**Raport**

**Author:** 280462 Bartosz Wacławiak

**Date:** 07.06.2025

**Course:** MSiD - laboratories

**Dataset:** [**Predict Student’s Dropout and Academic Success**](https://www.kaggle.com/datasets/naveenkumar20bps1137/predict-students-dropout-and-academic-success)

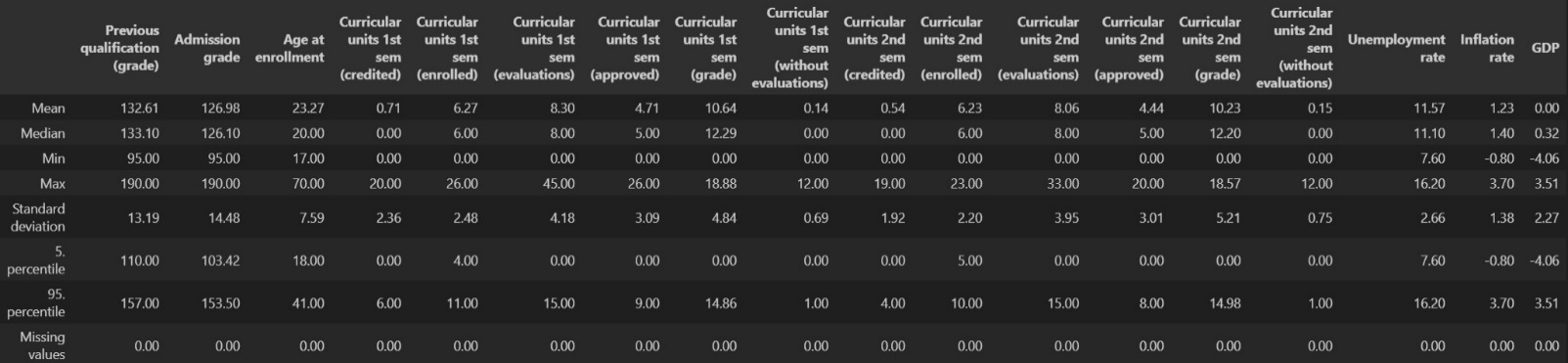
**Part 1**

This section presents a comprehensive visual analysis of student academic behavior and performance using a variety of statistical charts. The goal is to uncover meaningful patterns and relationships between student characteristics, academic outcomes, and external factors such as economic indicators.

The charts explore variables such as age at enrollment, admission grades, course performance, financial status (debtor or scholarship holder), and macroeconomic context (GDP, unemployment, inflation). By analyzing these factors across different student groups (dropouts, graduates, and currently enrolled), we aim to identify trends that may explain academic success or failure and provide insights for better educational planning and policy-making.

Each visual is accompanied by a concise set of observations that highlight key findings and support data-driven interpretation.

**Feature statistics**

**Numeric features analysis**

**Most common categories**

* Application mode -> 1st phase - general contingent (38.61%)
* Application order -> 2nd choice (68.40%)
* Course -> Nursing (17.31%)
* Daytime evening attendance -> Daytime (89.08%)
* Debtor -> No (88.63%)
* Displaced -> Yes (54.84%)
* Educational special needs -> No (98.85%)
* Father's occupation -> Unskilled Workers (22.83%)
* Father's qualification -> Basic Ed 1st Cycle (4th/5th) (27.33%)
* Gender -> Female (64.83%)
* International -> No (97.51%)
* Matrimonial status -> single (88.58%)
* Mother's occupation -> Unskilled Workers (35.65%)
* Mother's qualification -> Secondary Education (24.16%)
* Nationality -> Portuguese (97.51%)
* Previous qualification -> Secondary education (84.02%)
* Scholarship holder -> No (75.16%)
* Target -> Graduate (49.93%)
* Tuition fees up to date -> Yes (88.07%)

**Chart types**

**Heatmap**

A heatmap is a graphical representation of data where individual values are represented by colors. In the context of correlation analysis, a heatmap visually displays the strength and direction of relationships between multiple variables. Darker or more intense colors indicate stronger correlations (positive or negative), making it easy to identify patterns and dependencies in the dataset.

**PCA**

PCA (Principal Component Analysis) is a dimensionality reduction technique used to simplify large datasets while preserving as much variance as possible. It transforms the original features into a new set of uncorrelated variables called principal components. These components help visualize complex data in two or three dimensions and are useful for detecting structure, clusters, or outliers.

**Histogram**

A histogram is a type of bar chart that shows the distribution of a numerical variable by dividing the data into intervals (called bins) and counting how many values fall into each interval. It provides a visual overview of the shape, spread, and central tendency of the data, helping to identify patterns.

**Error plot**

An error plot is a visualization that shows the variability or uncertainty around data points, often using vertical or horizontal bars to represent error margins. These may reflect standard deviation, confidence intervals, or standard error, helping to interpret the reliability and precision of the displayed values.

**Box plot**

A box plot (also known as a box-and-whisker plot) is used to summarize the distribution of a dataset based on five key statistics: minimum, first quartile (Q1), median, third quartile (Q3), and maximum. It highlights central tendency, variability, and potential outliers, making it easy to compare distributions across different groups.

**Violin plot**

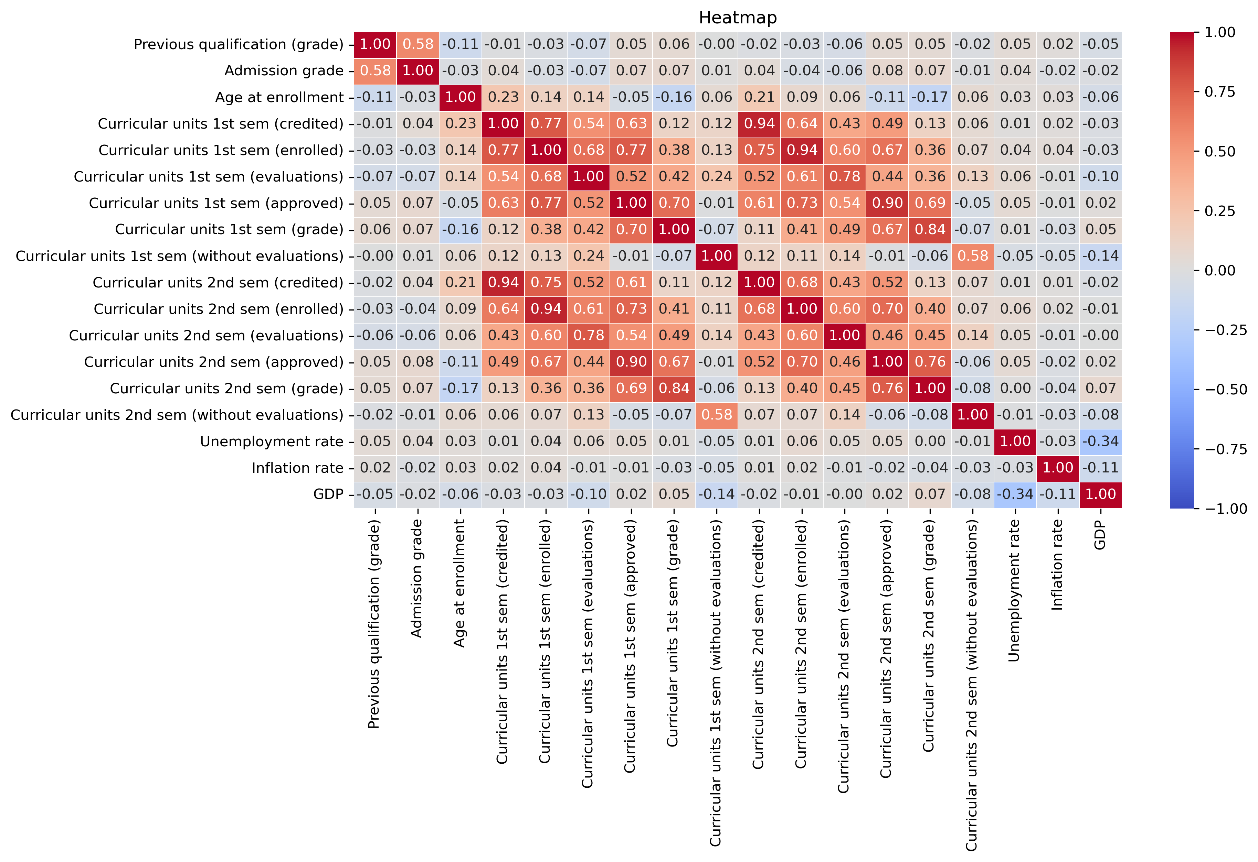
A violin plot is a combination of a box plot and a kernel density plot. It displays the distribution, probability density, and summary statistics of a dataset. The width of the violin shape shows the frequency of data points at different values, while the internal box plot provides information on median and quartiles. Violin plots are useful for comparing multiple groups and visualizing the shape of the data distribution.

**Regression plot**

A regression plot is a visualization that shows the relationship between two variables along with a fitted regression line, which models the trend in the data. It often includes a confidence interval around the line to indicate the uncertainty of the estimate. Regression plots are useful for identifying linear or nonlinear trends, correlations, and the strength of associations between variables.

**Data overview**

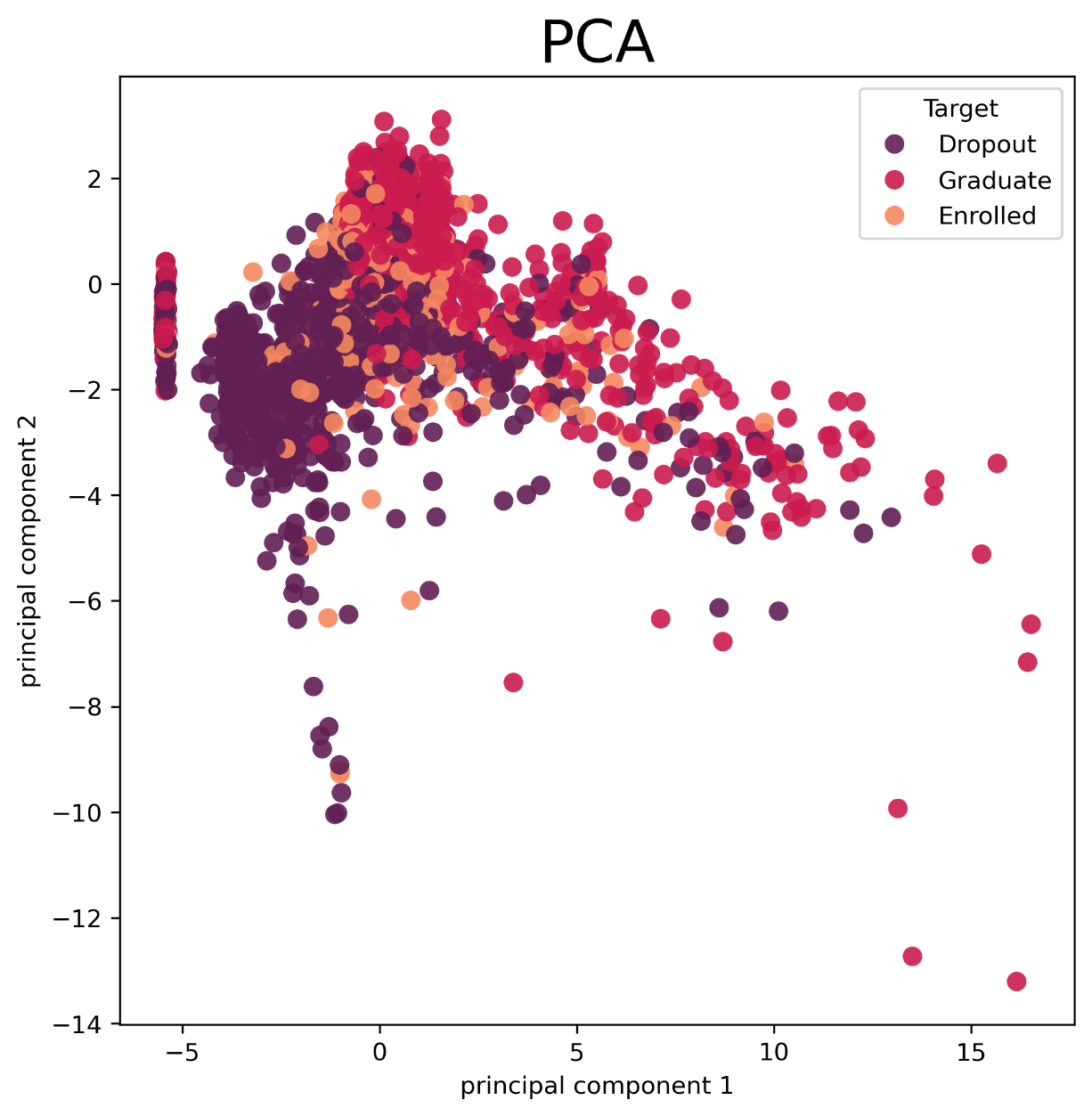
**Heatmap**

****

Observations

* The parameters related to curricular units in the 1st semester and in the 2nd semester (credited, enrolled, evaluations, approved, grade) show strong mutual correlation, which is expected since they describe related aspects of academic performance.
* The is a strong positive correlation between admission grade and previous qualification grade (0.58). This suggests consistency in academic performance before and during university admission.
* A slight negative correlation is observed between age at enrollment and academic performance indicators, particularly with grades. This may indicate that younger students tend to perform slightly better.
* A clear negative correlation between GDP and unemployment rate is observed (−0.75), reflecting a typical economic relationship where higher GDP often aligns with lower unemployment.
* The **inflation rate does not show significant correlation** with either GDP or unemployment rate, which is somewhat **unexpected** and may require further investigation.

