Engineering Educators Bringing the World Together
2025 ASEE Annual Conference & Exposition
Palais des congrès de Montréal, Montréal, QC • June 22–25, 2025 ASEE
Paper ID #49246

Study on the Use of Random Forest Classifier model and Multi Output Classifier model for Predicting Student Academic Performance and Identifying Area of Concern

Mr. Kevin Huang, Troy High School

Kevin Huang is a student at Troy High School in Fullerton CA. As a participant of the Troy Tech program, he has worked as a student intern at several professor's research labs in the College of Engineering and Computer Science at California State University Fullerton to study the use of machine learning for predicting student academic performance and how machine learning may be used to identify at-risk students for an early warning and possible intervention.

Ivan Zimmerman
Dr. Doina Bein, California State University, Fullerton

Dr. Bein has an extensive publication record: 13 book chapters, 19 journal articles, and 69 conference papers. Four of her conference papers have received the best paper awards. She was awarded (as PI or co-PI) several research and teaching grants from AF

Study on the Use of Random Forest Classifier model and Multi-Output Classifier model for Predicting Student Academic Performance and Identifying Areas of Concern

Kevin Huang[1], Ivan Zimmerman[1], Doina Bein[2]
[1] *Troy High School, Fullerton, CA, USA*
[2] *Computer Science Department, California State University, Fullerton, CA, USA*

Abstract

This paper explores the use of machine learning to identify key factors that may connect to a student's academic performance and how it may be used to predict student learning outcome at an early stage, specifically, by utilizing two machine learning models: the Random Forest classifier and the Multi-Output classifier. The Random Forest Classifier is widely used for classification tasks. It operates by constructing multiple decision trees during training and selecting the mode of their predictions for a given input, identifying the most significant factors affecting outcomes. On the other hand, a Multi-Output Classifier is specifically designed for multi-label or multi-output classification tasks, where each instance can be linked to multiple labels or output variables. It may be used for predicting several target variables simultaneously, for example, assessing a student's grade and engagement level simultaneously, and our Multi-Output classifier uses a neural network backend. In this paper, several datasets sourced from Kaggle containing student background information and academic engagement and performance data were processed using the above two classifier models. The steps for cleaning, preparing and analyzing the data were discussed in this paper. The results show that the Random Forest classifier is very effective in identifying key factors that may connect to a student's academic engagement and performance such as number of units completed in previous semester, grades from previous semester, and tuition fee payment status with an accuracy of 85.9% for the predictions on the test data: 94.5% correctly on the prediction of non-dropouts and 67.9% correctly on the prediction of dropouts. Furthermore, the same set of data was processed by the Multi-Output Classifier neural network resulting in accuracy scores ranging from 83.5% to 94.2% for the five target variables, providing valuable insights to educators for advocating tailored support for at-risk students.

Introduction

Multiple research projects have shown the effectiveness of using early warning for supporting at-risk students in engineering education as it may help tailor support, enhance learning outcome and improve the retention rates of those students who might otherwise drop out (Atif et al, 2020; Tinto, 2012).

Early warning systems usually relied on the analysis of student performance data and engagement indicators to predict which students may need additional support (Akçapınar et al,

2019; Shafiq et al, 2022). Studies also show that the use of predictive analytics, a key component of early warning systems, is effective to target and improve first year student attrition (Seidel & Kutieleh, 2017) and it works for online STEM learning as well (Yu & Wu, 2021).

In recent years, with the growing popularity of machine learning techniques, a variety of machine learning algorithms have been utilized to analyze student data and predict student learning outcomes to identify at-risk students. The uses of different machine learning techniques including decision trees, support vector machines, classification, regression, and neural networks for predicting student performance were discussed in multiple recent studies (Al-Tabarwi et al., 2019; Dabhade et al., 2021; Gull et al., 2020).

Furthermore, in-depth research has been done into the relative performance of traditional single-output predictive mathematical models in student performance prediction for an engineering course (Huang & Fang, 2013). They concluded that while multiple linear regression models can accurately predict the average exam results of the class, the most accurate model for predicting individual outcomes is the support vector machine model. They also emphasized the importance of expanding inputs beyond static measures of previous student performance – like previous exam scores and GPAs – into data that give a more holistic view of the student, like mindset questionnaires and personal/demographic information. Howard, Meehan, and Parnell conducted a similar study in 2018 comparing the prediction performance of Bayesian Additive Regressive Trees (BART), Random Forests (RF), Principal Components Regression (PCR), Multivariate Adaptive Regression Splines (Splines), K-Nearest Neighbours (KNN), Neural Networks (NN), and Support Vector Machines (SVM). They concluded that the most promising model was BART: combined with clustering and other methods, this model has an average prediction error of 6.5 prediction points six weeks into the semester (Howard et al., 2018). In another study, the effectiveness of the Majority Voting model, in which submodels vote in order to determine the final classification of a student, was established (Said et al., 2024). This approach greatly increases accuracy, resulting in an accuracy of 92.7%.

Among the different machine learning methods, classification is considered one of the most essential tasks for analyzing student performance data because it enables the identification of patterns in student performance and behavior, allowing educators to categorize students into distinct groups based on factors such as grades, background, and engagement, helping in the early identification of at-risk students, enabling timely interventions that improve academic performance and reduce dropout rates. Additionally, classification allows for personalized learning strategies, better resource allocation, and the development of predictive models that forecast future performance to make data-driven decisions and optimize teaching and learning strategies.

