Final Project Report:
Predicting Students' Dropout and Academic Success

Andres Machado, Bret Geyer, Jackson Small, Jadan Colon, and Thomas Tibbetts

Project Statement:

Student retention is a major problem for institutes of higher education, with many resources devoted to analyzing what causes students to drop out or graduate late. When the graduation rate falls, so does a school's ranking and reputation. For this reason it's desirable to reach out to these students early in their academic path, so they can receive extra counseling or attention that may improve their situation. For this project, we attempt to use data collected from university students to predict whether they will be graduates, dropouts, or still enrolled at the end of a typical program term. Some variables of particular interest are the student's age at enrollment, scholarship status, and early grades. We will use classification algorithms with considerations for reducing class imbalance. The objective is to identify students at risk of failing to graduate, so that they can be prioritized for assistance.

Description of Data:

Broadly speaking, this project's focus is predicting the academic success of university students. We are working with a tabular dataset which lists 37 attributes for each of 4,424 students at the Polytechnic Institute of Portalegre, and contains contains 18 categorical features which encode information such as the student's program, marital status, application mode, and their parents' level of education. Some of the quantitative variables encode the student's age and numerical evaluations (0-100) of their admission profile and previous qualifications, and other quantitative variables which detail the number of curricular units for which the student was enrolled, approved, and credited during their first two semesters. The dataset's target variable is a categorical column which designates each student as one of the following:

- **Dropout**; Indicating that the student dropped out by the end of the program's normal term. Where 1421 students classify as Dropouts.
- **Enrolled**; Indicating that the student was still enrolled (not graduated) at the end of the normal term of the program. Where 794 students classify as Enrolled.
- **Graduated**; Indicating the student graduated by the end of the normal term of the program. Where 2,209 students classify as Graduated.

More information about feature definitions and dataset origins can be found here:
https://archive.ics.uci.edu/dataset/697/predict+students+dropout+and+academic+success

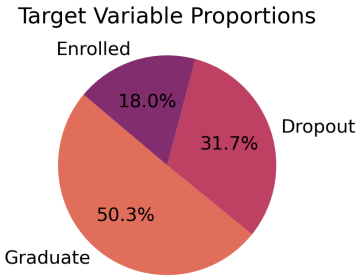
Data Exploration:

One of the dataset's issues is containing many categorical features, each of which consists of many different levels. For example, the "Application mode" variable alone has 18 levels, some of which apply to fewer than 10 students. If all of these categorical features were one-hot encoded, they would add hundreds of sparse columns to the dataset and create unnecessary bulk during the data exploration. For this reason, it was decided that all categorical

feature levels representing less than 2% of the dataset would be binned into one level called “Other”. This significantly reduced the unnecessary complexity in the dataset.

While checking the data, we also noticed that there were 180 students listed as having enrolled in 0 curricular units during both their first and second semester. This may be attributed to faulty data collection or other unusual circumstances. For the purpose of predicting student success, we considered these records to be highly anomalous and decided to remove them from the dataset. After this removal, the dataset contained 4,244 records.

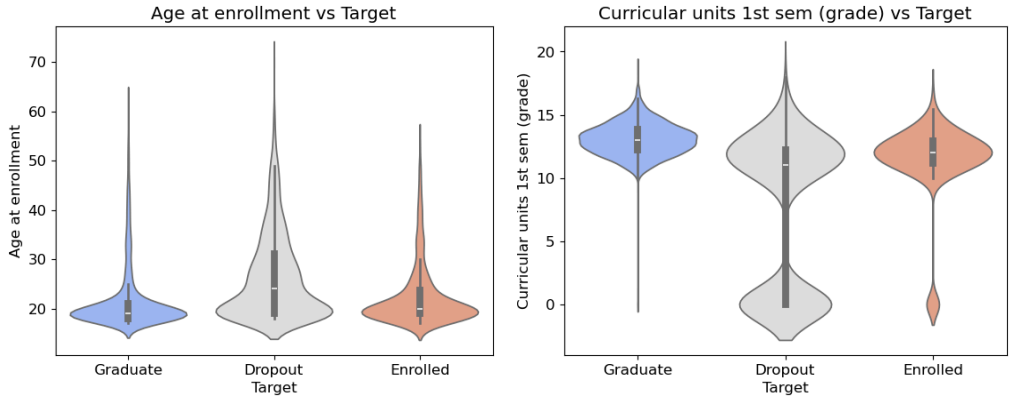
It’s important to note that the target classes are imbalanced. About 50% of the students are graduates, 18% are enrolled, and 31.7% are dropouts. This will inform our modeling and experimental design. To account for the unbalanced classes, it may be necessary to use methods like Stratified K-Folds or SMOTE.