**PCA**

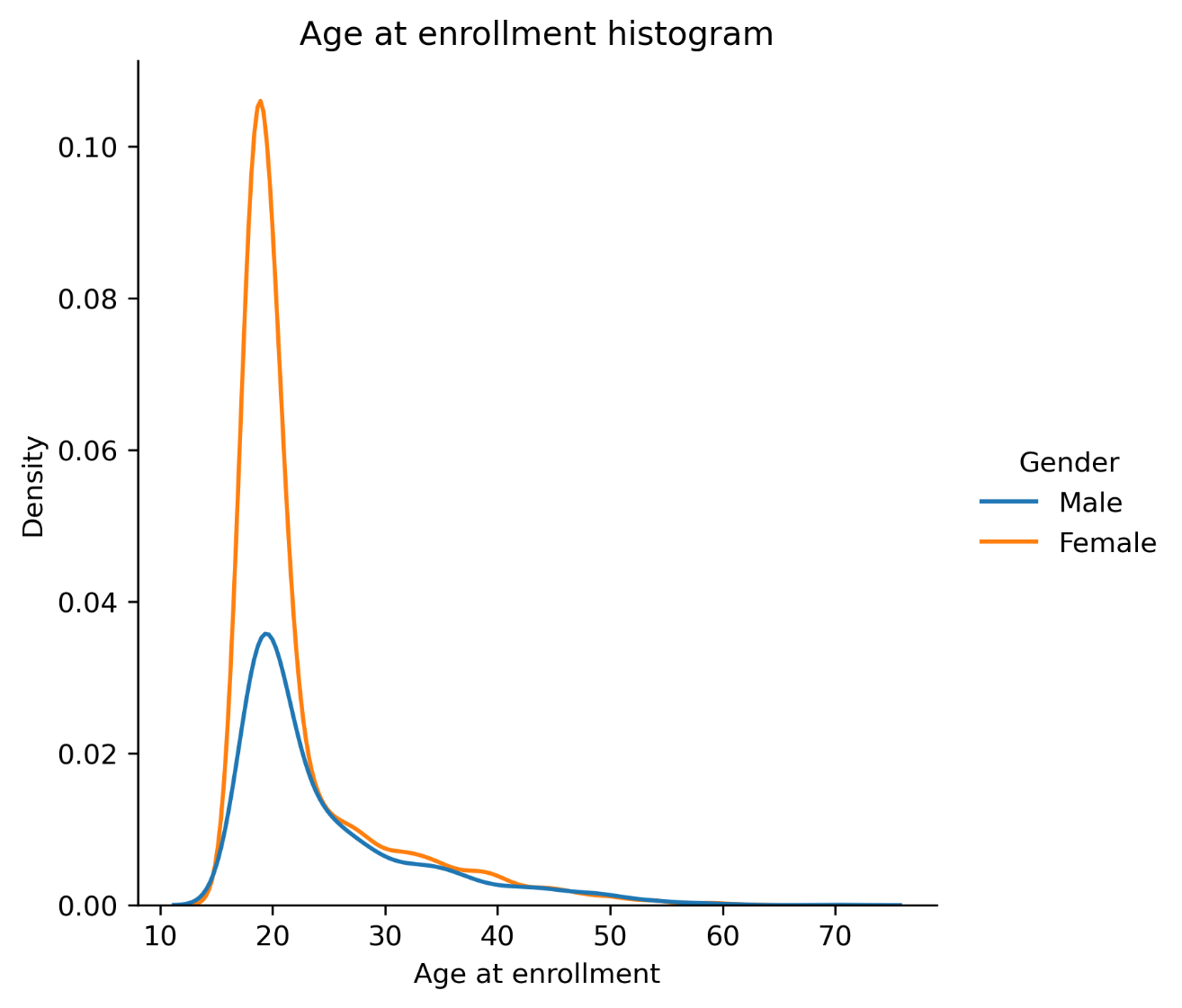
****

Observations

* The dropout students category is primarily located on the left side of the chart, indicating a clustering pattern along lower values of the first principal component.
* The graduate students category tends to be distributed toward the upper regions of the chart, showing some separation from other categories in terms of both principal components.
* The enrolled students category is more dispersed across the plot but is often positioned between the dropout and graduate student clusters, reflecting transitional characteristics.

**Correlations**

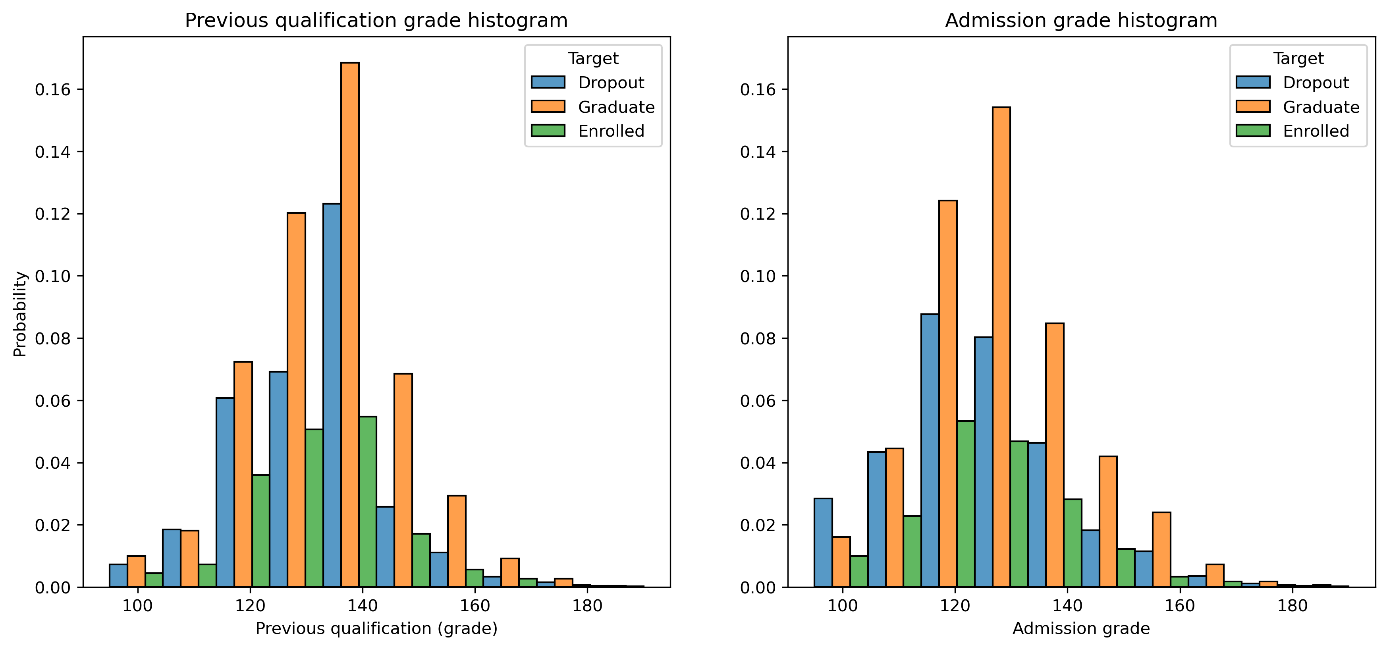
**Gender**

****

Observations

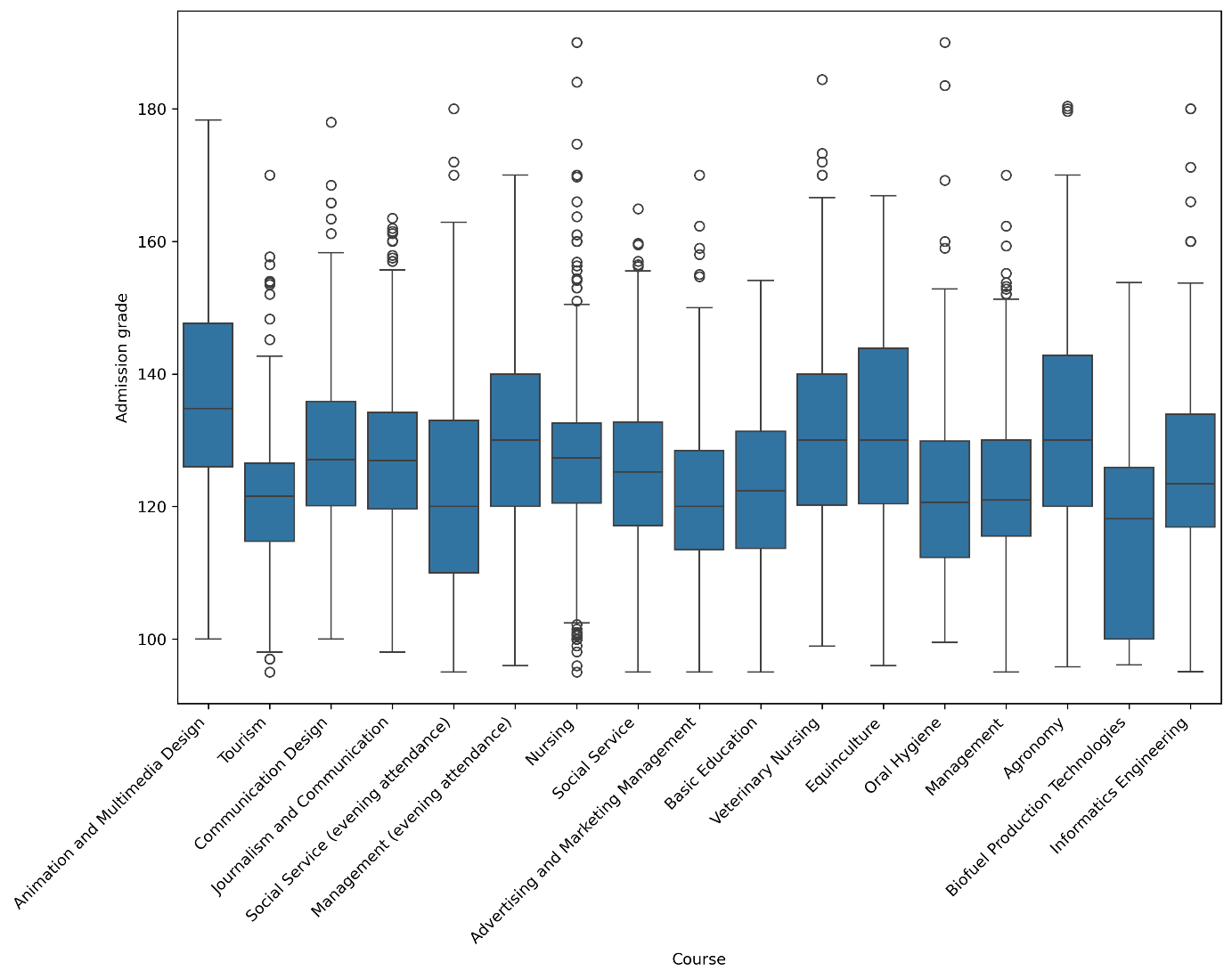
* The majority of students, regardless of gender, enroll around the age of 18 to 20, as shown by the sharp peak in that range.
* The distribution for female students is higher in density than for male students, suggesting there are more female students in the dataset.
* Both male and female age distributions are right-skewed, meaning there are fewer older students, but a small number do enroll later in life—even beyond age 50.
* After the main peak, the number of enrollments steadily declines for both genders, though the distributions remain somewhat similar in shape.

**Previous qualification and admission grades**

****

Observations

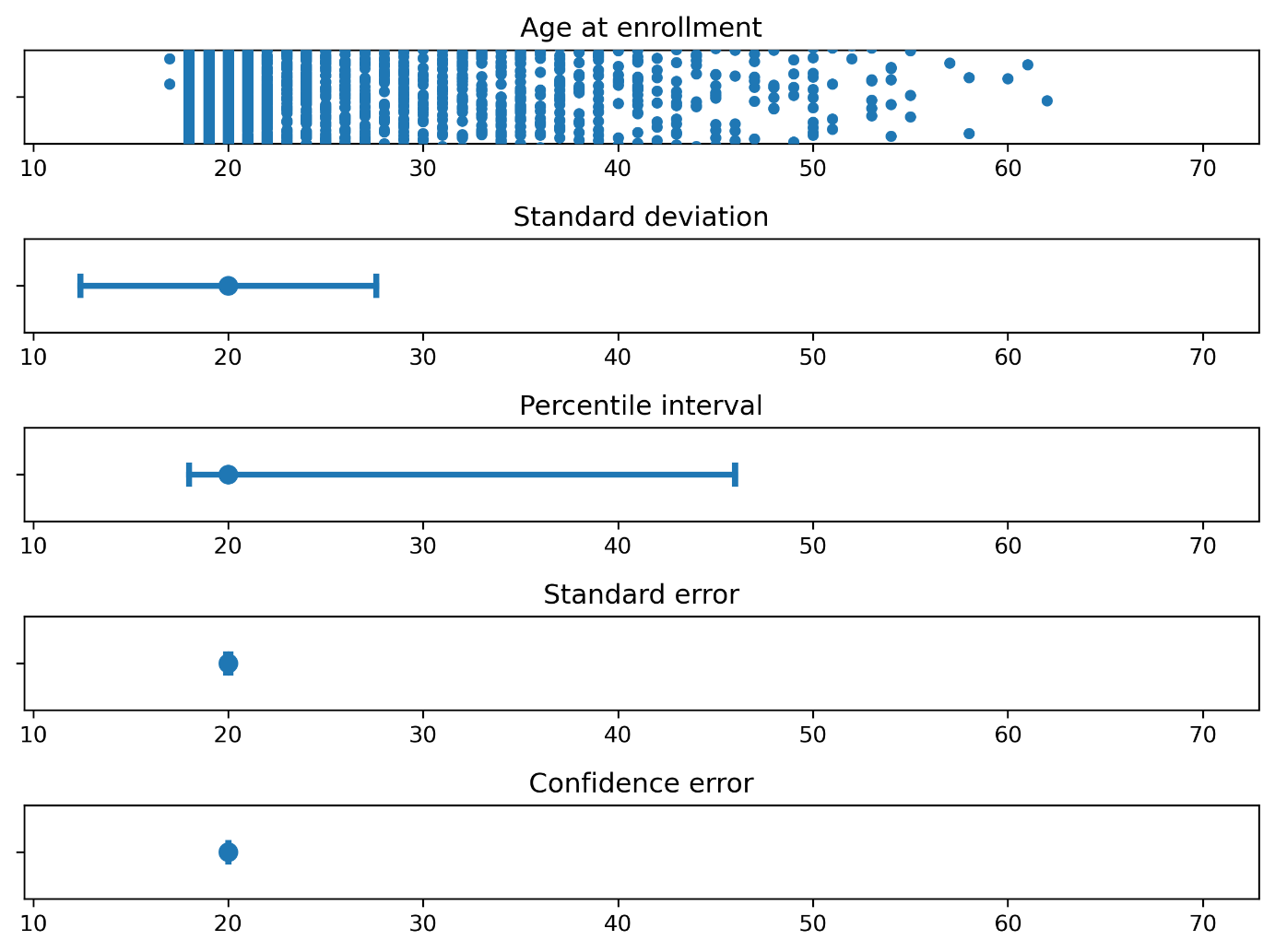
* The distribution of both previous qualification grades and admission grades follows a bell-shaped curve, with most students scoring between 120 and 140.
* Dropout students (blue bars) are more present in the lower grade ranges, particularly around 110–130, which may suggest a link between lower grades and higher dropout rates.
* Overall, students with higher previous and admission grades are more likely to graduate, indicating a potential predictive relationship between early academic performance and university outcomes.



