Machine Learning I

The Wizardry School Enrollment

Final Project

Group 08

18 November 2023

Carolina Almeida	20221855
Duarte Carvalho	20221900
Francisco Gomes	20221810
Margarida Henriques	20221952
Marta Monteiro	20221954

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação
Universidade Nova de Lisboa

ABSTRACT

The following study aims to implement predictive modelling in the context of the education sector, namely, the esteemed wizardry schools. The objective is to predict the admission status of aspiring enchanters, a process that demands meticulous consideration, since only those deemed worthy are granted the privilege of attending these prestigious institutions.

To achieve this objective, a comprehensive Machine Learning pipeline was constructed. This involves a thorough preprocessing and analysis of historical student admission data, the creation of pertinent features for prediction, and the evaluation of the relevance of existing features through diverse methods. Additionally, the study incorporates data partitioning and the selection of optimal parameters, via GridSearchCV.

The culmination of these efforts led to the development of a predictive model, *Decision Tree Classifier*, tailored to align with the desired goal while maintaining relatively low bias. After several evaluation efforts (which include analyzing the best_score_ model, obtaining the F1 Score, Balanced accuracy, Precision and Recall as well as studying the effects of data splitting), this model proved to have superior performance, achieving the most promising scores considering the circumstances of an unbalanced dataset.

Despite effectively addressing the imbalance challenge, further investigation suggests the potential for even more sophisticated techniques such as under and oversampling, in future research occasions.

This study successfully attains its outlined objectives, enhancing the possibility for a more organized and accurate admission process for wizardry schools. The findings not only contribute to this specific context but also underscores the significance of Supervised Learning techniques in educational settings. Moreover, it sets the stage for future work, paving the way for continuous improvement within the educational sector.

KEYWORDS

Machine Learning; Data Partitioning; Predictive Model; Unbalanced Dataset; Supervised Learning

INTRODUCTION

Due to not only their grandeur but also their ability to unlock the hidden powers within each student, the esteemed wizardry schools stand as the epicentre of aspirations for young wizards and witches who wield the spark of magical potential. However, not every enchanter is deemed worthy of attending these prestigious institutions. Therefore, discerning who shall earn the privilege of stepping into the world of wizardry becomes a priority, that, must be determined precisely. Hence, delving into their qualifications, backgrounds and unique magical potential must be done to successfully tackle the mission.

The work here conferred relies on a representative sample of the data, including information about past student's admission status. Using the training set we must develop a predictive model and then use it to make accurate predictions on the testing set.

Additionally, it is important to delve into the limitations inherent in this study as well as conduct a comprehensive analysis of potential expansion beyond the present context and dataset.

In answering this question, this document is organized as follows: 'Background' provides a theoretical explanation for the techniques and algorithms not lectured during the practical classes, 'Methodology' details the methods employed as well as the materials used, 'Results' organizes the findings, 'Discussion' takes and interprets the findings, 'Conclusion' presents the outcome of the work and, finally, 'References' contains sources for external materials used.

BACKGROUND

SEQUENTIAL FEATURE SELECTION

The identification of pertinent features constitutes a crucial aspect to the success of a model. Therefore, the incorporation of Feature Selection methodologies is essential when developing a Machine Learning project. Within the Wrapper Methods, Sequential Feature Selection (SFS) emerges as a prominent procedure, encompassing many Greedy search techniques, that seek for the optimal set of features in an interactive process. This method, within its framework, delineates two distinctive approaches, Forward Selection and Backward Selection. For this project we have implemented the former, employing the *SequentialFeatureSelection* module, from the *Scikit-learn* library.

The methodology of Forward-SFS is characterized by its incremental nature, that is, the process initiates with an empty set of chosen features and fits them to a model instance. Once a predetermined stopping condition is satisfied, which in our case is defined by the number of chosen features, the results from these fitted models are evaluated using Cross Validation. The feature that generated the model with the highest score is assimilated to the selected features. This way, only the most influential features are incorporated.

EXTREME RANDOMIZED TREES, EXTREME GRADIENT BOOSTING, SUPPORT VECTOR MACHINES

During the project's development phase, three distinct machine learning algorithms were employed to construct predictive models. These algorithms, which were not covered in practical sessions, encompassed Extreme Randomized Trees (Extra Trees), Extreme Gradient Boosting (XGBoost), and Support Vector Machines (SVM). The theoretical explanation of the aforementioned algorithms is reported immediately following.

