Data Exploration Project

EduPredictors

July 4, 2024

Elisabeth Barar (2825002)

Johanna Unruh (5132960)

Table of Contents

1. Topic and Motivation

Student retention is a significant concern for educational institutions worldwide. High dropout rates not only affect the financial stability of universities but also have long-term implications for students' careers and personal development. Predicting student success and dropout rates can help universities implement targeted interventions, thereby improving retention rates and ensuring better educational outcomes [1]. This project is motivated by the desire to provide data-driven solutions to enhance student support services and optimize academic performance.

This project aims to leverage machine learning techniques to predict student academic success, specifically focusing on whether students are likely to drop out or graduate. By analyzing various factors that influence student outcomes, we can create predictive models that help educational institutions identify at-risk students and take proactive measures to support them.

The goal is to train and evaluate multiple machine learning models to determine the most accurate one for predicting student outcomes. The results will be visualized through a Streamlit app, enabling university administrators and counselors to input student data and receive real-time predictions. This information can then be used to offer personalized support to students at risk of dropping out.

A suitable use case could be a university administration which wants to proactively identify students at risk of dropping out. By predicting which students are likely to drop out, the university can implement targeted interventions to support these students, thereby improving retention rates and overall academic success.

2. Related Work

Several studies have explored the application of data mining and machine learning techniques to predict student dropout and academic success. Here, we review some relevant works in this domain:

Pérez et al. [2] applied data mining techniques to predict student dropout at a Colombian university. They used decision trees, neural networks, and naive Bayes classifiers on a dataset containing academic and socioeconomic factors. Their results showed that the decision tree model achieved the highest accuracy in predicting student dropout.

Naseem et al. [3] proposed an ensemble decision tree model to predict student dropout in computing science programs. They combined multiple decision tree algorithms, including

random forest, gradient boosting, and extreme gradient boosting. Their model achieved an accuracy of 87.5% in predicting student dropout.

Mayahi and Al-Bahri [4] developed a machine learning-based approach to predict student academic success. They employed various algorithms, such as logistic regression, decision trees, random forest, and gradient boosting. Their results indicated that the gradient boosting algorithm outperformed other methods, achieving an accuracy of 92.3% in predicting student success.

Nema and Palwe [5] investigated the use of machine learning algorithms for predicting students' academic success. They compared the performance of algorithms like logistic regression, decision trees, random forest, and support vector machines. Their findings suggested that the random forest algorithm performed best, with an accuracy of 87.2% in predicting academic success.

These studies highlight the potential of machine learning techniques in predicting student dropout and academic success. However, the effectiveness of different algorithms may vary depending on the dataset and the specific problem being addressed.

Our project aims to provide a comprehensive comparison of various supervised machine learning algorithms, including tuning these models, to identify the most suitable approach for predicting student dropout in a higher education context. We included decision trees, k-nearest neighbors (kNN), random forest, and naive Bayes algorithms. Moreover, we implemented a graphical user interface (GUI) to make the prediction models more accessible and user-friendly for educational administrators.

3. Used Technologies and Libraries

In this project, we utilize several Python libraries and technologies to perform data analysis, data visualization, and machine learning tasks. We use Pandas [6] DataFrames to perform data analysis and manipulation. In addition, we use Matplotlib [7] and Seaborn [8] to visualise different results.

Scikit-learn [9] is used to prepress the data as well as train and evaluate the machine learning model for predicting student dropout. Additionally, we use NumPy [10] to transform the data to NumPy arrays which are needed for training the machine learning model using scikit-learn. Moreover, we use the spearman function from SciPy [11] to evaluate which columns have a high correlation to the target variable. Furthermore, we use the library Importnb [12] to import and execute Jupyter Notebooks as Python modules. This enables us to split the project in three different notebooks making the project more structured.