Observations

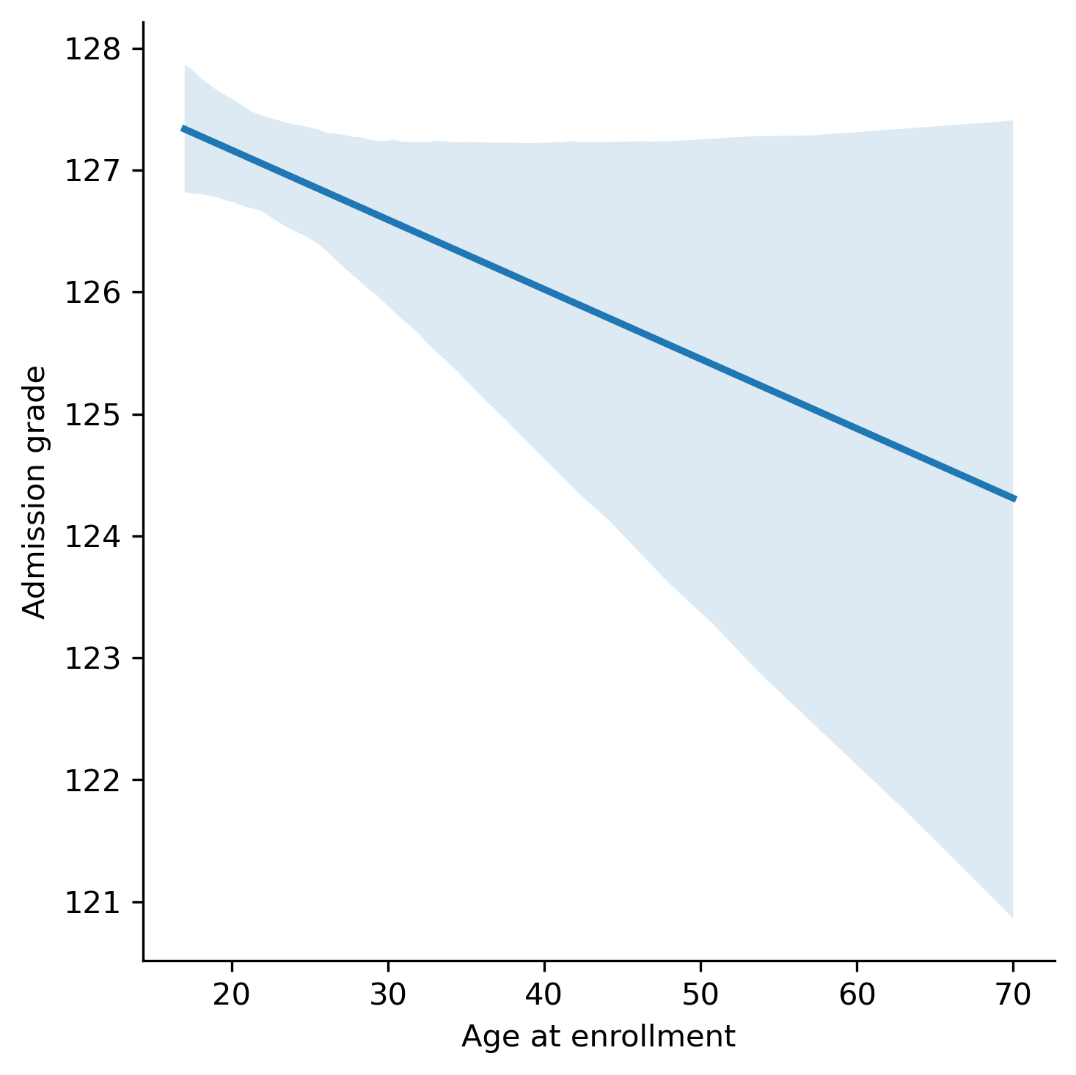
* Students enrolled in the Animation and Multimedia Design course have the highest admission grades on average, with relatively few low-performing outliers.
* In contrast, students in Biofuel Production Technologies tend to have lower admission grades, with a more compressed distribution and a lower median compared to other courses.
* Several courses, such as Tourism, Journalism and Communication, and Management, show wider variability in admission grades, indicating more diverse student performance at entry.
* Courses like Oral Hygiene and Equiniculture display relatively high median scores and narrower interquartile ranges, suggesting more consistent performance among admitted students.
* The presence of outliers across many courses indicates that while most students fall within a typical range, there are exceptional cases of both high and low performance across all programs.

**Age at enrollement**



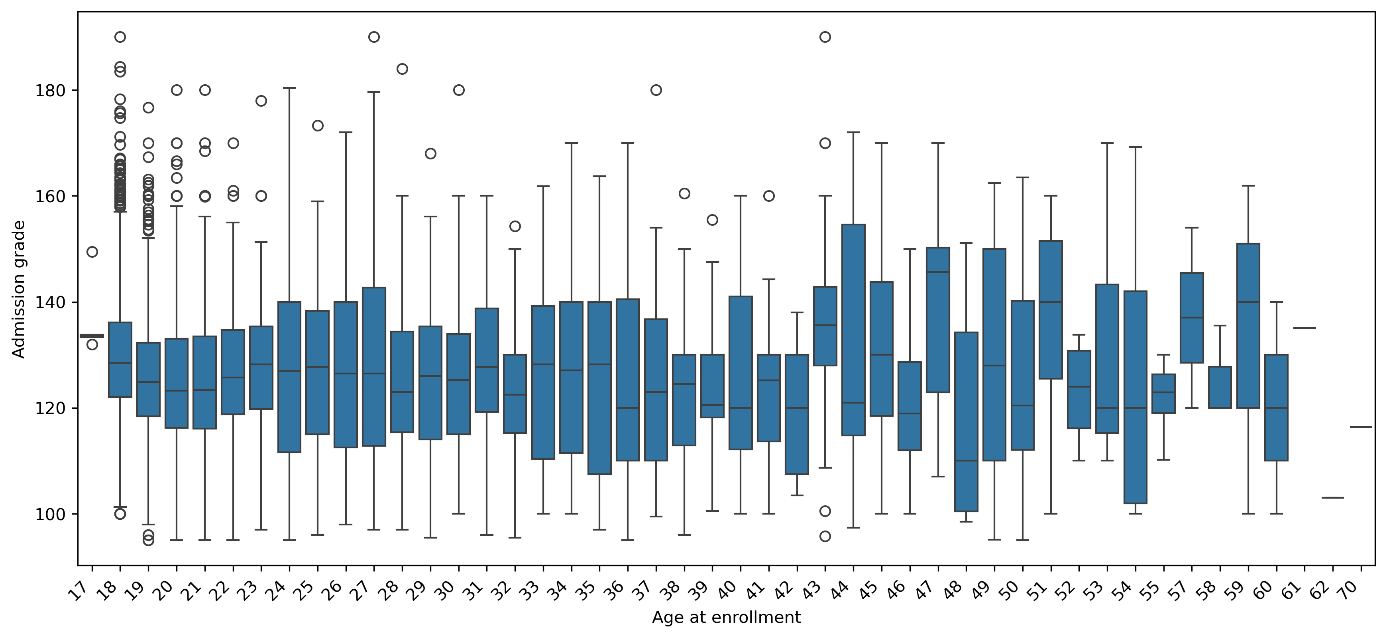
Obsevations

* Most students enroll between the ages of 18 and 25, as shown in the dot plot and supported by all statistical intervals (standard deviation, percentile, and error bounds).
* The standard deviation and percentile interval indicate that the bulk of enrollment ages fall roughly between 18 and 45, with some outliers extending to age 70.



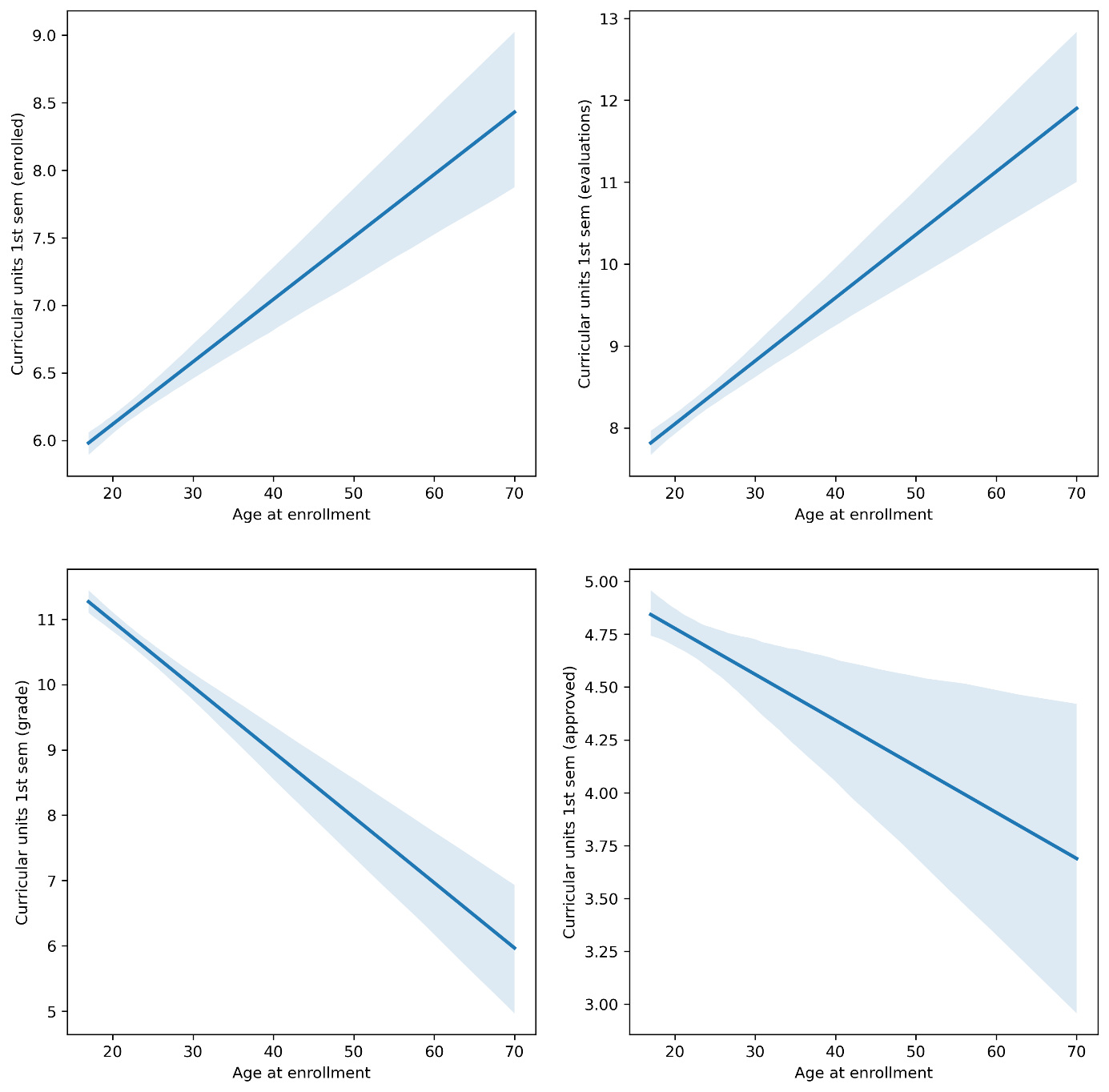
Observations

* The regression plot reveals a slight negative correlation between age at enrollment and admission grade — suggesting that students who enroll at an older age tend to have slightly lower admission grades on average.
* However, the confidence interval around the regression line widens significantly for older students, indicating less certainty in the trend due to sparser data in those age ranges.



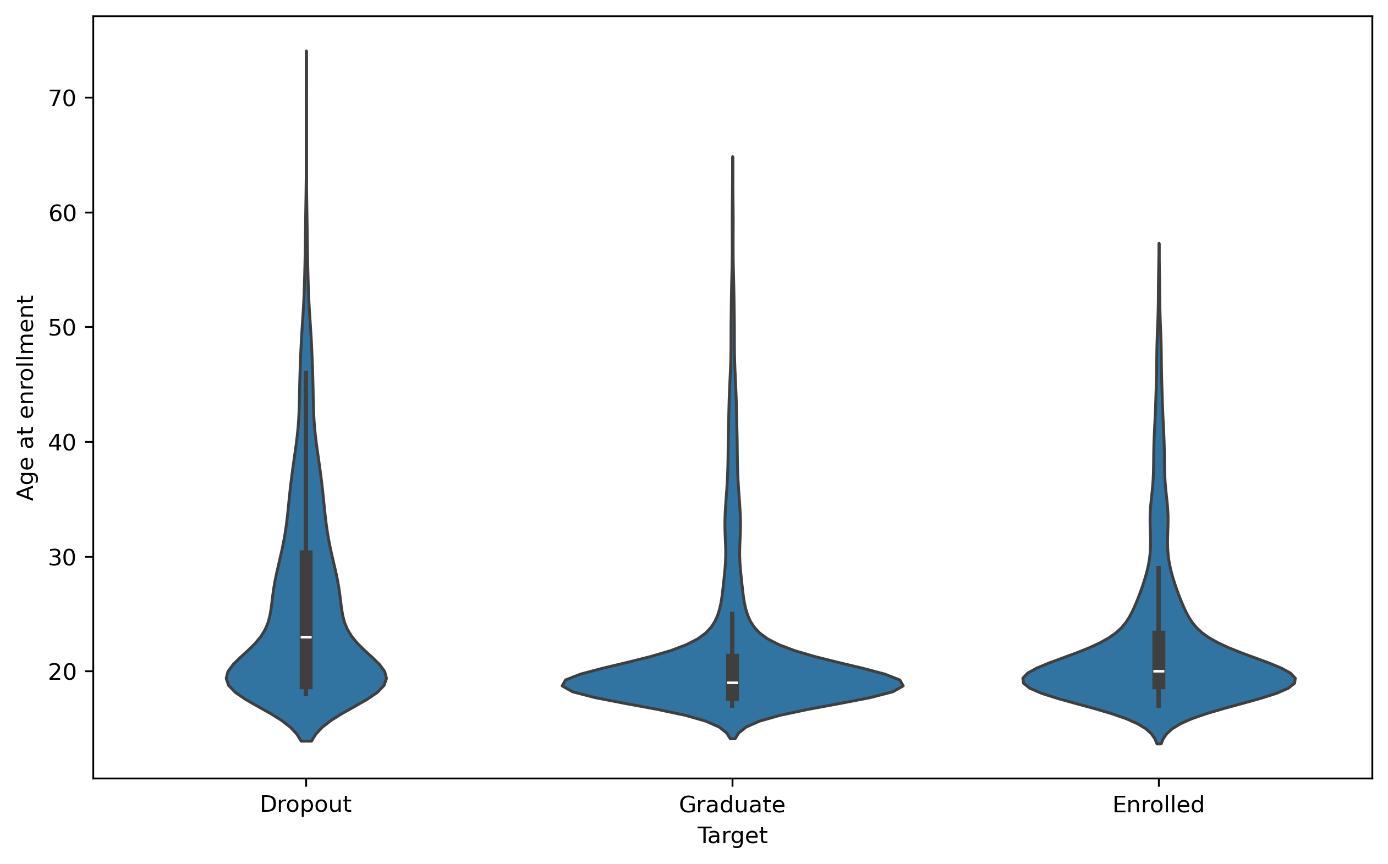
Observations

* The boxplot of admission grade by age supports this trend: younger students (especially those around age 18–21) tend to have higher median grades, while older age groups show more variability and often lower medians.
* Overall, while the trend suggests older students perform slightly worse in admission scores, the effect may be partially due to the smaller sample size for non-traditional-age students.



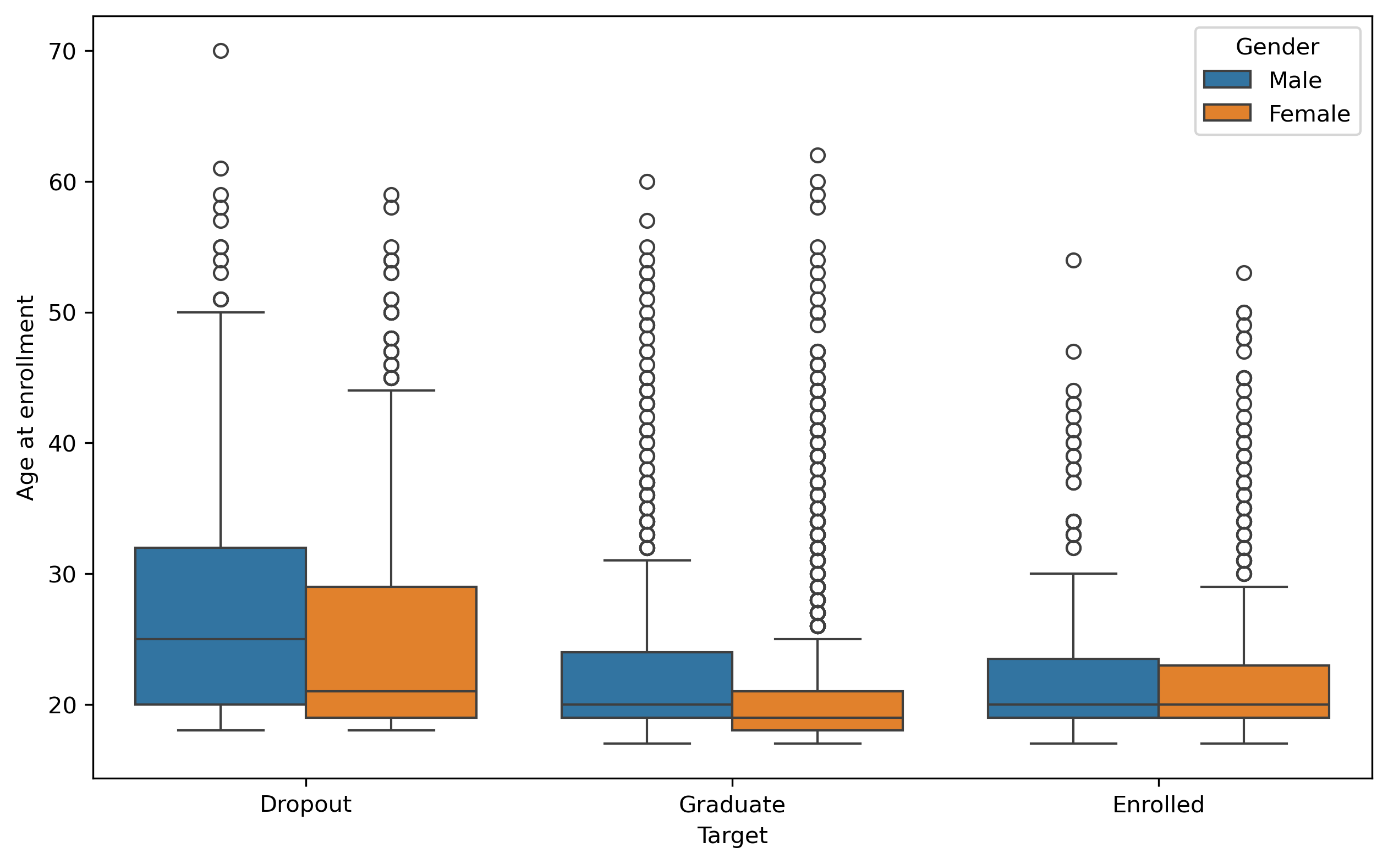
Observations

* Although older students tend to enroll in more classes and complete more evaluations, this does not translate into better outcomes, as both their average grades and number of approved units tend to decrease with age.
* The increasing number of evaluations taken by older students may indicate greater effort or course load, but the decreasing success rate suggests they may struggle more with academic performance, possibly due to external responsibilities or time away from formal education.



