José Antonio Sánchez Castro

Student dropout prediction

Contents

[Introduction and objectives 2](#_Toc183183497)

[Introduction 2](#_Toc183183498)

[Objectives 2](#_Toc183183499)

[Characterization, pre-processing, and explanation of techniques for variation in accuracy across different training splits using cross-validation. 2](#_Toc183183500)

[Hyperparameter tuning 8](#_Toc183183501)

[Hyperparameter tuning techniques 8](#_Toc183183502)

[Grid Search 8](#_Toc183183503)

[Random Search 8](#_Toc183183504)

[Evolutionary Algorithms 8](#_Toc183183505)

[Conclusions 8](#_Toc183183506)

[References 8](#_Toc183183507)

[Github link 8](#_Toc183183508)

# Introduction and objectives

## Introduction

A student dropout is a person who leaves school or university before finishing their degree.

Dropout can occur at any level (college, high school …) and is influenced by multiple factors including academic difficulties and financial problems.

This phenomenon affects these individuals and their society, deeply impacting them financially and socially. Not only does it affect the individual’s employment prospects, earning potential and overall well-being, but also has a broader impact on the economy.

Machine Learning (ML) is a powerful tool that can be used for predicting and understanding student dropout. Multiple algorithms (like decision trees, random forests and neural networks) have been used in this type of prediction in the literature.

For instance in Osemwegie & Amadin, 2023 logistic regression was used to predict student dropout with a 98,9% performance.(Osemwegie and Amadin, 2023)

## Objectives

The objectives of this project are:

1. Develop a machine learning model able to predict student dropout accurately based on the UCI dataset “Predict Students' Dropout and Academic Success”. (Realinho Valentim and Baptista, 2021)
2. Identify key factors affecting student dropout.
3. Evaluate model performance using accuracy, F1 score…

# Characterization, pre-processing, and explanation of techniques for variation in accuracy across different training splits using cross-validation.

In order to analyze the data, its cleaning and pre-processing is necessary.

I conducted some exploratory data analysis (EDA) beginning by using the pandas function shape to see the amount of observations and features in my dataset.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

The dataset contains 4424 observations and 37 features.

I followed by checking for missing or NA values, as well as what type of features (numerical or categorical) are in the dataset. I used the info function. As you can see in the picture below, the non-null count corresponds to the total number of observations – so there are no null values -. Also, all the data types are numeric except for the target variable.

Tabla

Descripción generada automáticamente

I continued by checking some basic statistics of the data, using the describe function. It can be seen that the features have very different values and distributions. However, it is not clear if there are any outliers yet ( this will be checked later in an outlier analysis ).

Tabla

Descripción generada automáticamente

Since the target variable is categorical I have used label encoding to make sure it works for every machine learning algorithm.

Interfaz de usuario gráfica

Descripción generada automáticamente con confianza baja

Now that everything is encoded, I began the outlier analysis. First, I scaled the data and used boxplots to see the amount of outliers.

Gráfico, Gráfico de dispersión

Descripción generada automáticamente

As it can be seen in the illustration above, there is a big quantity of outliers for almost every feature.

Because the data contains so many outliers, loss of information is possible. Therefore instead of using handling methods and the interquartile range, I just deleted the rows that were below 0,5 percentile or above the 99,5 percentile.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

The result was the deletion of 276 observations or 6,24% of the original dataset rows.

I also checked if the dataset is balanced:

Gráfico, Gráfico de barras

Descripción generada automáticamente

The dataset is highly imbalanced, with graduate being the majority and enrolled the minority class. This will need to be solved before the machine learning stage.

A correlation matrix was also made to check possible redundancies in the features. Four groups of 2 features were seen to have high correlation as per the picture below:

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

This might indicate redundancy. Due to the number of features in the dataset (37) I considered that feature selection could be appropriate to avoid overfitting. I used the importance instead of techniques like Principal Component Analysis as I want these features to remain interpretable.

I divided the data in training and test data and used both SMOTE and NearMiss to make the dataset balanced across the different splits. Then, the random forest was used to identify the features with an importance index less than 0,03. These features were dropped from the dataset.