In this paper, the use of two classification methods: the Random Forest Classifier (RFC) model and the Multi-Output Classifier (MOC) model is discussed as they show very promising results

in analyzing student performance data for providing reference to early warning and intervention in engineering education.

The Random Forest Classifier is a widely used machine learning algorithm for classification tasks that operate by constructing multiple decision trees during training and selecting the mode of their predictions for a given input. This ensemble method enhances accuracy and helps control overfitting while ranking feature importance, which aids in identifying the most significant factors affecting outcomes (Parmar et al., 2018; Breiman, 2001; Prasad et al., 2002). Due to its minimal requirement on data preprocessing and the straightforwardness for implementation (Scikit-Learn, 2024), the Random Forest algorithms have been established as a strong contender in student performance prediction. Huynh-Cam, Chen, and Le (2021) achieved an 80.00% accuracy rate using only demographic information, allowing strong predictions even before the beginning of the learning period and leading to greater applicability in practice. Another benefit of this class of model is the explainability of the generated rule sets. When making predictions that will directly affect students, one must be able to explain why they were targeted for additional learning resources or for possible remedial work. Explainability becomes more crucial when model predictions are expanded beyond grades.

On the other hand, the Multi-Output or Multi-Label Classifier is a machine learning model specifically designed for multi-label or multi-output classification tasks, where each instance can be linked to multiple labels or output variables. It is a machine learning model that is used for multi-output classification problems, where the goal is to predict multiple target variables simultaneously. Each target variable may belong to a different class, and the model makes predictions for all these target variables at once. This is particularly useful in scenarios where there are multiple related tasks that need to be solved at the same time, such as predicting multiple attributes or labels for each instance (Zhang et al., 2014; Xu et al., 2020; Xue et al., 2023). For educational settings, the Multi-Output Classifier may be very beneficial in cases where each sample may fit into more than one category, such as assessing a student's grade and engagement level simultaneously, thus improving educational strategies.

While the Random Forest Classifier has been studied extensively in predicting student academic performance, there are few efforts in applying the Multi-Output Classifier to predict student learning outcomes. A recent study tried to use multiple output models utilizing learning data from the first six weeks of a course platform to predict students' homework, experiment, midterm, and final scores six weeks later, achieving good results (Xue et al., 2023).

Built on the above prior studies, this paper investigates some of the implementation issues for utilizing those two classifiers in analyzing student data aiming to provide some comparison and improvement for the utilization of those two classifier methods in educational settings.

Methodology

In this paper, datasets sourced from Kaggle containing student background information and academic engagement/performance data for college freshman students were processed using the two classifier models. Our analysis is based on a comprehensive dataset of 4,424 student records. The data used in this study includes information such as parent occupation, student financial status, scholarship status, student qualification upon admission, student employment status, tuition and fee information, morning or evening attendance, number of units enrolled and credited in the first semester, student grades in first semester, number of units enrolled and credited in the second semester, and student grades in the second semester. Although the initial dataset contained second semester grades, they were excluded from both model's inputs to avoid bias by ensuring predictions were based on only pre-semester factors alone. In this study, a 'dropout' refers to any student who left their studies at any stage, rather than those only at the end of the second semester. 'Debtor' and 'Educational special needs' are binary indicators, indicating whether a student has unpaid fees or has been classified as needing extra educational support, respectively. Additionally, 'Curricular units 1st sem (evaluations)' refers to qualitative assessments such as conduct and participation, while 'Curricular units 1t sem (grade)' represent academic performance.

The following are the general steps for implementing the Random Forest and Multi-Output classifier methods (Scikit-Learn, 2024) with our adaptations.

Random Forest Classifier

- Data Cleaning and Preparation

  The first step was preparing the dataset, focusing on selecting the most relevant features for predicting student performance: whether students dropped out. For this step, we needed to ensure that the data used to train the model accurately reflected patterns of dropout without being too closely tied to specific details that could skew the results. For example, columns related to second-semester grades and curricular unit data were removed because they directly indicated whether a student had dropped out or not.

- Data Preprocessing

  In this step, data were split into two categories: vector X_2 contains all the predictor variables (the features), while vector y_2 holds the target variable, which is the dropout status (whether a student dropped out or not). By removing second semester student performance data, we ensured that the model wouldn't be biased by data that was too closely linked to the target variable. Instead, the remaining features were intended to help predict dropout by identifying patterns that were more generalized across the entire academic experience. This approach helped ensure that the model would be more versatile and capable of predicting dropout across different scenarios.

- Splitting the Dataset

  Once the data was cleaned and irrelevant columns were removed, the next step was splitting the dataset into two sets: training and testing. Seventy percent of the data was used to train the model and the 30% left was reserved for testing model performance.

  As a result, we split the data into two subsets:

  a) Training Set: This portion of the data is used to train the model, allowing it to learn the underlying patterns between the features (X_2) and the target variable (y_2), which in our case is student dropout status.
  b) Testing Set: After the model has been trained, it is tested on this portion of the data to evaluate how well it can predict outcomes for unseen instances. This simulates the real-world application of the model, where it will need to predict the dropout status of students who were not part of the training process.