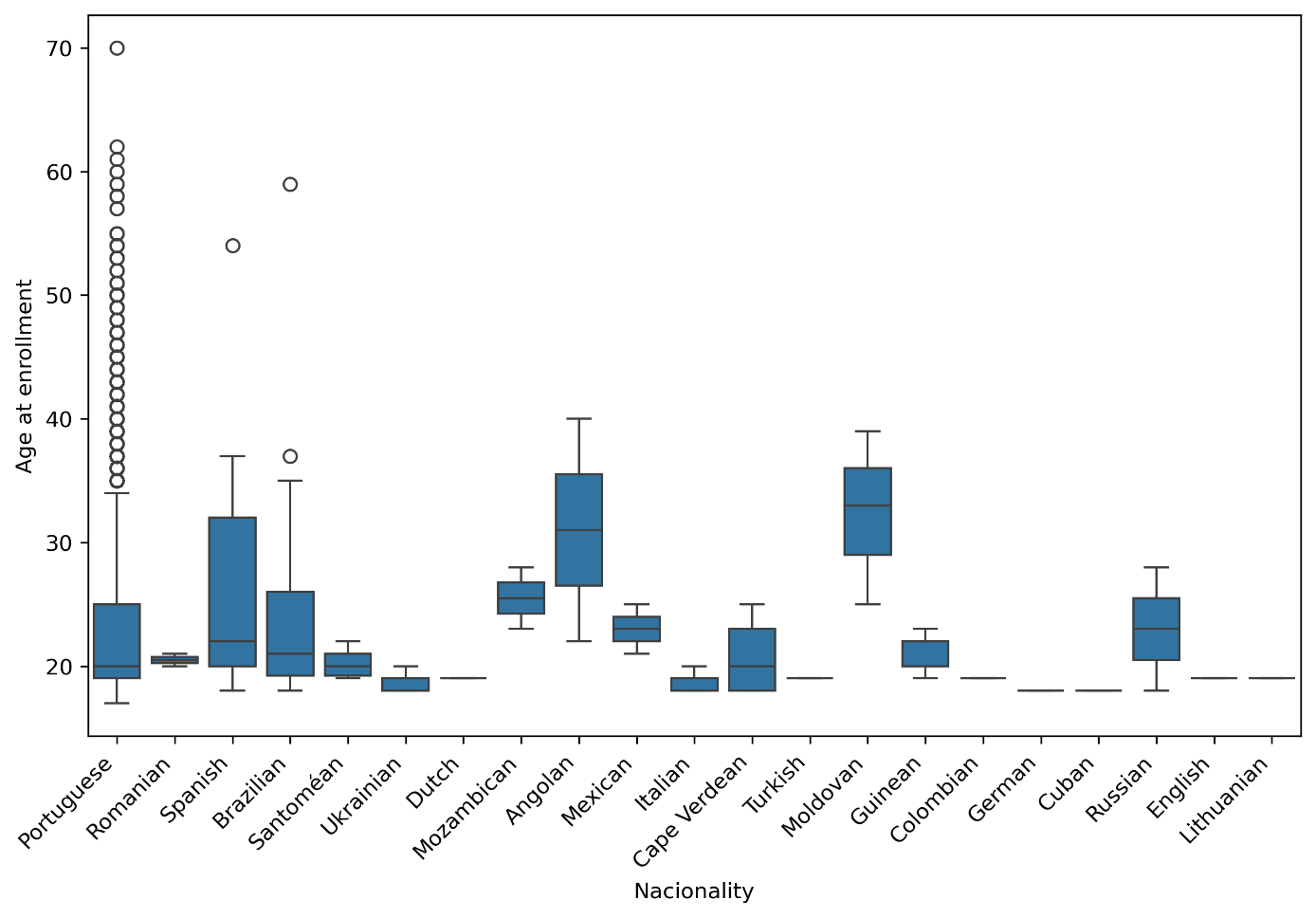
Observations

* Students who drop out tend to have a wider age distribution, including a significant number of older students, compared to those who graduate or remain enrolled.
* The graduate and enrolled groups show a much narrower age distribution, centered around 18–22 years, suggesting that younger students are more likely to succeed or persist in their studies.
* The increased presence of older students among dropouts supports the earlier findings that age may negatively affect academic outcomes.



Observations

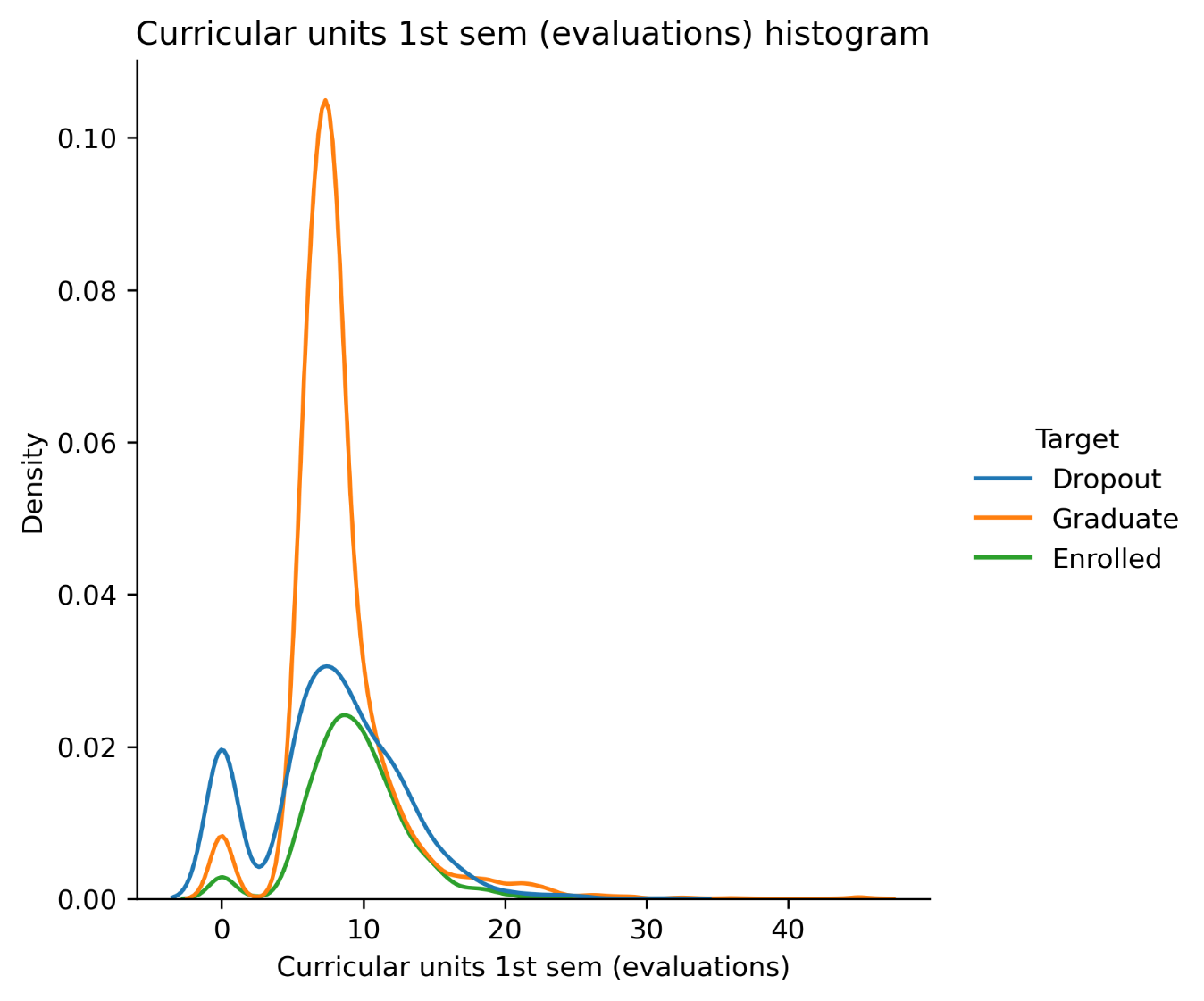
* Among students who drop out, males are on average older than females, and also show a wider spread of ages, including more older outliers.
* In the graduate and enrolled groups, both genders tend to enroll at a younger and more similar age, with distributions mostly centered around the early 20s.
* The age gap between genders is most noticeable among dropouts, suggesting that older male students may be at higher risk of not completing their studies.

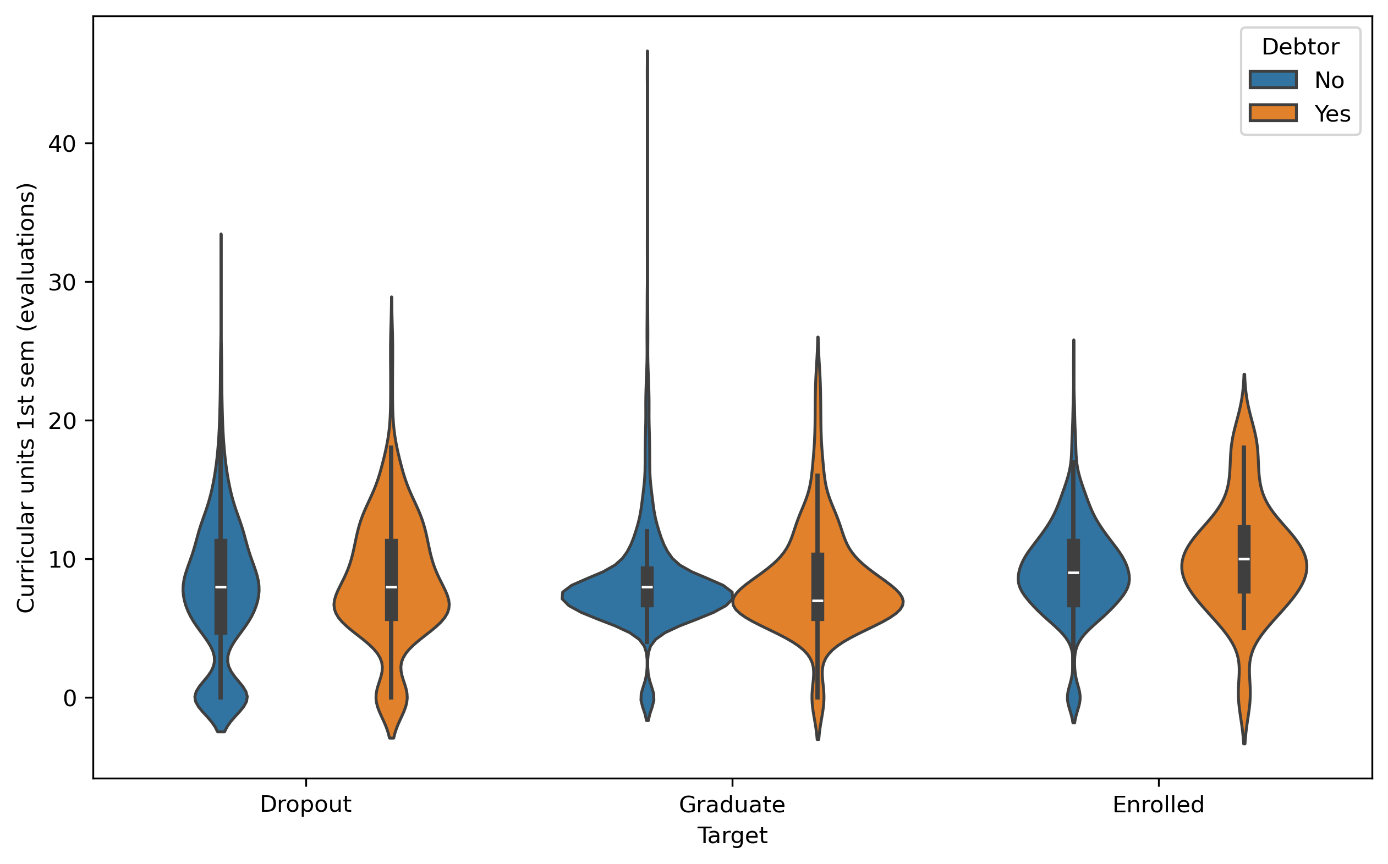


Observations

* Portuguese students show the widest range in age at enrollment, including the largest number of middle-aged and elderly students.
* Students from Brazil, Spain, Angola, and Moldova tend to enroll at older ages, with many having median enrollment ages in the late 20s or early 30s.
* In contrast, nationalities such as Romanian, Santomean, Ukrainian, and Cape Verdean display more compact age distributions, typically centered around early adulthood (18–22 years).
* These patterns suggest potential cultural, economic, or immigration-related factors influencing the timing of higher education enrollment across different nationalities.