Extreme Randomized Trees (Extra Trees), classified as an ensemble machine learning method, builds upon the Decision Tree Classifier and Random Forest Classifier principles. Its unique differentiators lie in the fact "that it splits nodes by choosing cut-points fully at random and that it uses the whole learning sample to grow the trees". The final prediction, "by majority vote in classification problems and arithmetic average in regression ones", is derived through the aggregation of the predictions of all the trees.

Extreme Gradient Boosting (XGBoost), a "scalable end-to-end tree boosting system" that provides a parallel tree boosting framework, renowned for its efficiency in addressing various data science problems with speed and accuracy.

Support Vector Machines (SVM), a "supervised machine learning algorithm used for both classification and regression". It is known for its effectiveness in "high-dimensional data" and its versatility in modelling different types of data.

These algorithms played a pivotal role in the development phase, contributing significantly to the selection of the most effective model for generating the final predictions. The implementation of Extreme Randomized Trees, Extreme Gradient Boosting and Support Vector Machines was employed using appropriate modules from the dedicated libraries, those being, the *Scikit-learn* for the first two and the *XGBoost* for the last.

METHODOLOGY

SETTING THE ENVIRONMENT AND DATA PREPARATION

The development of this project was undertaken within the Jupyter Notebook, serving as the primary programming environment. In addition, a well-structured table of contents, based on those established during practical classes, was created to enable quick and efficient access to information relevant to the desired topic.

The first and crucial phase of any Machine Learning project involves data preparation. This intricate process entails the cleaning, transformation, and organization of raw data into a format more conductive to modelling. This process initiates with the meticulous importation of pertinent datasets and fundamental libraries such as *Pandas*, *NumPy and Scikit-learn*, which sets the groundwork for subsequent analytical endeavours.

Upon successful data importation, the focus shifts to the acquisition of preliminary insights. Methods such as *.head()* and *.info()* are employed to examine the initial rows, offering an overview of the data's structure, types and missing values, thereby identifying potential challenges.

In an effort to optimize memory usage and enhance computational efficiency, specific variables' data types were strategically converted into more appropriate ones. This was immediately followed by a meticulous assessment of missing data, which, recognizing the fundamental requirement for non-null values in the majority of Machine Learning algorithms, assumes outstanding importance. A threshold was established whereby variables exhibiting more than 50% missing entries were deemed to be of limited predictive value and were thus excluded from further analysis. The remaining missing values were judiciously imputed using the *KNNImputer* class from the *sklearn.impute* module. Through a systematic evaluation process, it was determined that setting the number of neighbours (k) to 1 yields the most accurate imputation for the dataset, minimizing discrepancies between original and imputed values.

DATA EXPLORATION AND TREATMENT

Prior to proceeding with the model construction phase, preliminary insights were derived by exploring both the training and test datasets. This manual exploration involved the use of the *.describe()* method to extract general descriptive statistics for the numerical features. This process allowed an understanding of the general behaviour exhibited by the school admission data and the target variable and thereby facilitating the identification of potential inconsistencies and outliers.

For enhance understanding, a function was developed to systematically highlight variables containing outliers. This function served as a valuable tool for identifying data points that derivate significantly from the expected distributions, aiding in the early detection of potential anomalies.

To further fathom how the individual features are distributed in relation to the target, bar charts and histograms were employed for visual representation. The graphical representations served as means to discern any features that exhibit a notably different distribution compared to the dependent variable.

FEATURE ENGINEERING

During the data exploration and treatment phase, it became evident that certain features could be introduced, potentially enhancing the depth of the dataset. For this reason, the two following features were created:

1. **Experience Level Category**: This newly introduced feature serves as a categorical predictor that categorizes students into specific groups, providing insights into their progression within the wizardry school, in terms of their skill and knowledge. It includes the categories Novice, Intermediate, Advanced and Master.
2. **Financial Background Category**: This is a categorical predictor derived from the existing "Financial Background" feature. It classifies students into distinct groups, providing a more nuanced understating of their economic circumstances – Low, Medium or High.

DATA PARTITIONING

The dataset underwent preparations for the modelling stage, specifically, the training data was further divided into a training set and a validation set. This partitioning serves two main purposes, those being, optimizing the model and measuring its performance, thereby enabling the comparison between different models. Essentially, 70% of observations from the training dataset were allocated for the model training, while the remaining 30% were designated for model validation.