- Model Selection and Hyperparameter Tuning

  After splitting the dataset, the next step was model selection and hyperparameter tuning to ensure the model learns effectively from the training data and is ready to perform well on the test data. For a Random Forest Classifier, there are several key hyperparameters that shape how the decision trees in the forest grow and how they make predictions. In our model, we focused on four important hyperparameters: n_estimators (number of trees in the forest), max_depth (how deep each tree can grow), min_samples_split (minimum number of data points required to split a node), and min_samples_leaf (minimum number of samples that a leaf node or the final node in a tree should have).

- Model Training and Optimization

  To find the best combination of those hyperparameters, we used Grid Search, which is where we test all possible combinations of hyperparameters to see which one gives us the best results. It's extremely useful, but can be computationally expensive, especially when there are a lot of combinations to check. But when done right, it's worth the time because it helps us find the most effective setup for the model. Furthermore, this Grid Search function we used doesn't just test all the combinations but also uses cross-validation, which means it splits the data into different parts and trains the model multiple times to get a better sense of how well it performs. As a result, this not only tells us how well the model performed on the training data but also makes sure it wasn't just memorizing the data but was actually learning patterns that could be applied to new data.

- Prediction and Model Evaluation on Test Data

  Once we obtain the best parameters from model training and optimization, we can then use them to evaluate the model on the test data to make predictions. This is where we see how well the model works on data it has never seen before, which is a good test of how the model will perform in real-world situations. For this step, a test accuracy score is calculated, which gives us an idea of how accurate our model is in predicting whether a

student will drop out. A higher test accuracy means the model is doing a good job at generalizing the data and making accurate predictions.

Multi-Output Classifier (Neural Network)

The steps for implementing the Multi-Output Classifier are very similar to those for a single-output classifier like the ones shown above for the Random Forest Classifier but with the addition of handling multiple target variables.

To achieve this goal, in our case, the 'Target' object column is split into three binary columns: Target-Graduate, Target-Dropout, and Target-Enrolled. We use one-hot encoding, in which only one of the three columns can be true for a given sample. After that, we identify the five columns that will be predicted. These include the above columns as well as 'Debtor' and 'Educational special needs'. Then, a scan was done on the data to remove invalid and outlier ones. Accordingly, the dataset is split into train and test subsets with twenty percent of the data allocated to the test set, and eighty percent into the training set. Next, the data is corrected for class imbalance by duplicating samples that have a 1 in the 'Debtor' or 'Educational special needs' columns. Specifically, the 'Debtor' samples are weighted 4 times more than regular samples, while 'Educational special needs' samples are weighted 40 times more. The final effect of this correction is that 3618 synthetic samples are added to the training set, increasing the data size by 89.01%. This resulted in a ~30% and ~20% increase in positive sample identification for the 'Debtor' and 'Educational special needs' classes respectively. Finally, we use sklearn's StandardScaler utility to normalize the mean and standard deviation of each column to 0 and 1, respectively.

With these steps of data preparation done, we may then implement the Multi-Output Classifier model. Specifically, our Multi-Output Classifier (MOC) is composed of three dense neuron layers: the first, which uses the ReLU activation function, has 64 neurons; the second, which also uses the ReLU activation function, has 32 neurons; and the final layer, which uses the Sigmoid activation function, has 5 neurons. The ReLU (Rectified Linear Unit) activation function is the standard choice for dense layers because of its effectiveness and simplicity. It can be expressed mathematically as ReLU(x) = max(0, x). The Sigmoid activation function is used in order to map the values to zero or one for classification. It can be expressed mathematically as

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{1}$$

It is chosen over the similar Softmax function because unlike the Softmax function, the Sigmoid function does not assume single-class output. Additional Dropout layers are inserted between the dense layers with a dropout probability of 0.2 in order to reduce dependence on a single input variable and aid generalization. It uses the tensorflow library's implementation of the Adam

optimization algorithm, which is “computationally efficient and … well suited for problems that are large in terms of data and/or parameters.” (Kingma et al, 2017) The Adam algorithm is especially effective here because of its built-in weight decay, which mitigates the negative effect of including many low-correlation features. This weight decay works by incorporating the number of active weights into the loss function as shown below, incentivizing the model to “zero out” inputs that do not substantively affect the result.

$$\text{Loss} = \frac{1}{N}\sum_{i=1}^{N} -\left(y_i \cdot \log(\hat{y}_i) + (1 - y_i) \cdot \log(1 - \hat{y}_i)\right) \tag{2}$$