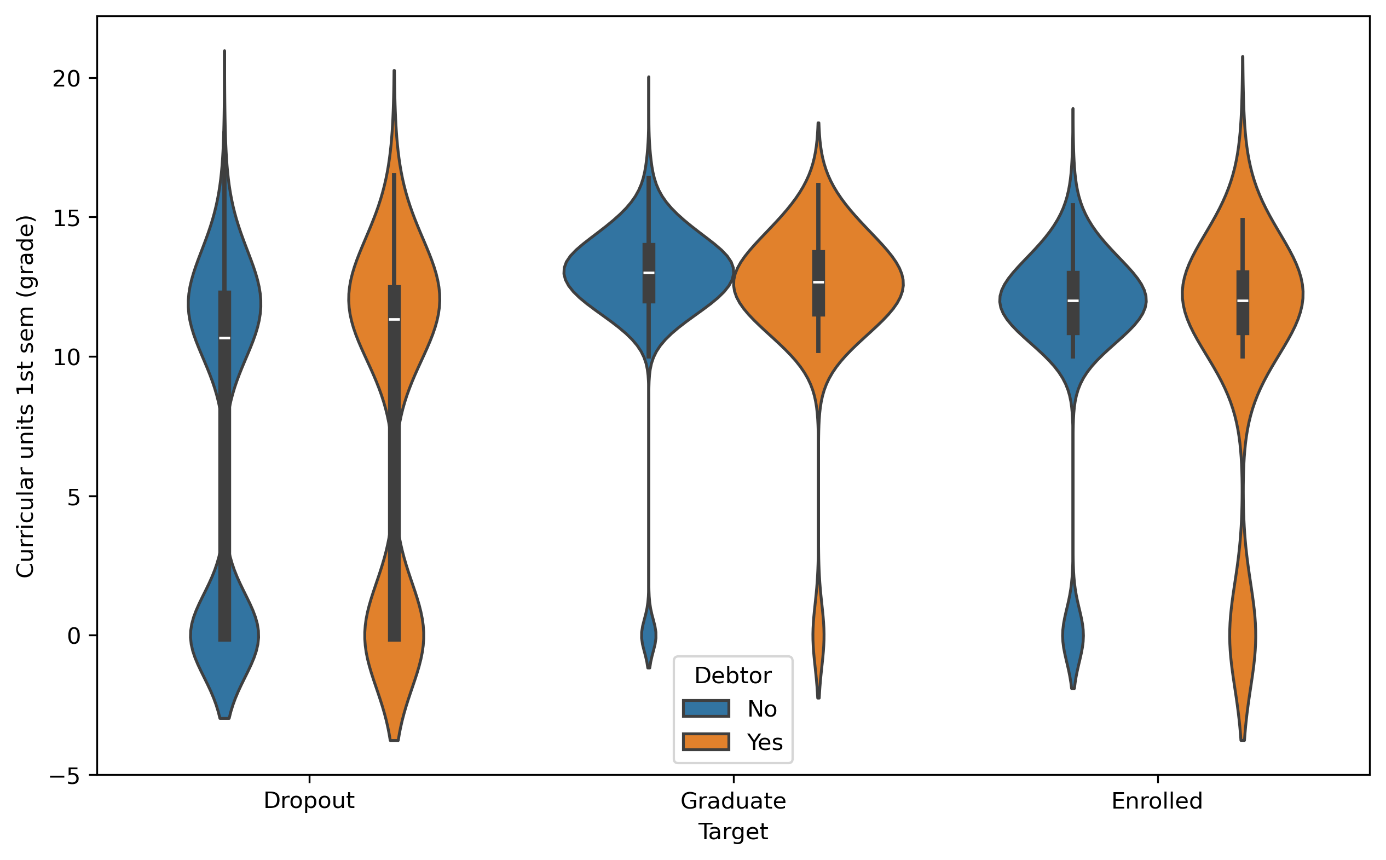
**1st semester**

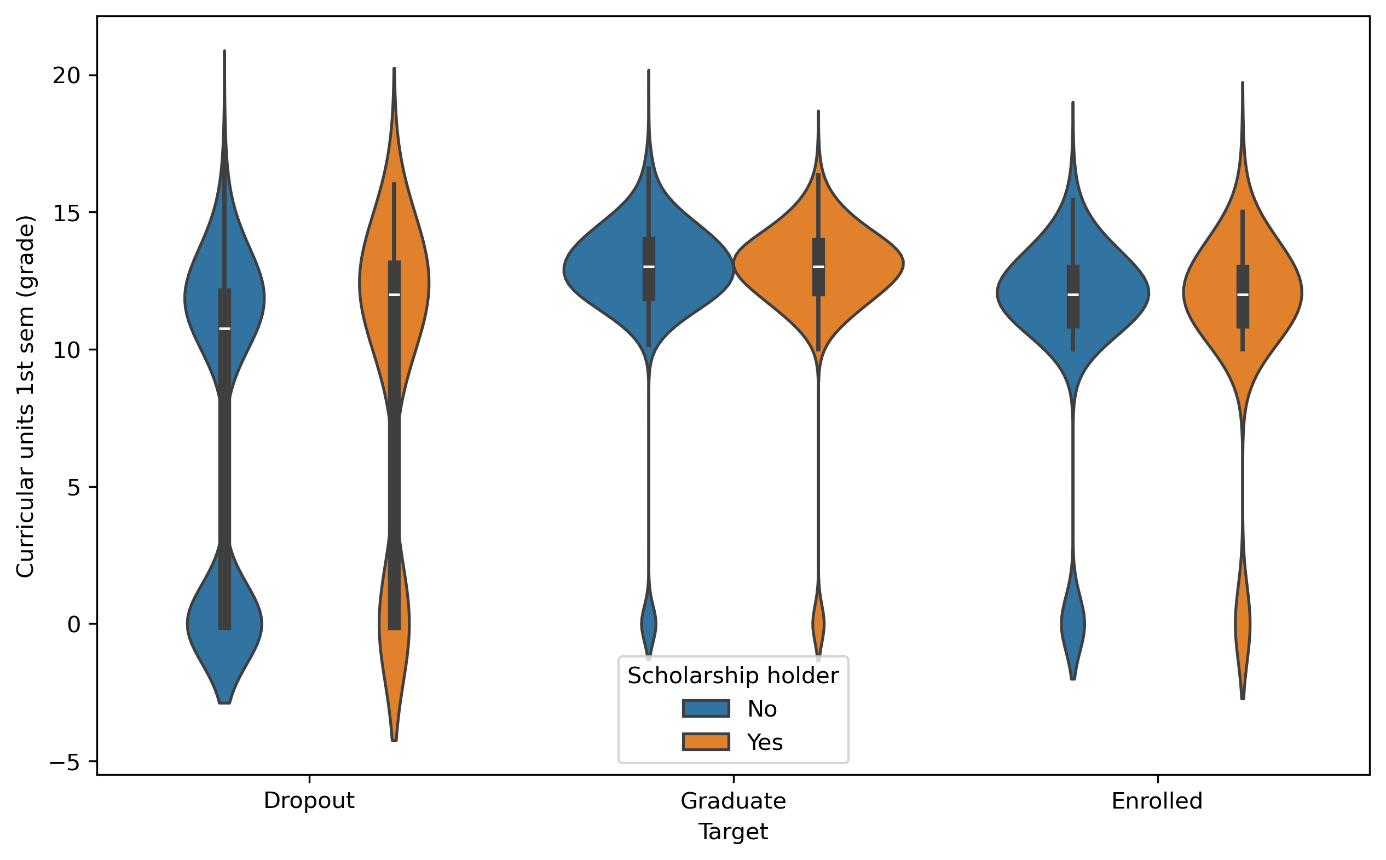
****

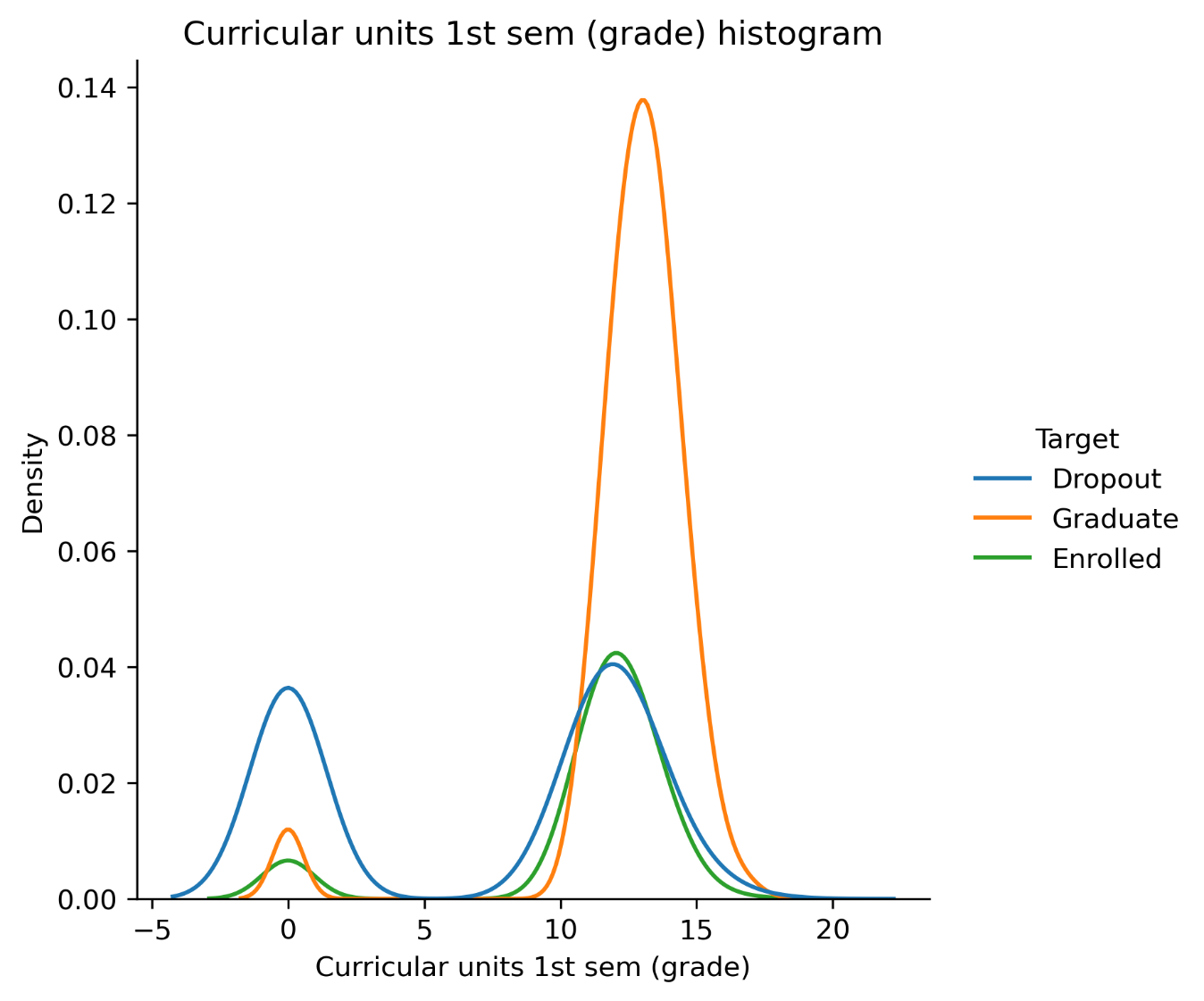
****

Observations

* Dropout students often have zero or very low numbers of evaluations during the first semester, indicating disengagement early in the academic process.
* The histogram shows a sharp peak around 7–9 evaluations for graduate students, suggesting they typically complete most of the expected assessments.
* Debtor students (those with financial or academic liabilities) tend to attempt more evaluations on average compared to non-debtors, possibly reflecting pressure to prove academic progress or fulfill specific requirements.

