-third of the data is not used for training and can be used to testing. Error estimated on these out of bag samples is the out of bag error.

The confusion matrix in Table XI shows clearly how the algorithm predicted the correct class for each class value. For ”Very Low” had 6 errors and 18 correct predictions, reflected on the Class Error of .25 or 25%, ”Low” had 4.8% error, ”Middle” had 8% error and for High 1.6% error.

We can also extract the feature importance matrix, in figure XII with metrics on the importance of each feature related to each of the prediction class values. As we expected from the initial analysis of the features, some of them carry little value and even get in the way of a good prediction, deducing from their negative importance values.

These metrics confirm our initial analysis of the data, and there are features that do not contribute and even reduces its accuracy. In the figures 9 and 10 we have feature importance plotted to clearly show that PEG is the most important feature, second, it is followed closely

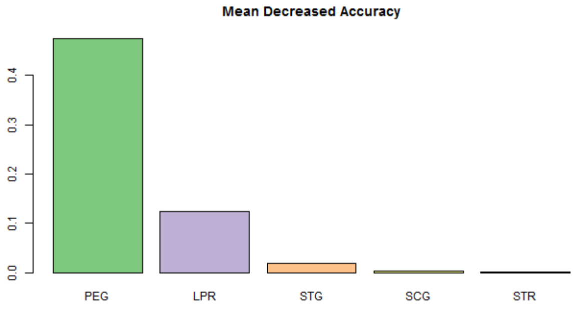

Fig. 9. Mean Decreased Accuracy Barplot

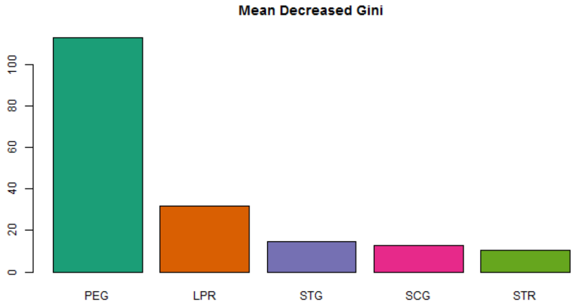

Fig. 10. Mean Decreased Gini Barplot

TABLE XIII
Random Forests

Class Value	Precision	Recall	F1
High	1.0000000	1.0000	1.0000
Middle	0.9677419	0.8824	0.9231
Low	0.8823529	0.9783	0.9278
Very Low	1.0000000	0.9231	0.9600

by LPR. STG, SCG, and STR have very little importance to the model, and some negative figures indicate that they may reduce the prediction accuracy. These features should have their data acquisition reviewed in order to gain more predictability.

So we ran the best model one more time, but with fewer features. The next formula for prediction still defines UNS as the dependent variable or prediction class and LPR and PEG are our independent variables or prediction features. We ran the training and prediction of Random Forests for this new formula. The model accuracy of 0.9517 was the same of the previous run. However, the other performance metrics can be seen in Table XIII are slightly different.

TABLE XIV
Confusion Matrix

Class Value	Very Low	Low	Middle	High	Class Error
High	0	0	2	61	0.0317
Middle	0	4	83	1	0.0568
Low	2	80	1	0	0.0361
Very Low	22	2	0	0	0.0833

TABLE XV
Feature Confusion Matrix

Feat	Very Low	Low	Middl	High	Mean Decr Acc	Mean Decr Gini
PEG	0.4914	0.4757	0.4243	0.5259	0.4680	111.5
LPR	0.1336	0.1666	0.1842	0.1356	0.1617	41.6
STG	0.0202	0.047	0.0153	0.038	0.0316	27.9

This time the OOB estimate of error rate is 4.65%, much lower than the previous run with all features. The confusion matrix in Table XIV shows a great improvement in comparison with the previous run. For ”Very Low” had 2 errors and 22 correct predictions, reflected on the Class Error of .83 or 8.3%, previously we had 25% error for this value, ”Low” achieved 3.6% and previously had 4.8% error, ”Middle” got 5.7% and before had 8% error and finally ”High” lowered from 1.6% error to 3.1%, previously the model had 62 correct predictions and this time we had 61. This is the price we paid in order to raise the overall prediction performance.

At last, the feature importance matrix, in figure XV shows the new relationship of the variables. The figures show no negative values and they are significantly consistent.

V. Conclusion and Future Work

The accuracy of Random Forests was better, as expected. Once again we restate our goal to use machine learning methods to show the strengths and weakness of the underlying business model, in this case a knowledge model. In addition we provided a way to use the data in a efficient way to achieve predictability. With this in mind we believe we achieved the desired results.

Now we know that some features are weaker than others and need to be removed from the prediction model in order to achieve greater accuracy. These features need to be re-engineered, but on the business model, to gather data more consistent with the desired results.

We also produced a prediction model able to determine the correct class value for a set of independent variables. This model can be used in various knowledge systems that need to predict the student future performance based on historical information.