Results

Correlation Heatmap analysis

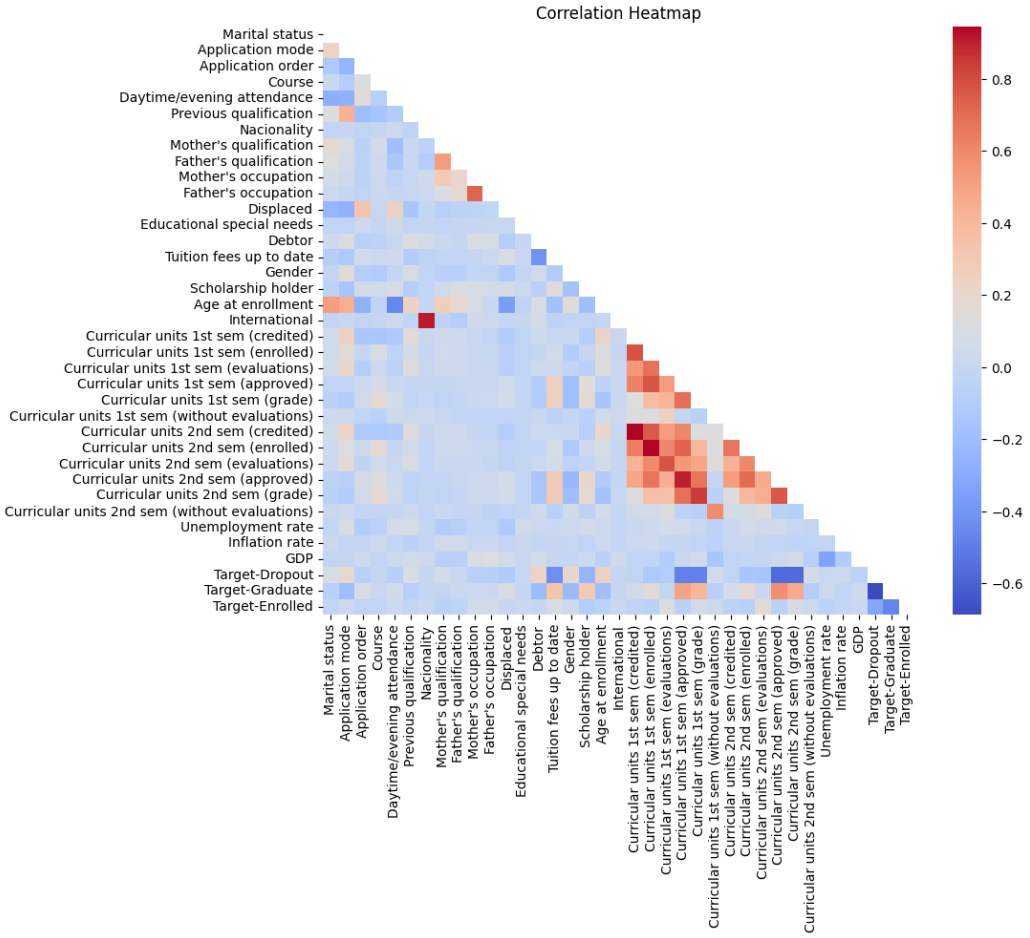

Figure 1: Heatmap Visualizing Data Correlations

Before implementing the two classifier models, we used the Seaborn library to generate a heatmap visualizing the correlations between different variables in the dataset as it would provide valuable insights on the validity of the dataset. As Figure 1 shows, the most immediately apparent feature (color coded in red) are the two squares at the intersections of the "Curricular units" feature families. This illustrates that students who score well on the exams in the first semester are likely to keep up their performance during the second semester. For another example, the heatmap also shows us that the "Tuition fees up to date" column will be instrumental in predicting which students are in debt.

Data Imbalance Correction

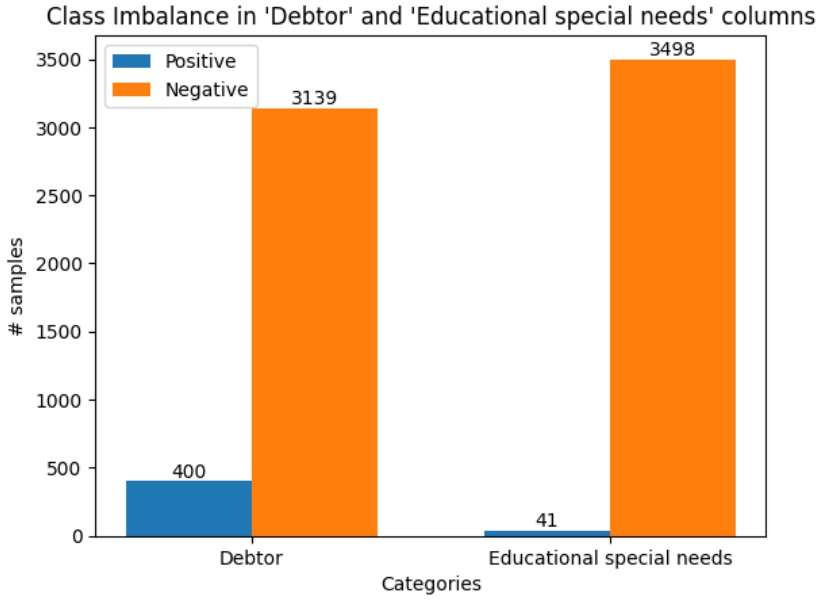

Figure 2: Illustration of Data Imbalance

In a typical test dataset, the 'Debtor' and 'Educational special needs' columns are severely imbalanced. In particular, students with educational special needs make up only 1.2% of the dataset (Figure 2). The lack of data here ultimately makes it very hard to predict if a student has special needs or not across the large number of input features. However, we can take steps to mitigate the effect of this lack by increasing the weight of the imbalanced classes. With only 41 instances in ESN, multiplying them by ~100x to equalize the columns would skew the other predictions towards whichever features happened to be overrepresented in the ESN samples. Instead, we manually choose constants to multiply the number of samples to a reasonable proportion of the total, keeping in mind that a false positive inadvertently notifying an instructor to a student possibly having educational special needs should be strongly disincentivized. The 'Debtor' samples weigh 4 times as much, while the 'Educational special needs' samples weigh 40 times more than a normal sample.