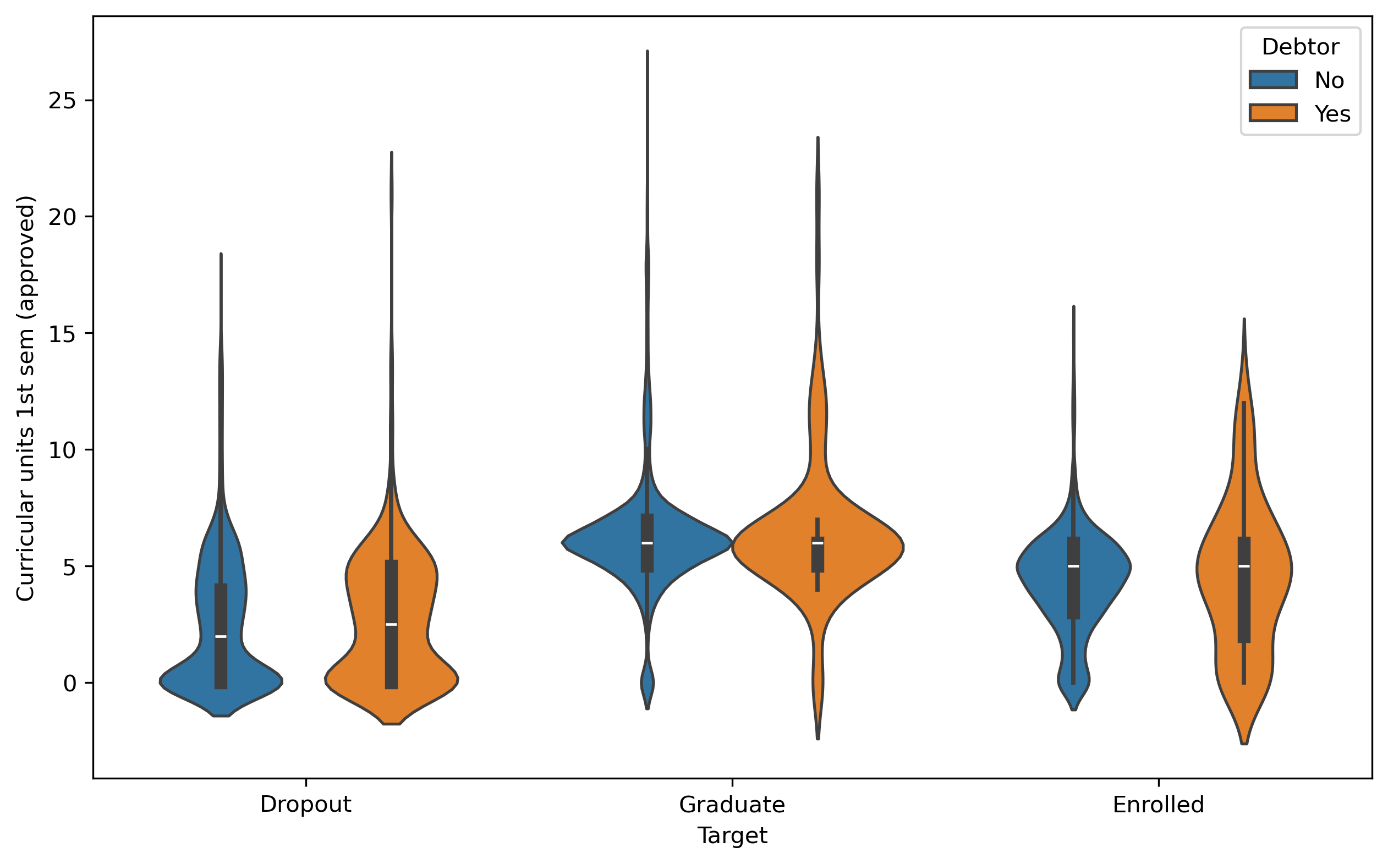


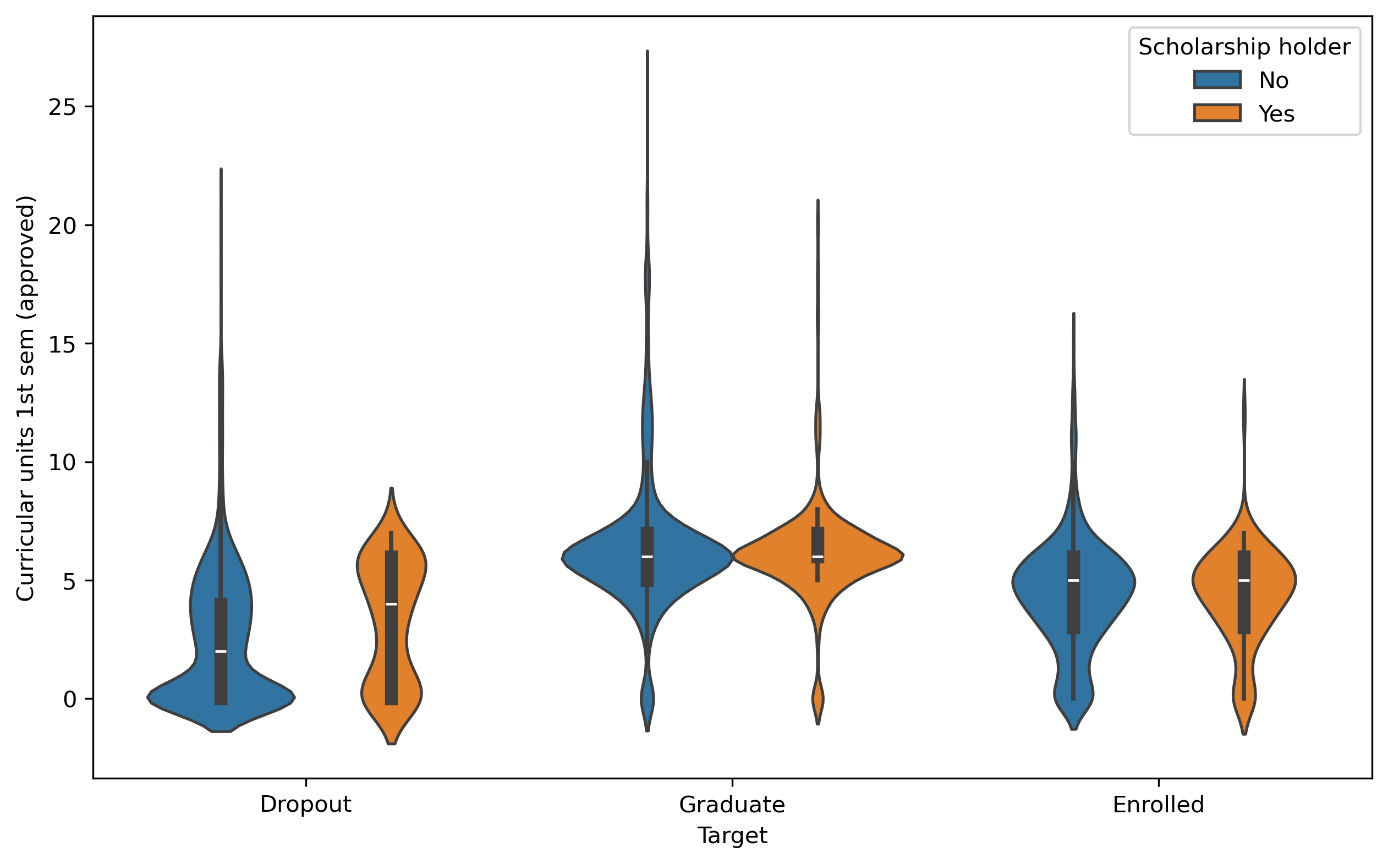


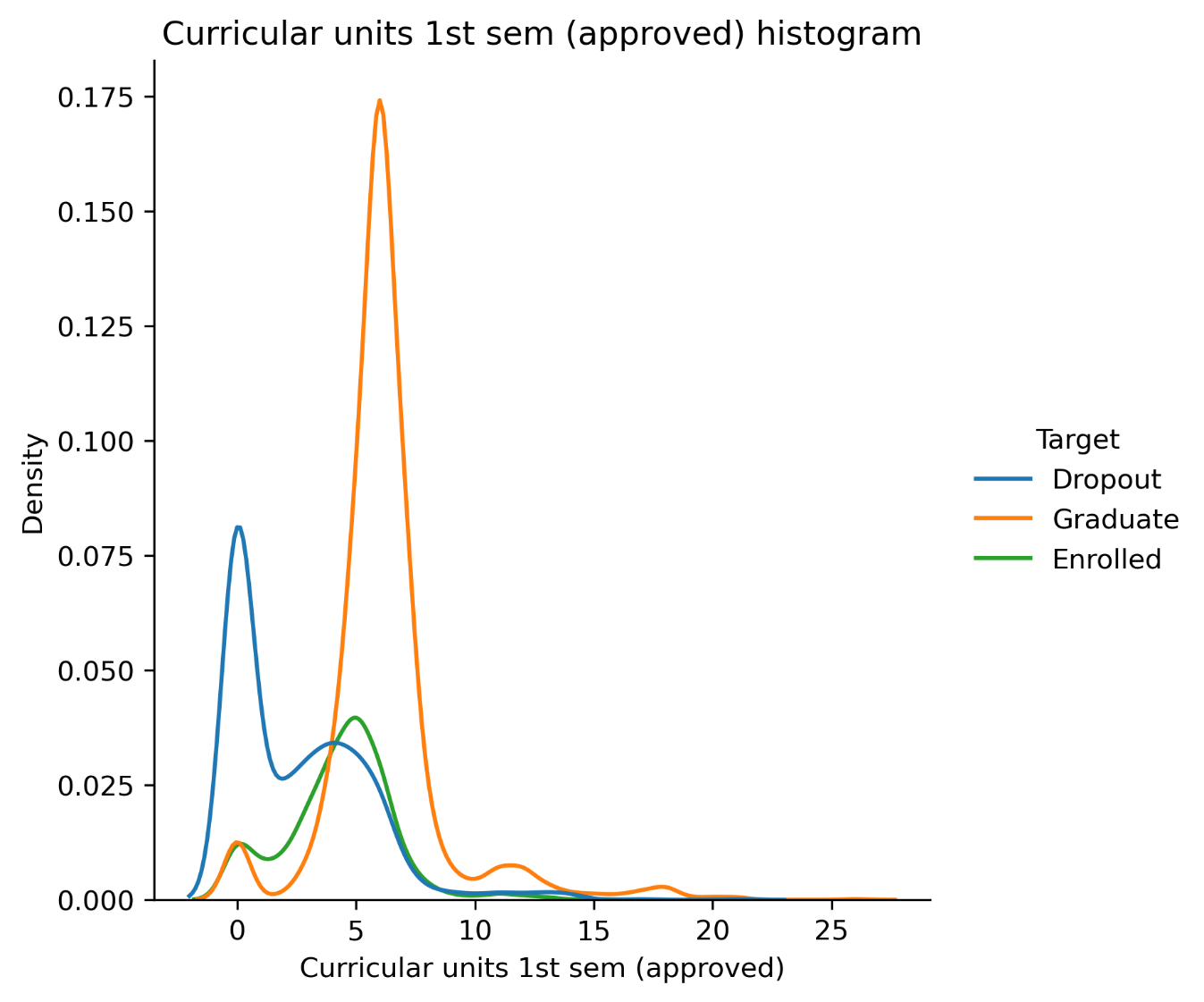


Observations

* A significant number of dropout students receive extremely low grades in their first semester, often close to zero, as clearly seen in the histogram.
* In contrast, graduate students tend to have high and consistent grades, typically centered around 13–14, with much lower variance.
* Debtor students and scholarship holders show more consistent and higher grades than their counterparts, especially among graduates and enrolled students. This may indicate that having financial or academic responsibility increases motivation or accountability.
* Among dropouts, students who were neither debtors nor scholarship holders tend to perform the worst — suggesting that external motivation may play a critical role in early academic success.
* Grade distributions for enrolled students are generally between those of dropouts and graduates, showing moderate performance and variation.



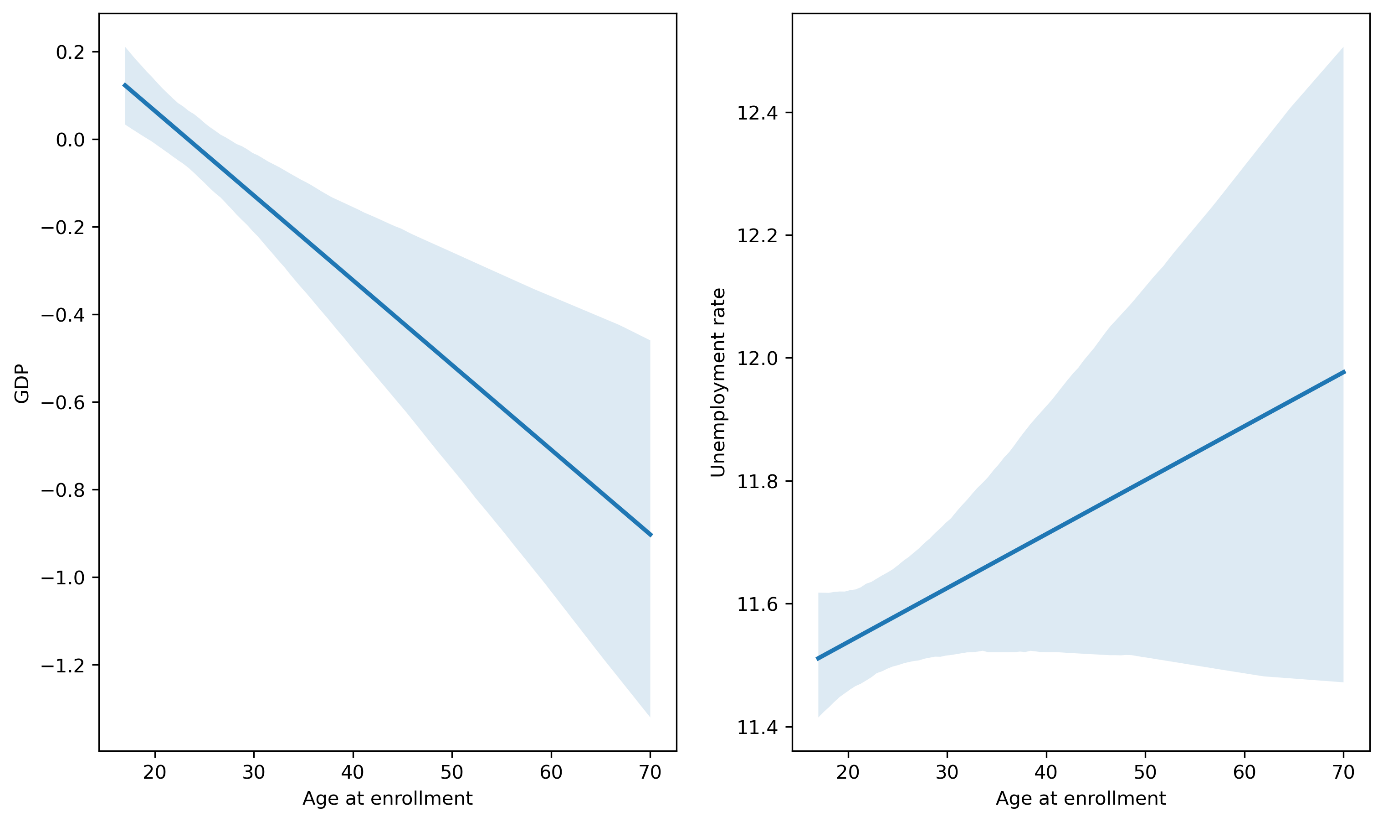




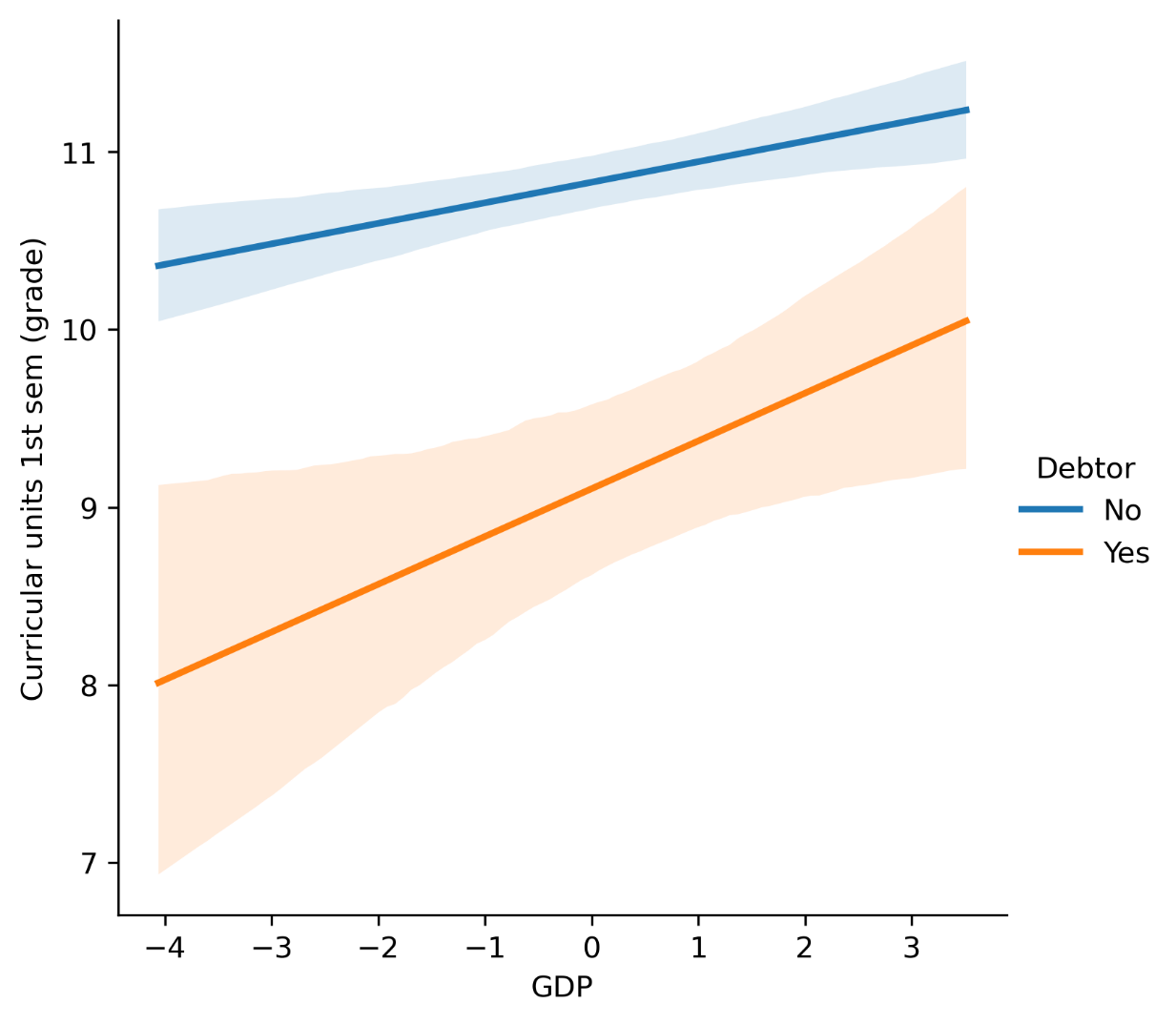
Observations

* Dropout students typically pass very few units in the first semester—most often between 0 and 2—highlighting a critical early academic struggle.
* The histogram confirms that graduates consistently pass the most units, typically around 6 or more, while enrolled students fall in between dropouts and graduates.
* Among dropouts, scholarship holders tend to perform slightly better than those without scholarships, indicating that financial support may buffer against failure.
* Debt status appears to have minimal impact on approval rates during the 1st semester, as distributions between debtor and non-debtor students are similar across all outcomes.