Data Visualization:

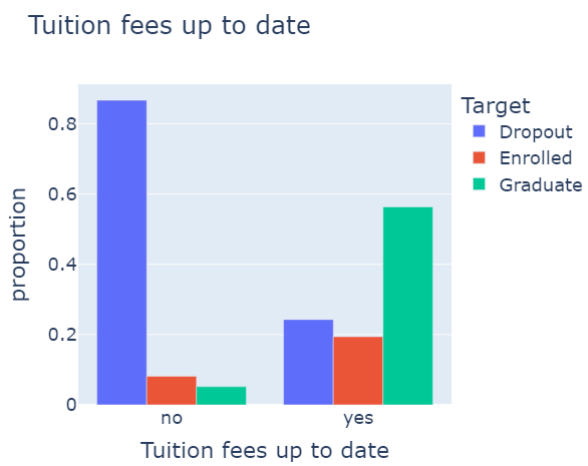
To understand the features that influence the students final outcome, we visualized key features in relation to the target variable. Each of the following plots helped us gain a deeper knowledge of the demographics, financial, and academic patterns that distinguish students from graduates, dropouts, and currently enrolled.

We discovered that students who became dropouts had a heavily right-skewed distribution of age of enrollment as compared to the other classes. We also noticed that within the first semester grades variable, a clear cluster of dropouts had grades centered around 0. In contrast, those students who graduated displayed a more balanced and higher spread of grades.



The “Tuition fees up to date” variable is a binary feature tracking whether the student had fully paid their tuition fees at the time of data collection. In the chart below, it can be seen that a high proportion of students who didn’t keep up with their tuition fees became dropouts. This finding makes us realize how much of an impact financial hardship can play in student retention. Additionally, it was found that students who held scholarships were much more likely

to graduate than those who did not. Overall, these visualizations capture risk factors associated with students who dropped out, graduated or are currently enrolled, but also highlight how early academic performance and financial support are needed for students' success.



Feature Selection:

When starting the feature selection, it was essential to consider how to handle each main variable data type-categorical/nominal and numerical. For each of these data types, feature selection was treated differently; for categorical features, we use Cramer's V statistic, which utilises the Chi-squared statistic and transformations to provide a value between 0 and 1 for a pair of categorical features. Alternatively, for numerical variables, we used Pearson's Correlation Coefficient and the SelectKBest algorithm with the ANOVA F-stat scoring function.

One observation worth noting was a tremendous amount of multicollinearity between 1st-semester features and their 2nd-semester counterparts. To generalize our features as much as possible, we opted to keep the 1st-semester features. We also dropped features that were not clearly defined when we collected the data.

Of the 17 categorical features the dataset provided, 14 were selected. However, out of the 19 numerical features the dataset provided, 7 were selected; meaning, we began with 36 features, but settled on 21 once selection was complete.

Preliminary Modeling:

We established a baseline of evaluation metrics for several different classification techniques, each using 5-fold stratified cross-validation on the dataset. The models were initialized without tuning of hyperparameters. The results are summarized in the table below:

Classification Algorithm	Avg. Precision			Avg. Recall			Avg. F1-Score		
	Grad.	Enr.	Drop.	Grad.	Enr.	Drop.	Grad.	Enr.	Drop.
Logistic regression	0.76	0.54	0.77	0.94	0.18	0.78	0.84	0.27	0.78

Naive Bayes (Gaussian)	0.74	0.28	0.73	0.69	0.49	0.51	0.71	0.36	0.59
K-nearest neighbors	0.71	0.38	0.74	0.85	0.26	0.64	0.78	0.31	0.69
Perceptron	0.74	0.40	0.77	0.92	0.17	0.71	0.82	0.23	0.74
Support vector machine	0.76	0.50	0.80	0.94	0.25	0.73	0.84	0.33	0.77

While evaluating the baseline models, we noticed that the “Enrolled” class had the lowest F1-score across all algorithms. This could be due to it being the smallest minority class in the data. Further modeling may benefit from using oversampling methods like SMOTE to balance the classes in the training set.