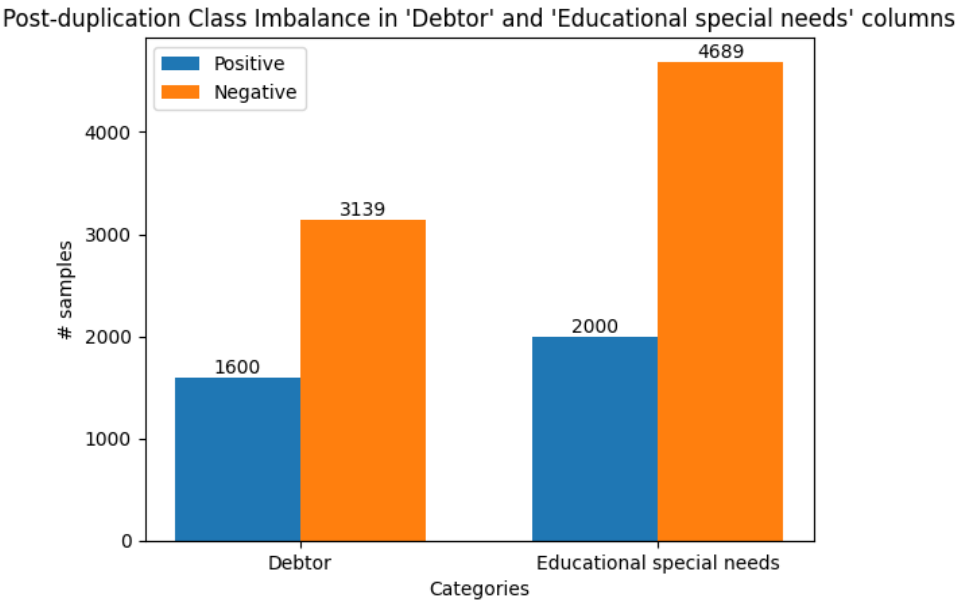

Figure 3 : Illustration of Data Imbalance Correction

After correcting the imbalance, as shown in Figure 3, the final class makeup has a reasonable distribution. Note that the rare samples that belong to both minority classes are duplicated in both rounds, increasing the final number of samples in 'Educational special needs'.

Hyperparameter Optimization and Model Configuration

To make sure our model works well, we focused on fine-tuning a few important settings, known as hyperparameters, for the Random Forest classifier. These settings help the model strike the right balance between being too simple or too complex.

- max_depth: 10
  By limiting how deep the trees in the Random Forest can grow, we prevent them from getting too complicated and fitting the training data too closely. A depth of 10 ensures the trees are still able to make meaningful decisions, but they won't overfit.
- min_samples_leaf: 4
  This setting makes sure that each decision node in the trees has enough data to work with before making a prediction. With at least four samples in each leaf node, the model avoids making predictions based on small or unreliable data groups.
- min_samples_split: 10
  This parameter ensures that the model only splits nodes when it has enough data. A minimum of 10 samples before splitting keeps the tree from getting too specific and overfitting to small details that aren't likely to hold up in new data.
- n_estimators: 200
  The model uses 200 trees in the Random Forest. Having a larger number of trees helps the model make more accurate predictions because the predictions of all the trees are averaged together, which smooths out any individual errors.

The hyperparameters were chosen with the goal of making the model both powerful and flexible, ensuring it could predict dropout rates effectively without becoming too specialized for just the training data.

Accordingly, we used 5-fold cross-validation to check how well our model could perform on different sets of data. This method splits the data into five parts and tests the model on one part at a time, which helps make sure the results as shown in table 1 are reliable.

Cross-Validation Performance and Test Accuracy

	precision	recall	f1-score	support
0.0	86.02%	94.54%	0.9008	898.0
1.0	85.63%	67.91%	0.7575	430.0
accuracy	85.92%	85.92%	0.8592	0.8592
macro avg	85.82%	81.23%	0.8291	1328.0
weighted avg	85.89%	85.92%	0.8544	1328.0

Table 1. sklearn.metrics.classification_report output for Random Forest classifier

- Test set accuracy (85.9%)
  When we tested the model on the data it hadn't seen before, it performed similarly with an accuracy of 85.9%. This score suggests that the model isn't just memorizing the training data but learning general patterns that work well on new data. This consistency indicates that the model is likely to be reliable in real-world scenarios, not just on the data it was trained on.