**Economic**

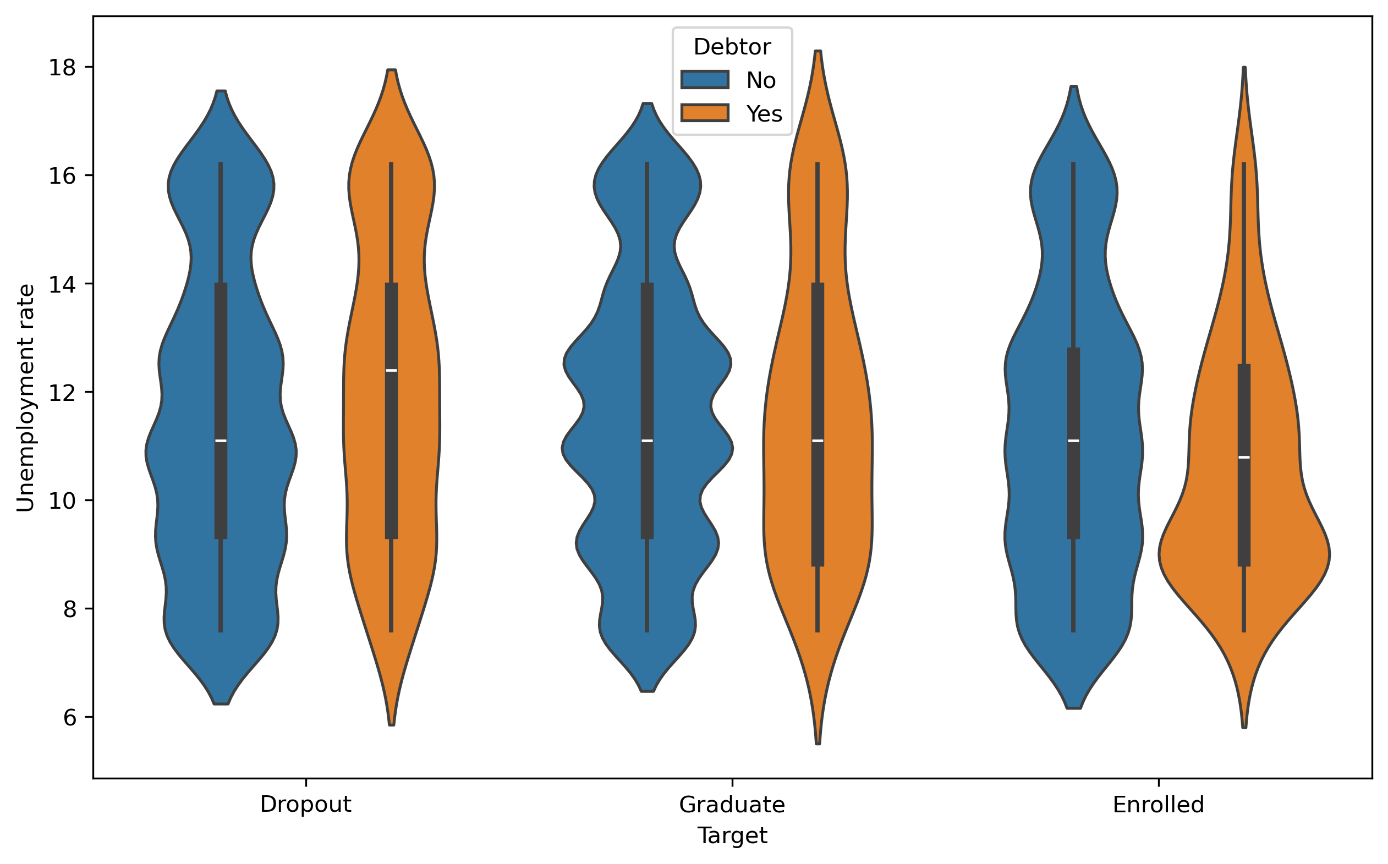
****

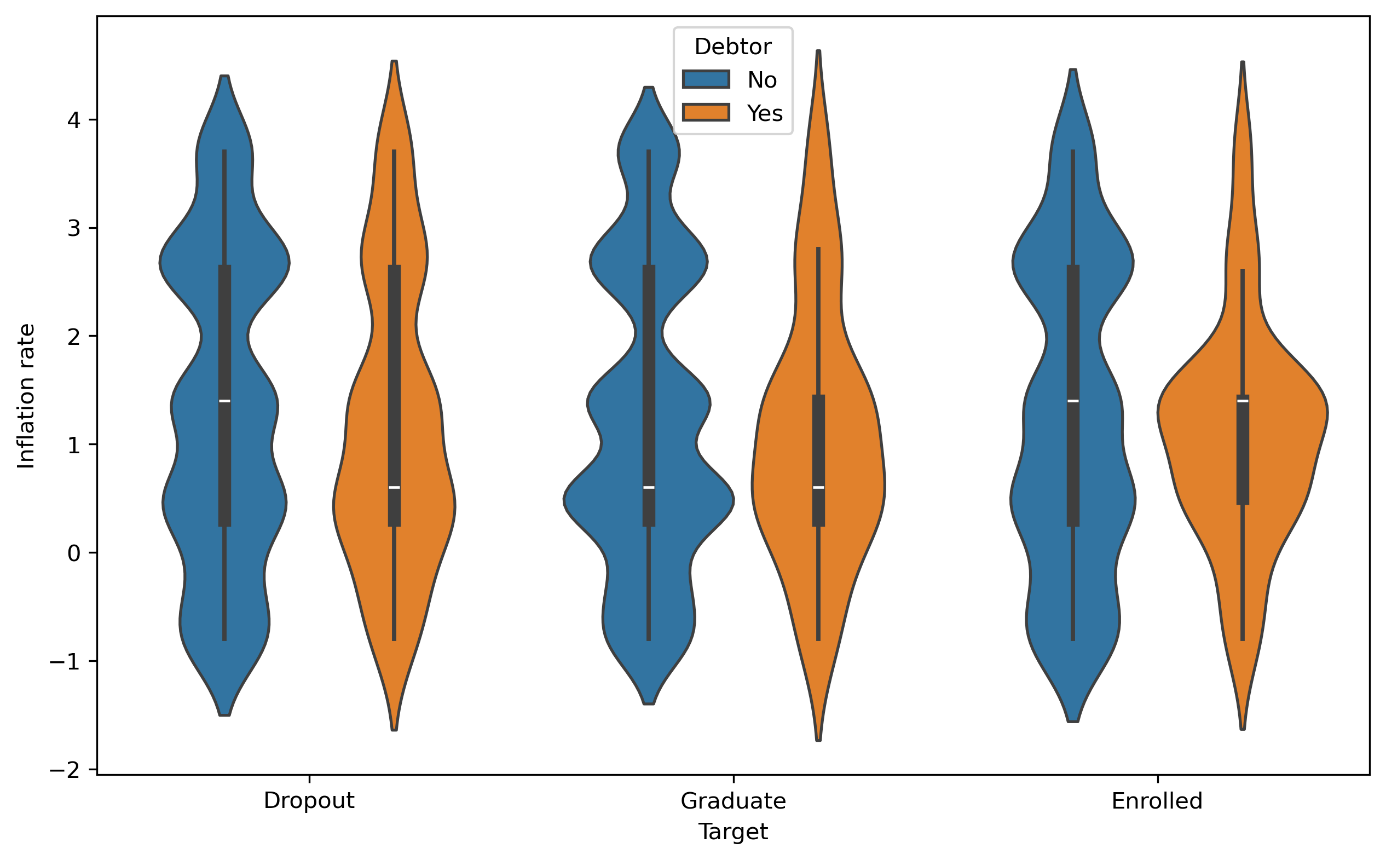
* The older the student, the more likely they enrolled during periods of lower GDP, suggesting that economic downturns may motivate adult learners to pursue education or reskilling.
* Similarly, there is a positive correlation between age at enrollment and unemployment rate, indicating that middle-aged and elderly students are more likely to enroll during times of higher unemployment.
* These patterns imply that non-traditional students might seek education as a response to unfavorable job market conditions, using academic programs as a way to reorient their careers or wait out economic hardship.

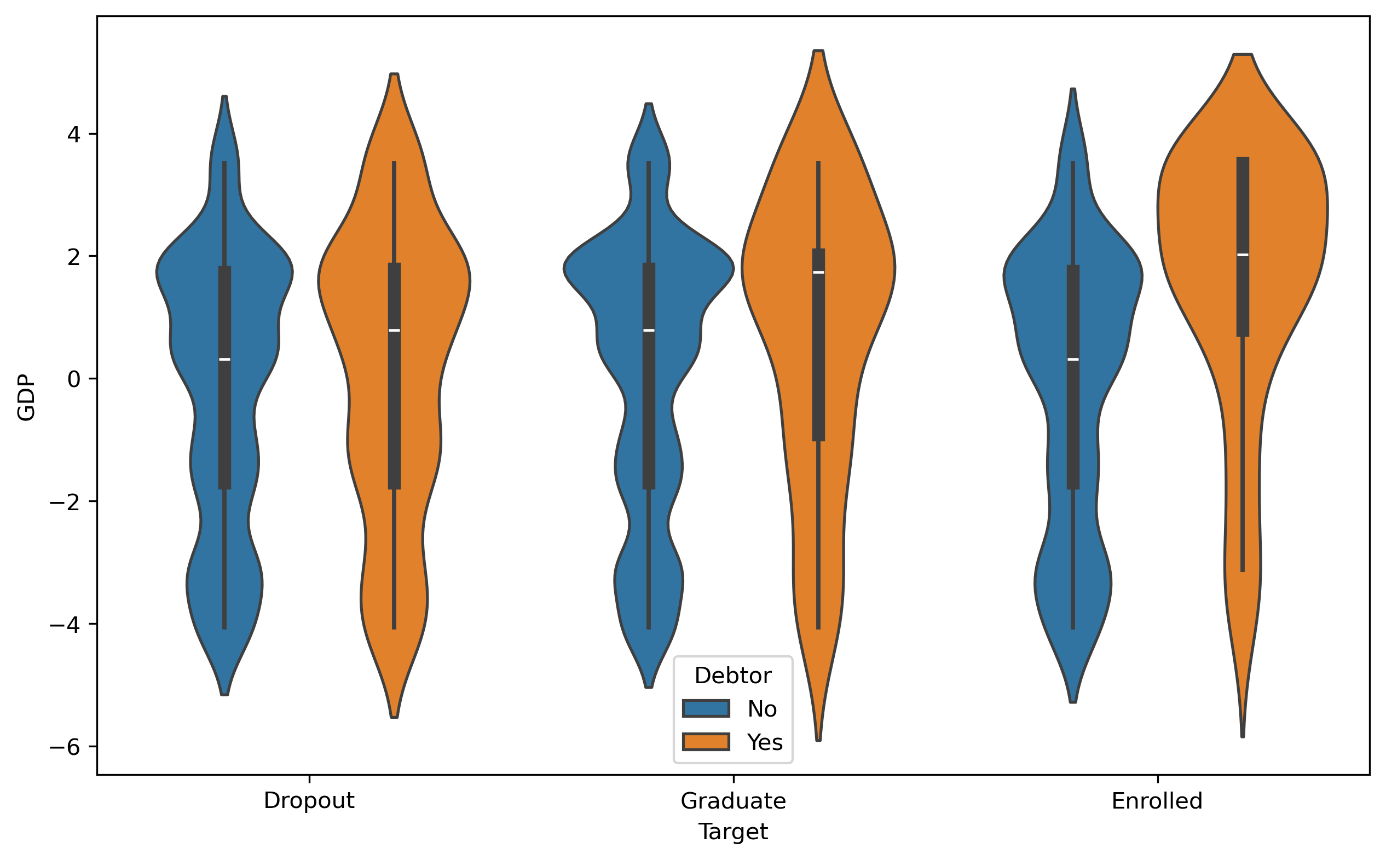


Observations

* Students tend to achieve better grades during periods of economic growth, as indicated by the positive correlation between GDP and 1st semester grades.
* Non-debtor students consistently perform better than debtor students across all economic conditions, though both groups benefit from a rising GDP.
* The performance gap between debtor and non-debtor students narrows as GDP increases, suggesting that economic stability may help disadvantaged students catch up.
* In times of economic decline (negative GDP), debtor students are especially vulnerable, with their grades dropping significantly more than those of their non-debtor peers.





****

Observations

* Students who are debtors are more likely to have enrolled during times of economic growth — indicated by the higher GDP values among debtors across all groups (dropout, graduate, enrolled).
* The unemployment rate is lower for debtor students, especially in the enrolled group, suggesting that students may accumulate debt during stable job market periods, potentially feeling safer to take financial risks.
* A similar trend is seen with inflation: debtor students typically enrolled during low-inflation periods, which could signal greater financial confidence or loan availability.
* These patterns suggest that favorable macroeconomic conditions (higher GDP, lower unemployment and inflation) are associated with increased student indebtedness, likely due to greater accessibility to credit and perceived economic safety.

**Part 2**

**Data portioning**

In each notebook the dataset is divided into the following sets

* training set – 64%
* validation set - 16%
* test set - 20%

**Model training and prediction using a pipeline**

Following column transformer was used:

* for numerical features:
  + replacement of missing values with the median
  + standardization
* for categorical features:
  + replacement of missing values with the most frequent category
  + one-hot encoding

The following models from the scikit-learn libarary were used:

1. Logistic Regression (LogisticRegression)
2. Decision Tree Classifier (DecisionTreeClassifier)
3. Support Vector Machine (SVC)

Each model was trained and evaluated five times for accuracy on the training, validation, and test datasets.

1. Logistic Regression (LogisticRegression)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Dataset | 1 | 2 | 3 | 4 | 5 |
| Train | 0.8100 | 0.8093 | 0.8100 | 0.8191 | 0.8093 |
| Validation | 0.7740 | 0.7797 | 0.7938 | 0.7429 | 0.7768 |
| Test | 0.7797 | 0.7864 | 0.7763 | 0.7853 | 0.7729 |

1. Decision Tree Classifier (DecisionTreeClassifier)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Dataset | 1 | 2 | 3 | 4 | 5 |
| Train | 0.8001 | 0.8117 | 0.8025 | 0.8011 | 0.8061 |
| Validation | 0.7373 | 0.7218 | 0.7641 | 0.7302 | 0.7246 |
| Test | 0.7458 | 0.7435 | 0.7627 | 0.7277 | 0.7401 |

1. Support Vector Machine (SVC)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Dataset | 1 | 2 | 3 | 4 | 5 |
| Train | 0.8365 | 0.8400 | 0.8410 | 0.8365 | 0.8301 |
| Validation | 0.7825 | 0.7458 | 0.7458 | 0.7571 | 0.7782 |
| Test | 0.7808 | 0.7842 | 0.7819 | 0.7785 | 0.7842 |

The Logistic Regression and SVC models achieve similar accuracies across training, validation, and test sets. On the other hand, the Decision Tree Classifier achieves worse results, which might be the matter of parametrization.

**Closed-form Linear Regression**

Linear regression has the following equation:

where

y – vector of observed values

X – matrix of observations (rows) and features (columns)

w – vector of weights

Then the normal equation to compute the optimal weights *w* directly:

**Linear Regression with Gradient Descent**

This implementation iteratively minimizes the MSE over the training data to optimize the weights.

The linear regression model assumes

The weights are iteratively updated using

Where the gradient for the const function is given by

**Linear Regression implementations**

Each linear regression implementation was trained and evaluated five times for mean squared error on the training, validation, and test datasets

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Dataset | 1 | 2 | 3 | 4 | 5 |
| Train | 136.4140 | 135.5521 | 136.2144 | 137.0914 | 134.5934 |
| Validation | 128.7778 | 131.9047 | 130.4448 | 126.2236 | 135.9268 |
| Test | 131.2311 | 130.6790 | 130.6887 | 130.8080 | 130.3862 |

1. Closed-form Linear Regression

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Dataset | 1 | 2 | 3 | 4 | 5 |
| Train | 136.5241 | 135.7230 | 136.3940 | 137.2926 | 134.7437 |
| Validation | 128.9442 | 131.9764 | 130.1461 | 126.5491 | 136.1407 |
| Test | 131.4243 | 130.6763 | 130.1332 | 130.7191 | 130.5336 |

1. Linear Regression with Gradient Descent

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Dataset | 1 | 2 | 3 | 4 | 5 |
| Train | 136.4140 | 135.5521 | 136.2144 | 137.0914 | 134.5934 |
| Validation | 128.7778 | 131.9047 | 130.4448 | 126.2236 | 135.9268 |
| Test | 131.2311 | 130.6190 | 130.6887 | 130.8080 | 130.3862 |

1. Sklearn Linear Regression

All three implementations produce comparable results. However, the gradient descent approach turns out to be slightly less effective, which is expected due to its iterative nature and reliance on stochasticity. On the other hand, both the closed-form solution and scikit-learn's implementation yield identical results.