4. Results and Evaluation

We have trained four different models, tuned them, and summarized the results. For the evaluation, we will use the accuruacy score since we want to choose the machine learning model with the best overall performance, since it takes false positives and false negatives into account. The best-performing machine learning models are the following:

1. kNN: tuned model with an accuracy of 88.6%
2. Random Forest: untuned model with an accuracy of 90.4%
3. Decision Tree: tuned model with an accuracy of 88.7%
4. Naive Bayes: untuned model with an accuracy of 86.3%

As can be seen from the results above, all models performed well on the testing data as they all have an accuracy over 80%. The reason for this is that all the models we have chosen are suited to solve classification tasks. However, some models like Random Forest still performed better than others. The reasons for this and why some models performed worse will be explained in the following sections [13].

Out of all the four models, Naive Bayes performed the worst with an accuracy of 86.3%. A possible reason for this is that we used Gaussian Naive Bayes for training. This model assumes that each feature is normally distributed and that the dataset consists of continuous variables. However, this is not entirely the case. As can be seen in the plots in the Jupyter notebook "Evaluation" (see link in Appendix) for the four continuous variables, only the distribution for the feature "Admission grade" can be considered normally distributed. Also, the dataset contains numerically encoded categorical variables next to the continuous variables. These are possible reasons why the model performed worse than the other models [13].

The Decision Tree and the kNN models' performance is almost the same, with accuracies of 88.7% and 88.6%, respectively. Possible reasons for this can be that both can handle categorical (if they are encoded) and numerical variables. Also, the dataset we used is of small/medium size. If the dataset were larger, the performance of the models would differ more significantly. Then we could see which of these two models is better. Another possible reason for why they led to similar results is that all the features used for classification are relevant to the target variable, as the ones that were not needed were removed from the dataset. Both models can effectively capture the patterns and relationships within the data, which also leads to similar performance [13].

Random Forest is the model that performed the best on the dataset with an accuracy of 90.4%. Random Forest is an ensemble of Decision Trees. An ensemble allows error reduction, resulting in higher accuracy. The reason for this is that it is less likely to overfit than a Decision Tree because it combines the predictions of multiple (smaller) Decision Trees, which reduces the variance of the model. Lower variance results in better generalization and thus leads to

better performance on an unknown dataset. Decision Trees tend to have higher variance since there is a higher chance of overfitting because the tree can grow very complex. A complex tree might work well on the training data but perform worse on the testing data [13].

The five features of all 35 columns that are most important for decision-making in the Random Forest model are:

1. Curricular units 2nd semester (approved)
2. Curricular units 1st semester (approved)
3. Curricular units 2nd semester (grade)
4. Curricular units 1st semester (grade)
5. Tuition fees up to date

After evaluating, we have decided that the untuned Random Forest model is the model that performed best on the dataset due to the reasons listed above. Also, it has a good recall score for the class dropout which is what we wanted to achieve. This is why this model has been saved in a pickle file and can be tested with our streamlit-application (see link in appendix).

5. Discussion and Outlook

After completing the project, where we aimed to predict which students would graduate with a bachelor's degree and which would drop out, we were able to identify what worked well, what could have been improved, and the next steps to take if the project were to continue.

During the execution of this project, the data cleaning process was successful. Using proper correlation analysis, we were able to eliminate all irrelevant columns. Despite removing many columns, we achieved good to very good results with the models.

However, the dataset could have been larger to better train the models. Additionally, the cleaned dataset had a slight imbalanced ratio of "graduate" to "dropout." A balanced ratio would have been better, as the model now tends to predict "graduate" more often. Nevertheless, we did not want to remove more rows due to the dataset's size.

If the project were to be continued, these are the next steps we would take. Firstly, we could train more models, for example, different ensemble methods. Also, existing models could be further optimized. Moreover, the dataset is currently tailored to the American university system. To enable German universities to use this model, the data would need to be converted to the German grading system, and the options translated into German. Additionally, it could be implemented to not only predict whether someone will graduate but also with what grade they would graduate.

References

[1] S. Carballo. "2024 Guide: What is Student Retention in Higher Education?" Element451. (2024), [Online]. Available: https://element451.com/blog/what-is-student-retention.