ENCODING AND SCALING OF THE DATA

Once the data is partitioned, the next crucial step in the data exploration and treatment phase is data encoding. This step is particularly important when dealing with categorical variables that contain labels or categories rather than numerical values. Since most Machine learning algorithms perform optimally with numerical inputs, the conversion of these categorical variables into a numerical format becomes crucial. This process is commonly referred to as data encoding.

The chosen approach for data encoding was One Hot Encoding, where each unique category within a categorical variable is represented by its own numerical column. In this representation, the presence of a specific category is denoted by 1, while its absence is represented by 0.

Following the encoding of the data, data scaling was applied to the numerical features. Given the presence of outliers, the training, validation, and testing data (excluding the variable) were scaled using the **Robust Scaler**, since it was determined to be the most efficient in minimizing its influence.

FEATURE SELECTION: FILTER METHODS, WRAPPER METHODS AND EMBEDDED METHODS

In the pursuit of reducing data dimensionality and selecting impactful features for the prediction of student admissions, several Feature Selection techniques were employed, specifically, Filtering Methods, Wrapper Methods, and Embedded Methods.

Regarding Filter Methods, variance analysis was conducted across all features to identify invariant variables, which, if present, were subsequently removed from the datasets. Numerical features underwent through Spearman's Correlation Coefficient, also known as "Spearman's Rank". Pairs of features exhibiting high correlation (i.e., their correlation coefficient exceeded a threshold of |0.7|), were evaluated, being one of them eliminated. Additionally, it also helped identify features with the highest correlation values against the target value. Predictors with a coefficient above |0.08| were deemed relevant for predicting student admissions. Categorical predictors were subjected to a Chi-Squared Independence test to assess the independence between the features and the target, with a significance level of 5%. The function used for this was extracted from practical lessons.

Within the realm of Wrapper Methods, both Recursive Feature Elimination (RFE) and Forward Sequential Feature Selection (SFS) were applied to numerical features. The optimal number of predictors for each method was determined.

In addition, Embedded Methods were utilized, namely Lasso and Decision Trees, to evaluate the importance of numerical features. The results were visually represented to underscore the significance of each feature, using functions provided during the practical lectures.

To decide about predictor inclusion in the subsequent modelling phase, a majority vote system was used across all techniques, assigning equal weight to each method.

MODEL EXPERIMENTATION AND CRITERIA FOR EVALUATION

In the developmental phase of the model, various algorithms with diverse parameter configurations were employed to identify the most effective combination for constructing the predictive model. The *GridSearchCV()* method, from the *Scikit-learn* library, played a pivotal role in this process, tuning parameters to optimize the F1 score. The best-performing models were meticulously compared to determine the most suitable candidate for the final model assembly.

However, consistency in predictions on new data is crucial to prevent overfitting. This was assessed by applying two distinct cross-validation techniques: *K-fold and Repeated K-fold*. Cross-validation, a statistical technique, is employed to access the performance of machine learning models on unseen data. It involves splitting the data into k non-overlapping subsets, using one subset as a test set while the remaining subsets as training sets for *k* iterations.

While K-fold cross-validation represents a simple and common method that can be easily implemented using *Scikit-learn*, it may produce noisy results due to varied data splits in each run. To addressed this, the Repeated K-fold cross-validation was adopted, repeating the K-fold process n times with different randomizations in each repetition. This approach enhances the reliability of the model performance estimates by averaging across multiple runs.

It was deemed essential to prioritize metrics beyond the Kaggle score, which evaluates only 16 lines of our prediction. Instead, the F1 score results (in the Jupyter Notebook) for the positive class, were considered as the primary focus. This approach ensures a more accurate analysis, extending beyond what is reflected in the limited Kaggle evaluation.