We think that future experiments could repeat this investigation against different knowledge models and data sets, with other machine learning methods, such as neural networks.

We used R language to execute the analysis and prediction, including the graphics and data included in this paper, all code is on [21] GitHub.

Acknowledgment

This research was supported by Instituto INFNET. We would like to show our gratitude to professor Cassius Figueiredo for sharing his knowledge on R and algorithms, that we used throughout this research.

We also thank our course coordinator Eduardo Morelli who provided support and guidance that greatly assisted the research.

References

[1] H. T. Kahraman, “Designing and application of web-based adaptive intelligent education system,” *PhD. Thesis*, pp. 1–156, 2009.

[2] H. T. Kahraman, S. Sagiroglu, and I. Colak, “The development of intuitive knowledge classifier and the modeling of domain dependent data,” *Knowledge-Based Systems*, vol. 37, pp. 283–295, 2013.

[3] H. T. Kahraman, S. Sagiroglu, and İ. Colak, “A novel model for web-based adaptive educational hypermedia systems: Sahm (supervised adaptive hypermedia model),” *Computer Applications in Engineering Education*, vol. 21, no. 1, pp. 60–74, 2013.

[4] I. Colak, S. Sagiroglu, and H. T. Kahraman, “A user modeling approach to web based adaptive educational hypermedia systems,” in *Machine Learning and Applications, 2008. ICMLA’08. Seventh International Conference on.* IEEE, 2008, pp. 694–699.

[5] L. Breiman, J. Friedman, R. Olshen, and C. Stone, “Classification and regression trees, wadsworth international group, belmont, california, usa, 1984; bp roe et al., boosted decision trees as an alternative to artificial neural networks for particle identificatio n,” *Nucl. Instrum. Meth. A*, vol. 543, p. 57, 2005.

[6] C. Lazăr and M. Lazăr, “Using the method of decision trees in the forecasting activity.” *Petroleum-Gas University of Ploiesti Bulletin, Technical Series*, vol. 67, no. 1, 2015.

[7] F. C. Arnett, S. M. Edworthy, D. A. Bloch, D. J. McShane, J. F. Fries, N. S. Cooper, L. A. Healey, S. R. Kaplan, M. H. Liang, H. S. Luthra *et al.*, “The american rheumatism association 1987 revised criteria for the classification of rheumatoid arthritis,” *Arthritis & Rheumatism*, vol. 31, no. 3, pp. 315–324, 1988.

[8] M. M. Ward, D. C. Elli, L. M. Jack *et al.*, “Differential rates of relapse in subgroups of male and female smokers,” *Journal of Clinical Epidemiology*, vol. 46, no. 9, pp. 1041–1053, 1993.

[9] R. Díaz-Uriarte and S. A. De Andres, “Gene selection and classification of microarray data using random forest,” *BMC bioinformatics*, vol. 7, no. 1, p. 1, 2006.

[10] A. Bosch, A. Zisserman, and X. Munoz, “Image classification using random forests and ferns,” in *2007 IEEE 11th International Conference on Computer Vision.* IEEE, 2007, pp. 1–8.

[11] J. H. Friedman, “Greedy function approximation: a gradient boosting machine,” *Annals of statistics*, pp. 1189–1232, 2001.

[12] J. Leathwick, J. Elith, M. Francis, T. Hastie, and P. Taylor, “Variation in demersal fish species richness in the oceans surrounding new zealand: an analysis using boosted regression trees,” *Marine Ecology Progress Series*, vol. 321, pp. 267–281, 2006.

[13] R. Lawrence, A. Bunn, S. Powell, and M. Zambon, “Classification of remotely sensed imagery using stochastic gradient boosting as a refinement of classification tree analysis,” *Remote sensing of environment*, vol. 90, no. 3, pp. 331–336, 2004.

[14] L. Breiman, “Random forests,” *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[15] 2016. [Online]. Available: https://www.kaggle.com/

[16] M. A. Hernández and S. J. Stolfo, “Real-world data is dirty: Data cleansing and the merge/purge problem,” *Data mining and knowledge discovery*, vol. 2, no. 1, pp. 9–37, 1998.

[17] P. Royston *et al.*, “Multiple imputation of missing values,” *Stata journal*, vol. 4, no. 3, pp. 227–41, 2004.

[18] D. M. Hawkins, “The problem of overfitting,” *Journal of chemical information and computer sciences*, vol. 44, no. 1, pp. 1–12, 2004.

[19] 2013. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/User+Knowledge+Modeling

[20] V. Tsiriga and M. Virvou, “A framework for the initialization of student models in web-based intelligent tutoring systems,” *User Modeling and User-Adapted Interaction*, vol. 14, no. 4, pp. 289–316, 2004.

[21] 2017. [Online]. Available: https://github.com/omnibug/infnet$_{m}oduloA$