[2] B. Perez, C. Castellanos, and D. Correal, "Applying Data Mining Techniques to Predict Student Dropout: A Case Study," Medellin: IEEE, May 2018, pp. 1–6.

[3] M. Naseem, K. Chaudhary, B. Sharma, and A. G. Lal, "Using Ensemble Decision Tree Model to Predict Student Dropout in Computing Science," Melbourne, Australia: IEEE, Dec. 2019, pp. 1–8.

[4] K. Al Mayahi and M. Al-Bahri, "Machine Learning Based Predicting Student Academic Success," Brno, Czech Republic: IEEE, Oct. 2020, pp. 264–268.

[5] T. Nema and S. Palwe, "Predicting Students' Academic Success using Machine Learning Algorithms," Pune, India: IEEE, Aug. 18, 2023, pp. 1–7.

[6] pandas. "Pandas." (2024), [Online]. Available: https://pandas.pydata.org/ (visited on 06/20/2024).

[7] he Matplotlib development team. "Matplotlib: Visualization with Python." (2024), [Online]. Available: https://matplotlib.org/ (visited on 06/20/2024).

[8] M. Waskom. "Seaborn: Statistical data visualization." (2024), [Online]. Available: https://seaborn.pydata.org/ (visited on 06/20/2024).

[9] "Scikit-learn Machine Learning in Python." (2024), [Online]. Available: https://scikit-learn.org/stable/ (visited on 06/20/2024).

[10] NumPy team. "NumPy." (2024), [Online]. Available: https://numpy.org/ (visited on 06/20/2024).

[11] "SciPy." (2024), [Online]. Available: https://scipy.org/ (visited on 06/20/2024).

[12] Python Software Foundation. "Importnb 2023.11.1." (Nov. 2, 2023), [Online]. Available: https://pypi.org/project/importnb/ (visited on 06/20/2024).

[13] *Explanations Are Based on Knowledge from Previous Lectures*.

Appendix

You can find our repository here: https://github.com/hannah127/EduPredictors

Goal of this Project

The goal of this project is to predict academic success and thereby whether a student will dopout or graduate. Therefore, we will train four different Machine Learning (ML) models for classification. In addition, we will evaluate which model has the best prediction results and visiulize the model through a streamlit app.

Course of Action

For this we will do a data cleanup first. Secondly, we will train the models and try to improve them using hyperparameter tuning. Thirdly, we will evaluate the different models.

1. Data Cleanup

During the data cleanup we will remove all rows which have NaN values. Following, we will analyze the correlation. Since the dataset is for classification we use the Spearman correlation instead of the Pearson correlation which is for linear correlation. Moreover, we remove columns which have a small correlation with the target column. Finally, we remove all rows which have the target value "enrolled" to have a two class problem.

2. Training of ML-Models

We decided to train the four following models: k-Nearest-Neigbours, Random Forest, Decision Trees and Naive Bayes. To train to we split the dataset first in 70% training data and 30% testing data first. Secondly, we train those models using the Python library scikit-learn. Thirdly, we improve the performance of every model using hyperparameter tuning using GridSearchCV of scikit-learn.

3. Evaluation

Confusion Matrix and the ROC-curve are methods to compare and evaluate the performance of ML models. Moreover, we will calculate the accuracy, precision, recall and f1-score. This is helpful for evaluating and comparing the performance of the differnent values as well. Finally, we will interprete all results to justify which model has the best results.

Execution of the Notebooks

We decided to use Jupyter Notebooks, because the output (e.g. grafics) of the belonging cells can be seen and analysized below. Therefore, it is a convenient method to analyze data and ML models.

All the packages that are needed to execute the Jupyter Notebooks can be installed depending on the pip version with ‘pip install -r requirements.txt‘ or ‘pip3 install -r requirements.txt‘.

The order of the notebooks is the following:

1. Data Cleanup
2. Training ML Models
3. Evaluation

You can find our application here:

https://hannah127-data-exploration-projekt-app-bcdugr.streamlit.app/.