ASSEMBLING THE MODEL AND EXTRACTING PREDICTIONS

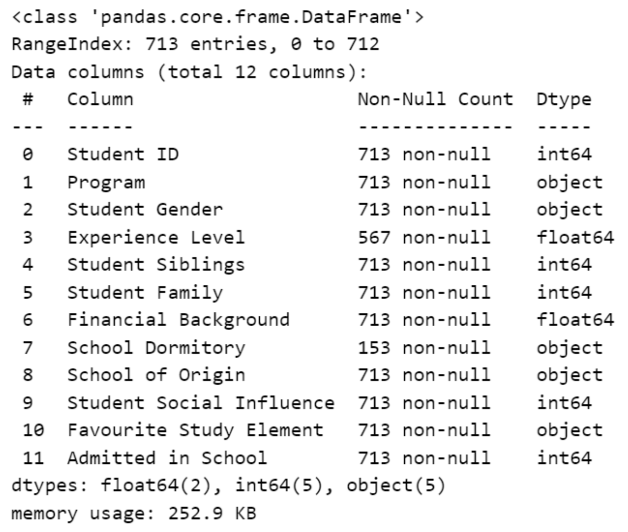
Upon identifying the most effective features and exploring various, the selected model was trained on 70% of pre-processed training data, which consisted solely of the optimal features and pre-determined parameters. Validation on the remaining 30% assessed its performance. The finalized model then predicted on the test dataset, and these predictions were submitted to the Kaggle competition.

RESULTS

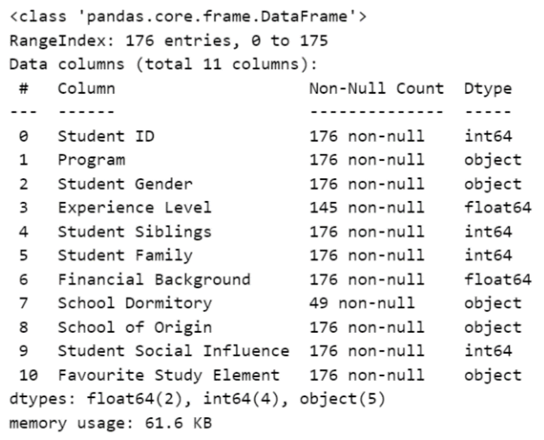
DATA PREPARATION

In the preliminary data analysis, a systematic examination was conducted on the first few rows of both the training and testing dataset. It aimed to assess data types, memory usage and the presence of missing. Regarding memory usage, appropriate data types were assigned to all features, except for "**Financial Background**". While analyzing the features, it was also concluded that **"School Dormitory"** exceeded the threshold of 50% of missing values, which resulted in its removal. Regarding **"Experience Level"**, the *KNNImputer* from *sklearn.impute* was employed.

After performing such adjustments, the decrease in memory usage was clear. The first row of images exhibits the memory usage prior to any modifications, while the second one refers to the memory usage after them.

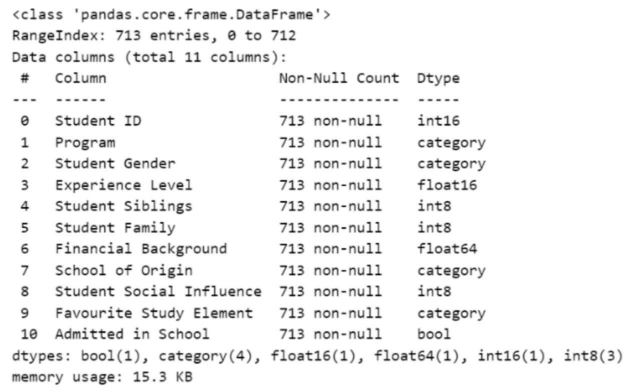


```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 713 entries, 0 to 712
Data columns (total 12 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Student ID                713 non-null    int64
 1   Program                   713 non-null    object
 2   Student Gender            713 non-null    object
 3   Experience Level          567 non-null    float64
 4   Student Siblings          713 non-null    int64
 5   Student Family            713 non-null    int64
 6   Financial Background      713 non-null    float64
 7   School Dormitory          153 non-null    object
 8   School of Origin          713 non-null    object
 9   Student Social Influence  713 non-null    int64
 10  Favourite Study Element   713 non-null    object
 11  Admitted in School        713 non-null    int64
dtypes: float64(2), int64(5), object(5)
memory usage: 252.9 KB
```

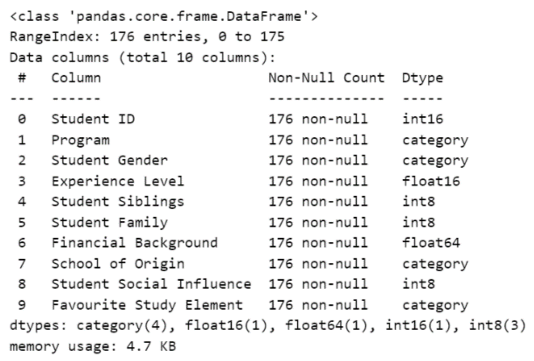


```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 176 entries, 0 to 175
Data columns (total 11 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Student ID                176 non-null    int64
 1   Program                   176 non-null    object
 2   Student Gender            176 non-null    object
 3   Experience Level          145 non-null    float64
 4   Student Siblings          176 non-null    int64
 5   Student Family            176 non-null    int64
 6   Financial Background      176 non-null    float64
 7   School Dormitory          49 non-null     object
 8   School of Origin          176 non-null    object
 9   Student Social Influence  176 non-null    int64
 10  Favourite Study Element   176 non-null    object
dtypes: float64(2), int64(4), object(5)
memory usage: 61.6 KB
```