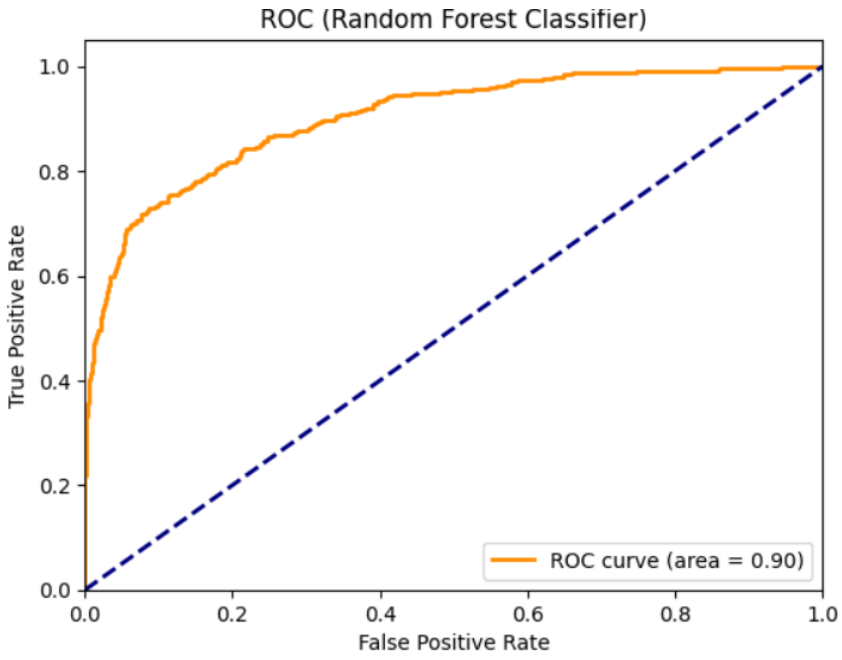

Figure 4: ROC Curve illustrating True Positive Rate vs False Positive Rate

The ROC curve displayed in Figure 4 shows the Random Forest Classifier's performance, achieving an area under the curve (AUROC) of 0.90. This high AUROC value indicates that the model successfully distinguishes between dropout and non-dropout students with a high degree of accuracy. The steep rise in the curve reflects a high true positive rate with minimal false positives, showing that the model effectively identifies true dropouts while minimizing false alarms. Such performance validates the model's ability for educational dropout prediction, highlighting its effectiveness in early identification of students who are at risk of dropping out.

These results confirm that the Random Forest model is strong at predicting dropout rates and is generalizing well, meaning it's not overfitting or underfitting.

Comparison of Actual vs. Predicted Dropout Status

We then compared the model's predictions with the actual dropout status of students to see which situations the model correctly assessed. This helps us understand where the model can improve.

- Recall Analysis
  The recall values for the 'Target-Dropout' classification overall reflect the practical impact of the model. With a recall of 94.5% for students labeled as 0.0 (none-dropouts), the model consistently identifies those who are likely to remain enrolled, ensuring minimal misclassification of these students. For students labeled as 1.0 (dropouts), the recall of 67.9% highlights the model's capability to correctly predict 67.9% of actual dropouts. While there is room for improvement in identifying more actual dropouts, this performance is meaningful within the context of early intervention efforts. By effectively capturing most dropout cases, the model supports targeted decision-making and enables the ability to take cautionary measures.

Multi-Output Classifier (Neural Network) Results

	Accuracy	Positive Recall	Negative Recall
Target-Graduate	85.65%	86.96%	84.38%
Target-Dropout	86.44%	73.72%	92.74%
Target-Enrolled	83.50%	18.06%	97.40%
Debtor	85.88%	54.37%	90.03%
Edu Special Needs	94.24%	20.00%	95.09%

Table 2. Results from Multi-Output Classifier

The final accuracy scores for the five output variables 'Target-Graduate', 'Target-Dropout', 'Target-Enrolled', 'Debtor', and 'ESN' (Educational special needs), from running the Multi-Output model, were 85.65%, 86.44%, 83.50%, 85.88%, and 94.24%, respectively.