We also noted that among the baseline models, the Gaussian Naive Bayes model obtained the best recall for the “Enrolled” class by a significant margin. However, we also saw that the Naive Bayes model underperformed in essentially every other metric. The best overall baseline performance was obtained by the Logistic Regression and SVM models. Both showed a relatively high accuracy of 0.76, and captured the “Graduate” class similarly well. They differ in that the Logistic Regression model achieved a higher recall of the “Dropout” class, while the SVM achieved higher recall of the “Enrolled” class. We would like to tune the hyperparameters of each of these models, check if oversampling the training data improves performance, then try to combine their individual strengths through an ensemble model.

Tuning Individual Models:

We tried to improve each individual model as much as possible before combining them into a final ensemble model. This consisted of tuning each model’s hyperparameters. We used the weighted F1 score as our metric because of our class imbalances. Below is a summary of the results of hyperparameter tuning each model:

- Logistic Regression: The most substantial opportunity for hypertuning this model is manually adjusting the impact of a regularization term, in order to prevent overfitting to the training data. We found that after hypertuning with GridSearchCV, the L1 penalty is optimal for our Logistic Regression model which promotes sparsity in our coefficients. The optimal parameters included a C value of 10, maximum iterations of 500, and using the ‘saga’ solver instead of ‘liblinear’. The weighted F1 score for our optimal model is 0.73. However, although overall model accuracy increased, the predictions for the Enrolled class still lagged behind.

- Support Vector Machine: The Support Vector Machine allows the use of different kernel functions, rather than the default radial kernel. It also permits adjustment of the parameter C, which controls the penalty of margin violations and thus affects the width of the margin. We found that after hypertuning with GridSearchCV, we found that a C value of 10 was optimal. Also, the radial kernel remained as the best kernel and leaving the ‘gamma’ parameter as ‘auto’ made the model optimal. The weighted F1 score for our optimal model was also 0.74.

- Naive Bayes: Due to the nature of the algorithm, there was not much flexibility to tune the Gaussian Naive Bayes model. For this reason, we decided to simply pass it through with default parameters to the later modeling processes.

Consideration of Oversampling Methods (SMOTE):

Suspecting that the imbalance of classes was behind the lackluster prediction of the "Enrolled" class, we considered the use of oversampling methods to rebalance classes in the training data. We specifically used the SMOTE (Synthetic Minority Over-Sampling Technique) algorithm to implement this rebalancing. Using the optimal hyperparameters we found in the previous section, we experimented with using SMOTE during the training of each model to see if it improved performance, particularly on the "Enrolled" minority class.

We found that over the three models when we applied SMOTE, the Enrolled class' F1 score increased. However, scores for the other classes decreased, with Naive Bayes taking the most dramatic hit. The score changes went as followed for each model:

Hypertuned Classification Algorithm	F1 Score Before SMOTE			F1 Score With SMOTE		
	Grad.	Enr.	Drop.	Grad.	Enr.	Drop.
Logistic regression	0.84	0.32	0.77	0.83	0.48	0.76
Naive Bayes (NOT tuned)	0.71	0.36	0.59	0.67	0.38	0.66
Support Vector Machine	0.84	0.35	0.77	0.82	0.44	0.74

As we can see, the effect of hypertuning and SMOTE had a positive effect on increasing the accuracy of the Enrolled class. This however came at a cost of lowering scores for other classes. For the sake of having a more balanced prediction, it may be more beneficial to choose the Logistic Regression with SMOTE as our best model. The F1 score for the Enrolled class increased the most while the other two classes didn't change dramatically. This experiment was worthwhile as we see that using a different resampling method like SMOTE helped increase model accuracy. Let's try one more thing to see if we increase classification predictability.

Ensemble Model:

As previously mentioned, each of the Logistic Regression, Naive Bayes, and SVM models displayed their own strengths and weaknesses. We thought that the best way to aggregate the advantages of each model would be to implement an ensemble model that averaged the predictions of each algorithm. For this objective, we used sci-kit learn's Voting Classifier model with the "soft voting" option. Soft voting enables the ensemble to make a decision by averaging each class's predicted probability among all of the constituent models.