Figure 1 – Initial Datatypes and *Memory Usage*



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 713 entries, 0 to 712
Data columns (total 11 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Student ID                713 non-null    int16
 1   Program                   713 non-null    category
 2   Student Gender            713 non-null    category
 3   Experience Level          713 non-null    float16
 4   Student Siblings          713 non-null    int8
 5   Student Family            713 non-null    int8
 6   Financial Background      713 non-null    float64
 7   School of Origin          713 non-null    category
 8   Student Social Influence  713 non-null    int8
 9   Favourite Study Element   713 non-null    category
 10  Admitted in School        713 non-null    bool
dtypes: bool(1), category(4), float16(1), float64(1), int16(1), int8(3)
memory usage: 15.3 KB
```



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 176 entries, 0 to 175
Data columns (total 10 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Student ID                176 non-null    int16
 1   Program                   176 non-null    category
 2   Student Gender            176 non-null    category
 3   Experience Level          176 non-null    float16
 4   Student Siblings          176 non-null    int8
 5   Student Family            176 non-null    int8
 6   Financial Background      176 non-null    float64
 7   School of Origin          176 non-null    category
 8   Student Social Influence  176 non-null    int8
 9   Favourite Study Element   176 non-null    category
dtypes: category(4), float16(1), float64(1), int16(1), int8(3)
memory usage: 4.7 KB
```

Figure 2 – Optimized Datatypes and *Reduced Memory Usage*

DATA EXPLORATION AND TREATMENT

A meticulous examination of feature distributions, both in general context and concerning the target variable, is crucial to identify possible outliers and inconsistencies prior to model development. Firstly, descriptive statistics provided some initial insights were, revealing that "Student Siblings" and "Student Family" exhibited the lowest standard deviations. Despite being difficult to confirm using only the descriptive statistics table, it was deduced that there were some outliers in the features **“Experience Level”**, **“Financial Background”**, and **“Student Siblings”**. To gain a deeper understanding of this matter, a function was developed to identify the presence of outliers and the presence of outliers in the features previously mentioned was confirmed, as well as in the **“Student Family”** feature. Also, an evaluation of the target variable revealed an uneven distribution, with only **34.35%** of students admitted, highlighting the imbalance of the dataset.

Visualizations were then crafted to better interpret distributions in relation to the target. Analysis of numerical data revealed that in **“Experience Level”** most students, regardless of their admission status, fall within the 20-30 experience level range, emphasizing the importance of segmenting this feature for a more nuanced analysis. Between **“Student Family”** and **“Student Siblings”**, as expected, there is a positive correlation which results in an identical right-skewness in both graphs. The existence of outliers in the features identified before were once again confirmed through visual inspection. Regarding categorical data, the features demonstrate an unbalanced distribution, which could bias further analysis.

FEATURE ENGINEERING

To enhance the comprehension of the different stages of students’ experience level and facilitate a more insightful analysis of their progression, in terms of skill and knowledge, within the wizardry school, a new feature **“Experience Level Category”** was created. The categories were defined as Novice, Intermediate, Advanced and Master.

Additionally, **“Financial Background Category”**, was created. This new feature provides a more nuanced understanding of the economic circumstances of the students and the categories were defined as Low, Medium, and High.

FEATURE SELECTION

The Feature Selection process started with Filter Methods. Initially, the existence of univariate variables was checked, and it was assessed that all the features exhibit non-zero variance, which indicates variability and diversity within the dataset. Then, an analysis of *Spearman Correlation* between the features and the target variable revealed a alack of highly correlated variables with the target. Nonetheless, a subtle correlation was observed between “Student Family” and “Financial Background” with “Student Siblings” which aligns with logical expectations. Given the absence of strong correlations, it was essential to employ alternative feature selection methods. The *Chi-Squared Test* recommended the exclusion of “**Favourite Study Element**”.