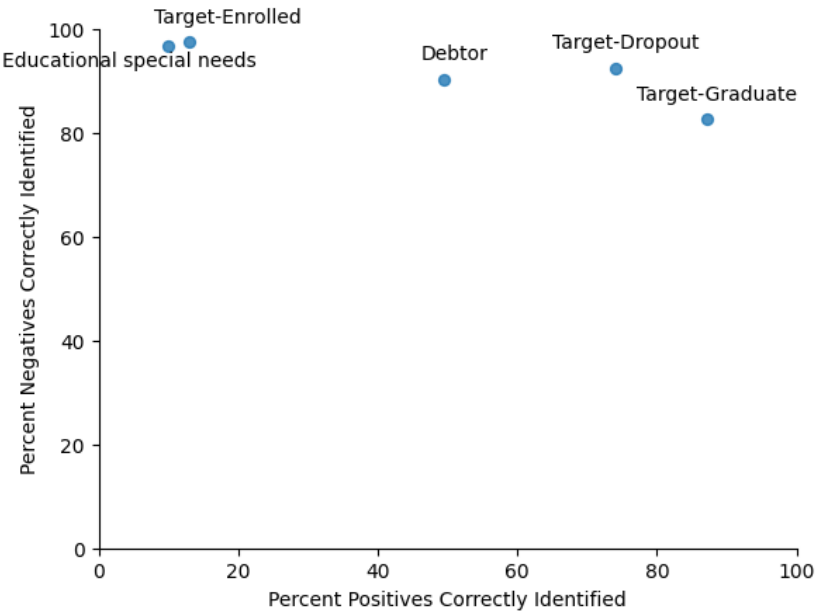

Figure 5: Illustration of True Negatives Identified

From table 2 and Figure 5, we can see that all five columns have very high accuracy for identifying negative variables but vary in how well they can identify positive samples based on class imbalance. Because of this class imbalance and lack of samples, the positive recall scores were low: 86.96%, 73.72%, 18.06%, 54.37%, and 20.00%, respectively. Note that the positive recall of 'Education special needs' is exactly twenty percent because the test dataset only contained ten positive samples. More careful data collection could solve the positive recall issue for ESN student prediction. Target-Enrolled is deemphasized in accuracy rating because of its similarity to Target-Dropout. Finally, the recall of negative predictions was high, an important metric given the impact of making incorrect positive assumptions about a student's financial or ESN situation. Because of the high negative recall, we can feel relatively confident about the accuracy of this model's prediction when it predicts that a student belongs to an additional-resource group.

Feature Importance and Insights

One of the advantages of using a Random Forest classifier is that it helps us understand which factors are most important in predicting dropout. By looking at which features have the most influence, we can gain insight into what causes students to drop out.

The top features that the model found connecting to student dropout include:

- Curricular units 1st sem (approved) (importance: 0.2538)
  Whether students completed their first semester successfully is a big indicator of whether they're at risk of dropping out. Those who fail or struggle early in their studies are more likely to disengage and drop out later.

- Curricular units 1st sem (grade) (importance: 0.1692)
  The grades students earn in their first semester are another strong predictor. Low grades suggest that students might be struggling academically, which can lead to frustration and dropout.
- Tuition fees up to date (importance: 0.1467)
  Financial issues also play a big role in dropout risk. Students who fall behind on tuition payments may be forced to leave due to financial constraints, making this feature a key predictor.
- Age at enrollment (importance: 0.0608)
  The age at which a student enrolls can also affect dropout risk. Older students might have more challenges, like balancing school with work or family obligations, which could impact their ability to stay enrolled.
- Curricular units 1st sem (evaluations) (importance: 0.0526)
  Performance on evaluations in the first semester shows how well students are managing the course content, and those struggling here might be at a higher risk of dropping out.

These more relevant features align with known factors that influence student retention and dropout, confirming that the model is making decisions based on meaningful and relevant data.

Although the model performed well, there are still ways to improve its accuracy and reliability. Some of these improvements could include:

- Reducing False Positives:
  The model sometimes incorrectly predicted that students would drop out when they didn't. By refining the model to reduce these false positives, we could avoid unnecessary interventions or misleading predictions.
- Adding More Features:
  Including more data, such as attendance patterns, social engagement, or mental health indicators, could help the model make even better predictions. These factors might reveal deeper insights into why students drop out.

Conclusion

Within our study, the Random Forest Classifier (RFC) has proven its effectiveness for predicting single-output outcomes in educational data. Its learning approach, which builds multiple decision trees and aggregates their predictions, allows it to handle diverse features, ranging from numerical performance metrics to also categorical factors. This method is particularly effective in avoiding overfitting, making it an ideal choice for large, complex datasets typical in educational contexts.

RFC's ability to rank features by importance also adds an additional layer of insight, as it can help educators identify the key factors connecting to student success or failure. For example, in our analysis, RFC was able to determine certain aspects that were key predictors of dropout risks. With this information, institutions could then create early warning systems for at-risk students, helping provide early interventions such as tutoring or mentorship. By focusing on the

factors that matter most, RFC enables more efficient resource allocation and allows people to take early measures to help students stay on track.

However, the limitation of RFC is that it often assesses each outcome in isolation. Educational success is connected to a multitude of factors, including emotional, behavioral, and social factors as well as academic performance. RFC overall excels at predicting specific outcomes, such as whether a student will drop out or pass a course. However, it has difficulty handling related outcomes simultaneously, which is where the Multi-Output Classifier (MOC) offers an advantage.

In addition to predicting a single outcome, for example whether a student would drop out, the Multi-Output classifier (MOC) can make additional predictions: whether the student is in debt and whether the student has educational special needs (ESN) in our case. This could allow educators to offer resources that are specific to these disadvantaged groups to those who need them most. For instance, a student whose actions suggest they are struggling with debt could be offered an on-campus work opportunity or receive information about scholarships. Similarly, a student who exhibits behavior that indicates they may have educational special needs could receive resources designed to help specifically ESN students. Crucially, students should not be confronted with the model's predictions but rather approached casually about additional opportunities, such as through an automated email like those ones students already receive.

While both RFC and MOC offer significant insights into assessing student performance, several challenges remain. One of the primary obstacles is the quality of the data used for training models. Missing values, inconsistencies, and biases in the dataset can all impact accuracy and skew the fairness of predictions. It is essential to ensure that the data is both cleaned and preprocessed properly, with an emphasis on eliminating any potential bias that could disproportionately affect underrepresented student groups.

Future Work

For future work, we plan to test the two classifier models on various types of student learning data, for example, the live student engagement and performance data obtained from a learning management system, as it would assist course instructors to better assess student academic needs. Using our own data instead of an online dataset might also help address severe class imbalance issues. Additionally, it may be worthwhile to build hybrid models that combine RFC's feature selection with MOC's multi-output prediction, as it would allow for more precise predictions and a deeper understanding of how different aspects of student engagement and performance are interrelated. Furthermore, it may be worthwhile to incorporate some qualitative data, such as teacher assessments, student feedback, and classroom observations, within the analysis. This could further improve the models' accuracy as current RFC and MOC predominantly focus on quantitative variables, but the inclusion of more qualitative insights could help capture the full spectrum of student experiences.