Texto

Descripción generada automáticamente

24 columns were dropped from each dataset.

Following this I used a GridSearch for hyperparameter tuning. In a first instance, I used RobustScaler and also the SMOTE and NearMiss methods. In a second instance, I didn’t use the RobustScaler since there was a lot of overfitting in the data.

# Hyperparameter tuning

Hyperparameter tuning aims to find parameters that will make a ML model’s performance achieve its maximum. These parameters are settings that are predefined in the model, such as the number of decision tree splits.

Hyperparameter tuning tries to achieve better model accuracy and better generalization (avoiding underfitting and overfitting of the model).

## Hyperparameter tuning techniques

Grid Search

Grid Search evaluates all the possible combinations of hyperparameters given. It checks every possibility, which makes it guaranteed to find the best possible combination of the given hyperparameters.(*3.2. Tuning the hyper-parameters of an estimator — scikit-learn 1.5.2 documentation*, no date)

Random Search

Since Grid Search can be computationally very expensive (evaluating the dataset for every hyperparameter), Random Search can be used in some circumstances.

It selects random combinations of hyperparameters from the given and evaluates them, instead of evaluating all of them like Grid Search would.(Bergstra and Bengio, 2012)

Evolutionary Algorithms

These algorithms are based on the natural selection and evolution process. In this case, a population of candidate solutions will be given (hyperparameter combinations). They will be assessed for fitness, with the fittest ones “surviving” (going to the next generation) and the other solutions being erased. Randomness, combination of hyperparameters combinations and selection are used in an iterative manner to achieve a solution.(Schmidt *et al.*, 2019)

# Conclusions

Across the three training splits 10%, 15% and 25%:

* The 15% split had the best results. The models had better accuracy and there were more good fits. This is probably because with 10% there is more training data, leading to the model learning the noise in it as well. With 25%, the model’s training data might not be enough to generalize well, which explains why the models appear as overfitted and have less accuracy.

Scaling vs no scaling:

* Scaling has had a minor negative effect in accuracy and generated more overfitting in most models. This can be due to the model weighing less significant features more heavily. Also, some algorithms like decision trees don’t need this scaling. The neural network in the 25% test split is a big exception (63% accuracy with, 19% without) , since neural networks heavily rely on scaling functions.

What model to choose?

When choosing the model, the interpretability of it is very important for a task like this. Also, seeing as the trees have the best accuracy, the decision tree in the 15% split without scaling is the best option.

What are the most important features?

Imagen de la pantalla de un celular de un mensaje en letras blancas

Descripción generada automáticamente con confianza baja

As it can be seen, “Curricular units 2nd sem” are the features with the most importance when predicting dropout.

# References

*3.2. Tuning the hyper-parameters of an estimator — scikit-learn 1.5.2 documentation* (no date). Available at: https://scikit-learn.org/1.5/modules/grid\_search.html#grid-search (Accessed: 22 November 2024).

Bergstra, J. and Bengio, Y. (2012) ‘Random search for hyper-parameter optimization.’, *Journal of machine learning research*, 13(2).

Osemwegie, E. and Amadin, F. (2023) ‘STUDENT DROPOUT PREDICTION USING MACHINE LEARNING’, *FUDMA JOURNAL OF SCIENCES*, 7, pp. 347–353. Available at: https://doi.org/10.33003/fjs-2023-0706-2103.

Pedregosa, F. *et al.* (2011) ‘Scikit-learn: Machine Learning in Python’, *Journal of Machine Learning Research*, 12(85), pp. 2825–2830. Available at: http://jmlr.org/papers/v12/pedregosa11a.html.

Realinho Valentim, V.M.M.M.J. and Baptista, L. (2021) ‘Predict Students’ Dropout and Academic Success’.

Schmidt, M. *et al.* (2019) ‘On the performance of differential evolution for hyperparameter tuning’, in *2019 international joint conference on neural networks (IJCNN)*, pp. 1–8.

The pandas development team (2020) ‘pandas-dev/pandas: Pandas’. Zenodo.