Regarding Wrapper Methods, *RFE* identified all numerical features as important. However, in *SFS* only two features were classified as important – “**Financial Background**” and “**Student Social Influence**”.

Delving into the Embedded Methods and according to both *Lasso Regression* and *Decision Tree Classifier*, the feature “**Student Family**” should be discarded. Additionally, the *Decision Tree Classifier* also assigned “**Student Siblings**” as an irrelevant feature for prediction.

The results of these procedures are all summarized in the tables below.

Numerical Data	Spearman's Correlation	RFE	SFS	Lasso	Decision Tree	Relevance
Experience Level	Discard	Keep	Discard	Keep	Keep	**Keep**
Student Family	Discard	Keep	Discard	Discard	Discard	**Discard**
Student Siblings	Discard	Keep	Discard	Keep	Discard	**Discard**
Student Social Influence	Discard	Keep	Keep	Keep	Keep	**Keep**
Financial Background	Discard	Keep	Keep	Keep	Keep	**Keep**

Table 1 - *Numerical Features' Selection Results*

Categorical Data	Chi-Square	Relevance
Program	Keep	**Keep**
Student Gender	Keep	**Keep**
School of Origin	Keep	**Keep**
Favourite Study Element	Discard	**Discard**
Experience Level Category	Keep	**Keep**
Financial Category	Keep	**Keep**

Table 2 - *Categorical Features' Selection Results*

MODEL RESULTS

Initially, within the scope of our methodology, a suite of classifiers and their parameter synergies were meticulously evaluated via *GridSearchCV()*. Using *.best_score_*, which is the mean cross-validated score of the best estimator, we were able to identify the *Decision Tree* Classifier as the one achieving the highest score. In the table below the top 5 models are registered, those being, *Decision Tree Classifier*, *Multilayer Perceptron Classifier*, *Extra Trees Classifier, XGBoost Classifier and Gradient Boosting Classifier* from *Scikit-learn*.

Model	Score
Decision Tree Classifier	0.7607845230246519
Multilayer Perceptron Classifier	0.754915084915085
Gradient Boosting Classifier	0.7470335094642454
XGBoost Classifier	0.7445049183283416
Extra Trees Classifier	0.744318953023163

Table 3 – *GridSearchCV() Scores*

Secondly, while the Kaggle results awarded a perfect score to the *DecisionTreeClassifier* without hyperparameter adjustments, this does not align with our comprehensive cross-validation findings, where the same model indicated signs of overfitting. The Kaggle score's perfection, based on a mere 16-line evaluation, may not reflect true model performance but rather a coincidental match within a limited dataset subset.

Our decision to proceed with the model predictions was based on a whole consideration of various performance metrics. These included the F1 score for the true label, balanced accuracy, precision, recall, and an in-depth analysis of the confusion matrix. This comprehensive approach allowed us to draw more informed and conclusive insights. The outcomes from the most effective models are delineated below.

Model		F1 Score	Precision	Recall	Balanced Accuracy	TP	TN	FP	FN
Decision Tree Classifier	**Training Score**	0.788	0.805	0.773	0.835	290	136	33	40
	Validation Score	0.700	0.766	0.645	0.768	123	49	15	27
Extra Trees Classifier	**Training Score**	0.765	0.881	0.676	0.813	302	125	21	51
	Validation Score	0.661	0.824	0.553	0.744	131	44	7	32
Random Forest Classifier	**Training Score**	0.783	0.863	0.716	0.827	302	125	21	51
	Validation Score	0.682	0.830	0.579	0.757	131	44	7	32
Support Vector Machine	**Training Score**	0.783	0.891	0.699	0.826	308	123	15	53
	Validation Score	0.688	0.878	0.566	0.761	132	43	6	33

Table 4 – *Evaluation Metrics*

The *Decision Tree Classifier* emerged as the top-performing estimator when gauged against the metrics we prioritized for model evaluation, employing specific parameters optimized for our predictive task. Those parameters are shown below.

Parameter	Value
criterion	'gini'
max_depth	5
max_leaf_nodes	7
min_samples_split	2

Table 5 – *Parameters of Decision Tree Classifier*

Incorporating all the evaluation steps, including the Kaggle score, our final analysis led us to choose the Decision Tree Classifier model. It not only performed admirably on the validation set but also demonstrated a significant balance between sensitivity and specificity, with a marked absence of bias towards any classification outcome.