The voting classifier accepts a weight parameter, useful for emphasizing the predictions of one constituent model over another. Using a grid search with 5-fold stratified cross-validation, we optimized the weight vector which resulted in an emphasis on the Logistic Regression and

SVM components with equal weights and a smaller weight on the Naive Bayes model. After defining a tuned voting classifier model and training on SMOTE-augmented data, it achieved the following results on the test set (equivalent results are displayed for the individual models for comparison purposes:

SMOTE-tuned models	Recall			Precision			F1-score			
	Grad.	Enr.	Drop.	Grad.	Enr.	Drop.	Grad.	Enr.	Drop.	Weighted
Voting Classifier	0.84	0.53	0.68	0.82	0.42	0.83	0.83	0.47	0.75	0.74
Log Reg.	0.85	0.52	0.70	0.82	0.44	0.82	0.83	0.48	0.76	0.74
SVM	0.85	0.48	0.67	0.80	0.41	0.82	0.82	0.44	0.74	0.73
NB	0.58	0.61	0.58	0.79	0.28	0.77	0.67	0.38	0.66	0.61

It can be seen that the results of the Voting Classifier differed only minutely from the individually-tuned Logistic Regression model.

Project Trajectory:

At first, we focused on testing baseline models and evaluating them using classification reports, paying attention to precision, recall, F1 score, and accuracy. However, a key challenge with this dataset was the class imbalance, clearly shown in the pie chart on page 2. Traditional classification metrics can be misleading under imbalance, as they often favor the majority class.

This issue became evident in our baseline results: the Graduate and Dropout classes performed well, while the Enrolled class was consistently misclassified. Recognizing this, we shifted our focus toward identifying models that improve performance across all classes, particularly the Enrolled class.

To better capture balanced performance, we adopted the weighted F1 score as our primary metric, as it accounts for both precision and recall while weighting by support. From here, we implemented hyperparameter tuning, SMOTE, and a voting classifier to address class imbalance more effectively. We then tracked F1 and weighted F1 scores to measure improvements over the baseline. Ultimately, we prioritized balanced performance across all classes over chasing a single high metric, leading to more meaningful model improvements.

Conclusion:

- **Summary:** In conclusion, we wanted to predict the student academic outcomes of either graduation, dropout, or enrollment by using financial, demographical, and academic performance data. After testing our models, the Logistic Regression model achieved the highest weighted F1-score of 0.74. We applied SMOTE to improve our model

performance by mitigating class imbalance to enhance the low recall for the small “Enrolled” class. We also tried to bolster the predictive power by ensembling Logistic Regression, the Support Vector Classifier, and the Gaussian Naive Bayes models into a soft voting classifier.

- **Strengths:** SMOTE and hyperparameter tuning led to a more balanced training set and a more reliable classification of the minority classes. Our models were able to predict “Graduated” and “Dropped” students with high precision and recall.
- **Short-comings:** Even our most highly-tuned models didn’t surpass a total accuracy of 75% on the test set. Even with considerations for target class imbalance like SMOTE, our classification models achieved less than 70% recall on the Dropout class and less than 60% recall for the Enrolled class. This means that a large number of at-risk students were not identified in the test set, and could potentially slip through the cracks.
- **Future work:** We could potentially address our project’s shortcomings by reframing the classification task. Rather than predicting the Enrolled and Dropout classes separately, we could combine them into a single class and treat it as a binary classification problem between “Graduate” and “Non-graduate”. If we accept the assumption that “Enrolled” students and “Dropout” students can benefit from the same extra assistance, then it might serve the objective better to combine these classes. In addition, if this framework were pursued, then the “Graduate” and “Non-graduate” target classes would have almost a completely even 50/50 split, which might encourage better predictive accuracy.

Resources:

Cramer's V Origination: Cramér, Harald. 1946. Mathematical Methods of Statistics. Princeton: Princeton University Press, page 282 (Chapter 21. The two-dimensional case).

Wikipedia Article with Cramer's V Formula: https://en.wikipedia.org/wiki/Cram%C3%A9r's_V

Medium Article going into using Cramer's V: https://medium.com/@manindersingh120996/understanding-categorical-correlations-with-chi-square-test-and-cramers-v-a54fe153b1d6

SelectKBest() documentation: https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.SelectKBest.html