References

1. Atif, A., Richards, D., Liu, D., & Bilgin, A. A. (2020). Perceived benefits and barriers of a prototype early alert system to detect engagement and support 'at-risk' students: The teacher perspective. *Computers and Education*, *156*, 103954-
2. Tinto, V. (2012). *Leaving college: rethinking the causes and cures of student attrition* (2nd ed., Vol. 37929). University of Chicago Press.
3. Akçapınar, G., Altun, A., & Aşkar, P. (2019). Using learning analytics to develop early-warning system for at-risk students. *International Journal of Educational Technology in Higher Education*, *16*(1), 1–20.
4. Shafiq, D. A., Marjani, M., Habeeb, R. A. A., & Asirvatham, D. (2022). Student Retention Using Educational Data Mining and Predictive Analytics: A Systematic Literature Review. *IEEE Access*
5. Seidel, E., & Kutieleh, S. (2017). Using predictive analytics to target and improve first year student attrition. *The Australian Journal of Education*, *61*(2), 200–218
6. Yu, C.-C., & Wu, Y. (Leon). (2021). Early Warning System for Online STEM Learning—A Slimmer Approach Using Recurrent Neural Networks. *Sustainability*, *13*(22), 1246
7. Al-Tabarwi, H., Ali, U. A. al-J., & al-Ajami, S. Q. (2019). Predicting students' performance using machine learning techniques. *Majallat Jāmi'at Bābil*, *27*(1), 194–205.
8. Dabhade, P., Agarwal, R., Alameen, K. P., Fathima, A. T., Sridharan, R., & Gopakumar, G. (2021). Educational data mining for predicting students' academic performance using machine learning algorithms. *Materials Today: Proceedings*, *47*, 5260–5267.
9. H. Gull, M. Saqib, S. Z. Iqbal and S. Saeed (2020), "Improving Learning Experience of Students by Early Prediction of Student Performance using Machine Learning," *2020 IEEE International Conference for Innovation in Technology (INOCON)*, Bangluru, India, 2020, pp. 1-4
10. Huang, Shaobo, and Ning Fang. "Predicting Student Academic Performance in an Engineering Dynamics Course: A Comparison of Four Types of Predictive Mathematical Models." *Computers and Education*, vol. 61, 2013, pp. 133–45
11. Howard, E., Meehan, M., & Parnell, A. (2018). Contrasting prediction methods for early warning systems at undergraduate level. *The Internet and Higher Education*, *37*, 66–75
12. Ben Said, M., Hadj Kacem, Y., Algarni, A., & Masmoudi, A. (2024). Early prediction of Student academic performance based on Machine Learning algorithms: A case study of bachelor's degree students in KSA. *Education and Information Technologies*, *29*(11), 13247–13270.
13. Parmar, A., Katariya, R., Patel, V., Baig, Z., Hemanth, J., Fernando, X., & Lafata, P. (2019). A Review on Random Forest: An Ensemble Classifier. In *International Conference on Intelligent Data Communication Technologies and Internet of Things (ICICI) 2018* (Vol. 26, pp. 758–763).
14. Breiman, L. (2001). Random forests. *Machine Learning*, 45(1), 5-32.

15. Prasad, A. M., Iverson, L. R., & Liaw, A. (2006). Newer Classification and Regression Tree Techniques: Bagging and Random Forests for Ecological Prediction. *Ecosystems (New York)*, *9*(2), 181–199.
16. Scikit-learn documentation (for Python users): Random Forest Classifier. https://scikit-learn.org/1.5/modules/generated/sklearn.ensemble.RandomForestClassifier.html
17. Huynh-Cam, T.-T., Chen, L.-S., & Le, H. (2021). Using Decision Trees and Random Forest Algorithms to Predict and Determine Factors Contributing to First-Year University Students' Learning Performance. *Algorithms*, *14*(11), 318-.
18. Zhang, M.-L., & Zhou, Z.-H. (2014). A Review on Multi-Label Learning Algorithms. *IEEE Transactions on Knowledge and Data Engineering*, *26*(8), 1819–1837.
19. Xu, D., Shi, Y., Tsang, I. W., Ong, Y.-S., Gong, C., & Shen, X. (2020). Survey on Multi-Output Learning. *IEEE Transaction on Neural Networks and Learning Systems*, *31*(7), 2409–2429.
20. Xue, H., & Niu, Y. (2023). Multi-Output Based Hybrid Integrated Models for Student Performance Prediction. *Applied Sciences*, *13*(9), 5384-.
21. Scikit-learn Documentation on Multi-output Classification: Multiple Output Classification https://scikit-learn.org/1.5/modules/multiclass.html
22. Kingma, D., & Ba, J. (2017). Adam: A Method for Stochastic Optimization. As obtained from https://arxiv.org/abs/1412.6980
23. Kaggle (2024). Education Dataset. https://www.kaggle.com/datasets/thedevastator/higher-education-predictors-of-student-retention