Moreover, its commendable performance was evident in the test data, where it secured an F1 Score of 0.8888, accounting for 11% of the test evaluations.

Finally, simple data split, *K-fold* and *Repeated K-fold cross-validation* techniques were employed to confirm the supremeness of the Decision Tree Classifier. Analyzing the scores affected by the different data splits, it becomes evident that the model does not present overfitting.

The scores obtained across both training and validation sets are showcased below.

	Train F1 Score	Test F1 Score
Simple Data Split	0.788406	0.700000
K-fold cross-validation	0.781684	0.746146
Repeated K-fold cross-validation	0.781439	0.723477

Table 6 – *Simple Data Split, K-fold and Repeated K-fold cross-validation*

DISCUSSION

The completion of this Machine Learning Project has provided several valuable insights into the process of predicting student admissions within a wizardry school. Moreover, it provided a better understanding for addressing real-world classification challenges. The execution of various stages, including data preparation, exploration and treatment, feature engineering, data partitioning, encoding and scaling, feature selection and model experimentation, has led to the identification of a Decision Tree Classifier as the optimal predictive model.

Despite the robustness of the Decision Tree Classifier, it is essential to acknowledge certain limitations, such as, the dataset's imbalance, with only 34.35% of students admitted, that could introduce bias. However, to address this concern, strategic evaluation metrics such as the F1 Score and Balanced Accuracy were prioritized. After careful research, it was concluded that "apart from using different evaluation criteria […]. Two approaches to make a balanced dataset out of an imbalanced one are under-sampling and over-sampling" (Ye Wu, 2022). In the first case the size of the abundant class is reduced, whereas in the second case, it is increased.

Remarkably, all the models studied demonstrated reasonable performances, underscoring the predictability of student admissions. This outcome suggests the potential for further in-depth research and implementation, aimed at optimizing the overall management of these schools.

The implementation of this predictive model holds crucial significance in the selection of candidates for enrolment in the wizardry schools. It not only aids in the identification of candidates with the highest potential, who align with the institution's academic standards and values, but also contributes to the optimization of the entire admission process. In addition to its role in the selection process, the model assists in the formulation of strategic investments.

The implications of this study extend beyond the specific context of predicting student admissions in a wizardry school. The methodologies employed can be applied to diverse Machine Learning projects. Future research could explore the generalizability of the Decision Tree Classifier to other educational institutions.

Another interesting extension would be to investigate additional features that may contribute to predictive accuracy, such as, studying the impact of interventions on admission rates. This way, we could observe their potential effect on admission outcomes, by simulating scenarios where certain interventions, such as, additional support programs or mentorship initiatives, were made.

CONCLUSION

Our team was tasked with the development of a supervised learning model capable of predicting which aspiring wizards and witches would be accepted into prestigious wizarding schools. Given the reputation of these institutions and their role in nurturing the unique potential of each student, it is crucial to have an accurate and refined model.

The proposed model, the Decision Tree Classifier, whose parameters are described in the results section, fulfilled the conditions necessary for a positive characterization of the model.

Based on this evaluation, and particularly on the characteristics describing the competence and potential of students, it became possible to observe admission patterns at various levels. This proactive tool aids schools in developing and enhancing strategies such as admissions management and student retention.

Moreover, from a business standpoint, the development of this tool also allows better management of available resources and appropriate redistribution of efforts. This can be seen as the starting point for implementing a data strategy in the education sector. In a broader context, this analysis can lead to additional research on the use of machine learning algorithms in the education sector, complementing previous ones.

This method holds potential for further optimization and broader application on a more general scale, such as predicting admissions to schools that differ from the focused schools. This ensures that only the most deserving candidates gain admission to education, thereby upholding the honour and prestige of these institutions.

REFERENCES

[1] Himanshi Singh, “Forward Feature Selection and its Implementation”, Analytics Vidhya, April 9, 2021
[2] P. Geurts, D. Ernst, and L. Wehenkel, “Extremely randomized trees”, Mach. Learn., vol. 63, no. 1, pp. 3–42, 2006.
[5] “Introduction to Boosted Trees”, dmlc XGBoost, 2022
[4] “Support Vector Machine (SVM) Algorithm”, GeeksforGeeks, June 10, 2023
[5] Jason Brownlee, “A Gentle Introduction to k-fold Cross-Validation”, Machine Learning Mastery, 2023, Online
[6] “Cross Validation in Machine Learning”, GeeksforGeeks, 2023, Online