**Logistic Regression with Gradient Descent**

This implementation iteratively minimizes the binary cross-entropy loss over the training data to optimize the weights.

The logistic regression model assumes

where is the sigmoid function, X is the input matrix, and w is the weight vector.

The weights are iteratively updated using

Where the gradient for the const function is given by

Each logistic regression implementation was trained and evaluated five times for accuracy on the training, validation, and test datasets

1. Custom Logistic Regression with gradient descent

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Dataset | 1 | 2 | 3 | 4 | 5 |
| Train | 0.8199 | 0.8124 | 0.8209 | 0.8244 | 0.8223 |
| Validation | 0.7910 | 0.8023 | 0.7881 | 0.7528 | 0.7726 |
| Test | 0.7571 | 0.7684 | 0.7661 | 0.7650 | 0.7684 |

1. Sklearn Logistic Regression

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Dataset | 1 | 2 | 3 | 4 | 5 |
| Train | 0.7838 | 0.7838 | 0.7831 | 0.7817 | 0.7821 |
| Validation | 0.7655 | 0.7868 | 0.7655 | 0.7797 | 0.7952 |
| Test | 0.7537 | 0.7559 | 0.7514 | 0.7548 | 0.7571 |

The custom logic regression implementation with gradient descent provides slightly worse evaluations compared to the scikit-learn’s implementation, which might be the matter of parametrization.

**Linear Regression with PyTorch**

The model is built as a simple neural network with:

* a single linear layer (only one layer of neurons that perform linear transformation   
  y = Wx+ b)
* Mean Squared Error (MSE) as the loss function
* an optimizer such as Stochastic Gradient Descent (SGD)

The training loop consists of:

* forward pass through the model (returns output / prediction with the current weights)
* computation of the loss using a cost function (MSE)
* backward pass (backpropagation, gradient computation to determine how each parameter / weight contributed to the output / prediction)
* weight update via the optimizer (adjusts weights to reduce the loss)

Linear regression was trained and evaluated five times for mean squared error on the training, validation, and test datasets for GPU and CPU

1. CPU

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 |
| Training time | 3.98 | 3.21 | 3.90 | 3.89 | 3.88 |
| Train | 133.4656 | 136.5765 | 137.6624 | 138.1828 | 138.0577 |
| Validation | 151.2813 | 138.6728 | 135.2426 | 132.1579 | 132.6512 |
| Test | 121.7871 | 123.2026 | 124.0619 | 122.4109 | 122.2585 |

1. GPU

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 |
| Training time | 6.13 | 6.03 | 5.93 | 5.94 | 5.97 |
| Train | 133.5033 | 136.5796 | 137.6080 | 138.1756 | 138.0409 |
| Validation | 151.162277 | 138.6446 | 134.9460 | 132.2841 | 132.5421 |
| Test | 121.6101 | 123.3583 | 123.9044 | 122.5772 | 122.2799 |

The comparison between CPU and GPU execution reveals that both configurations achieve similar results in terms of MSE. The difference is negligible, confirming that the model behaves consistently regardless of the computational backend.

Interestingly, the GPU runtime is higher than the CPU runtime, which may seem counterintuitive at first. However, this can be explained by the overhead associated with data transfer between the CPU and GPU, which includes CUDA library setup or VRAM initialization. For small datasets and not complicated models like in this case, the GPU advantage is not fully utilized. Nevertheless, more complex computing problems are where GPU truly outperforms CPU.

**Part 3**

**Introduction**

The third and final part of the project focuses on optimizing previously developed machine learning models to enhance their generalization and performance. For instance, this includes fine-tuning hyperparameters, addressing class imbalance, and testing various regularization strategies.

Each method was explored independently to assess its individual impact on model accuracy and generalization. Based on these findings, the most effective approaches were then combined to achieve best estimators for unseen data.

1. **Cross validation**

In this notebook, K-Fold cross validation was applied with k = 3, which means the dataset was split into three equally sized folds. The model was trained on two folds and validated on the remaining one, repeating this process three times so that each fold served as the validation set once. This approach helps to reduce the bias associated with random train/test split and provides a more robust estimate of model performance.

**Sklearn Logistic Regression**

|  |  |  |  |
| --- | --- | --- | --- |
| Metric | 1 | 2 | 3 |
| Accuracy | 0.7654 | 0.7749 | 0.7782 |
| Precision | 0.7539 | 0.7617 | 0.7609 |
| Recall | 0.7654 | 0.7749 | 0.7782 |

**Custom Implementation of Logistic Regression**

|  |  |  |  |
| --- | --- | --- | --- |
| Metric | 1 | 2 | 3 |
| Accuracy | 0.7661 | 0.7539 | 0.7741 |
| Precision | 0.7461 | 0.7317 | 0.7552 |
| Recall | 0.7661 | 0.7539 | 0.7741 |

**Interpretation**

The results from the 3-fold cross-validation indicate consistent model performance across all folds for both implementations.

* For Sklearn Logistic Regression the accuracy varies slightly, which suggests stable behavior and good generalization of the model. Fold 3 yields the best performance across all metrics (accuracy, precision, and recall), suggesting that the data split in this fold aligns best with the model’s learning dynamics.
* The custom implementation of logistic regression also shows comparable stability.

The small variance in metrics across folds in both cases indicates that neither implementation suffers from high variance or overfitting. The custom model performs closely to the sklearn baseline, confirming its correctness and reliability.

1. **Convergence diagrams**

Convergence diagrams provide a clear visualization of the mode’s learning process by tracking the value of the loss function across training iterations. They offer valuable insight into how effectively the model is optimizing and help identify key moments such as convergence or the onset of overfitting.

**The convergence plots for various sets of features**

1. Obraz zawierający tekst, Wykres, linia, diagram

   Zawartość wygenerowana przez AI może być niepoprawna.All features
2. Reduced features

* Age at enrollment
* GDP
* Unemployment rate
* Inflation rate
* Curricular units 1st sem (approved)
* Curricular units 1st sem (grade)
* Admission grade
* Previous qualification (grade)
* Marital status
* Application mode
* Application order
* Daytime/evening attendance
* Displaced
* Debtor
* Scholarship holder
* Gender
* Mother's occupation
* Father's occupation
* Mother's qualification
* Father's qualification

Obraz zawierający tekst, diagram, Wykres, linia

Zawartość wygenerowana przez AI może być niepoprawna.

1. Economic features

* Age at enrollment
* GDP
* Unemployment rate
* Inflation rate
* Mother’s qualification
* Father’s qualification
* Mother’s occupation
* Father’s occupation
* Debtor
* Scholarship holder

Obraz zawierający tekst, Wykres, diagram, linia

Zawartość wygenerowana przez AI może być niepoprawna.

1. Single feature – Age at enrollment

Obraz zawierający tekst, linia, numer, Wykres

Zawartość wygenerowana przez AI może być niepoprawna.

**Interpretation**

A noticeable decline in model accuracy can be observed for both the training and test datasets as the number of features decreases. This suggests that retaining the full set of features is beneficial for achieving optimal model performance.

**The convergence plots for dimensionality (polynomial features)**

1. degree = 1

Obraz zawierający tekst, Wykres, linia, diagram

Zawartość wygenerowana przez AI może być niepoprawna.

1. degree = 2

Obraz zawierający tekst, Wykres, linia, diagram

Zawartość wygenerowana przez AI może być niepoprawna.

1. degree = 3

Obraz zawierający tekst, Wykres, diagram, linia

Zawartość wygenerowana przez AI może być niepoprawna.

1. degree = 4

Obraz zawierający tekst, Wykres, linia, diagram

Zawartość wygenerowana przez AI może być niepoprawna.

**Interpretation**

We can notice that for our dataset, there is no significant overfitting for degrees 1 2 3. However the training time increases exponentially, therefore I would consider mostly

**The built-in convergence plots for my implementation of logistic regression**

1. degree = 1

Obraz zawierający tekst, Wykres, diagram, linia

Zawartość wygenerowana przez AI może być niepoprawna.

1. degree = 2 without bias

Obraz zawierający tekst, Wykres, diagram, linia

Zawartość wygenerowana przez AI może być niepoprawna.

1. degree = 2 with bias

Obraz zawierający tekst, Wykres, diagram, linia

Zawartość wygenerowana przez AI może być niepoprawna.

1. degree = 3 without bias

Obraz zawierający tekst, Wykres, diagram, linia

Zawartość wygenerowana przez AI może być niepoprawna.

1. Reduced features

* Age at enrollment
* GDP
* Unemployment rate
* Inflation rate
* Curricular units 1st sem (approved)
* Curricular units 1st sem (grade)
* Admission grade
* Previous qualification (grade)
* Marital status
* Application mode
* Application order
* Daytime/evening attendance
* Displaced
* Debtor
* Scholarship holder
* Gender
* Mother’s occupation
* Father’s occupation
* Mother’s qualification
* Father’s qualification

Obraz zawierający tekst, diagram, Wykres, linia

Zawartość wygenerowana przez AI może być niepoprawna.

1. Reduced features with degree = 3

* Economic features
* Age at enrollment
* GDP
* Unemployment rate
* Inflation rate
* Mother’s qualification
* Father’s qualification
* Mother’s occupation
* Father’s occupation
* Debtor
* Scholarship holder

Obraz zawierający tekst, Wykres, diagram, linia

Zawartość wygenerowana przez AI może być niepoprawna.

1. Single feature – Age at enrollment

**Obraz zawierający tekst, diagram, Wykres, linia

Zawartość wygenerowana przez AI może być niepoprawna.**

**Interpretation**

We observe that for various polynomial degrees, the model’s learning rate slows down and the error function fluctuates around similar values, with the exception of degree =3, where slight overfitting becomes noticeable after 500 epochs. Additionally, a signification decline in training performance is observed for datasets with reduced feature set.

1. **Regularization**

Regularization is a technique used to prevent overfitting in machine learning models by adding a penalty term to the loss function. This penalty discourages overly complex models by constraining the magnitude of the model’s coefficients, leading to better generalization on unseen data. The most common regularization methods include L1 (Lasso) and L2 (Ridge) regularization, each influencing the model behavior in different ways.

1. Without regularization

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Dataset | 1 | 2 | 3 | 4 | 5 |
| Train | 0.8092 | 0.8127 | 0.8098 | 0.8115 | 0.8040 |
| test | 0.7774 | 0.7684 | 0.7774 | 0.7684 | 0.7952 |

1. Lasso regularization (L1)

Training datasets

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 |
| 0.01 | 0.7302 | 0.7285 | 0.7248 | 0.7324 | 0.7240 |
| 0.1 | 0.7717 | 0.7802 | 0.7739 | 0.7782 | 0.7712 |
| 1 | 0.7983 | 0.7999 | 0.7974 | 0.7988 | 0.7907 |
| 10 | 0.8079 | 0.8096 | 0.8079 | 0.8090 | 0.8040 |
| 100 | 0.8084 | 0.8093 | 0.8093 | 0.8104 | 0.8045 |
| 1000 | 0.8087 | 0.8098 | 0.8090 | 0.8104 | 0.8048 |

Test dataset

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 |
| 0.01 | 0.7232 | 0.7243 | 0.7401 | 0.7096 | 0.7387 |
| 0.1 | 0.7706 | 0.7492 | 0.7638 | 0.7559 | 0.7851 |
| 1 | 0.7831 | 0.7605 | 0.7706 | 0.7638 | 0.7998 |
| 10 | 0.7729 | 0.7582 | 0.7661 | 0.7638 | 0.7941 |
| 100 | 0.7706 | 0.7593 | 0.7616 | 0.7582 | 0.7851 |
| 1000 | 0.7638 | 0.7559 | 0.7616 | 0.7559 | 0.7783 |

1. Ridge regularization (L2)

Training datasets

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 |
| 0.01 | 0.7759 | 0.7790 | 0.7762 | 0.7796 | 0.7718 |
| 0.1 | 0.7982 | 0.7994 | 0.7982 | 0.8028 | 0.7904 |
| 1 | 0.8093 | 0.8127 | 0.8098 | 0.8115 | 0.8040 |
| 10 | 0.8152 | 0.8175 | 0.8146 | 0.8166 | 0.8076 |
| 100 | 0.8152 | 0.8172 | 0.8149 | 0.8166 | 0.8082 |
| 1000 | 0.8158 | 0.8172 | 0.8155 | 0.8161 | 0.8076 |

Test dataset

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 |
| 0.01 | 0.7706 | 0.7514 | 0.7581 | 0.7503 | 0.7941 |
| 0.1 | 0.7842 | 0.7718 | 0.7718 | 0.7650 | 0.7998 |
| 1 | 0.7774 | 0.7684 | 0.7774 | 0.7684 | 0.7952 |
| 10 | 0.7695 | 0.7661 | 0.7729 | 0.7673 | 0.7919 |
| 100 | 0.7661 | 0.7650 | 0.7672 | 0.7627 | 0.7896 |
| 1000 | 0.7672 | 0.7627 | 0.7661 | 0.7605 | 0.7896 |

**Interpretation**

The accuracy of the logistic regression model with regularization varies across test sets depending on the value of the regularization parameter C, which controls the inverse strength of regularization (lower values imply stronger regularization). For Lasso (L1) regularization, the highest test accuracy is observed at C = 1, while Ridge (L2) performs best at C = 0.1 and C = 1. As shown in the data, lower C values (e.g., 0.01) result in underfitting, whereas higher values (e.g., 1000) lead to overfitting. Overall, models with regularization demonstrate slightly improved generalization compared to the unregularized baseline. Among the two, Ridge (L2) marginally outperforms Lasso (L1), though the difference is minimal and may not be significant in practice.

1. **Class balancing techniques**

Class imbalance can negatively impact model performance by biasing predictions toward the majority class. To address this, resampling techniques can be used to balance the dataset. Two common approaches utilized in this project include SMOTE (Synthetic Minority Oversampling Technique), which generates synthetic examples of the minority class, and RandomUnderSampling, which reduces the size of the majority class by randomly removing instances. These methods aim to improve the model’s ability to improve the model’s ability to learn from all classes equally, enhancing classification performance on imbalanced datasets.

Various logistic regression models were trained on imbalanced, oversampled and undersampled datasets.

1. Sklearn implementation of logistic regression – imbalanced dataset

Train

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Metric | 1 | 2 | 3 | 4 | 5 |
| Accuracy | 0.7785 | 0.8132 | 0.8101 | 0.8110 | 0.8040 |
| Precision | 0.7663 | 0.8046 | 0.8020 | 0.8021 | 0.7949 |
| Recall | 0.7785 | 0.8132 | 0.8101 | 0.8110 | 0.8040 |
| F1 | 0.7691 | 0.8029 | 0.8005 | 0.8009 | 0.7934 |

Test

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Metric | 1 | 2 | 3 | 4 | 5 |
| Accuracy | 0.7785 | 0.7684 | 0.7763 | 0.7684 | 0.7952 |
| Precision | 0.7663 | 0.7573 | 0.7620 | 0.7548 | 0.7829 |
| Recall | 0.7785 | 0.7684 | 0.7763 | 0.7684 | 0.7952 |
| F1 | 0.7691 | 0.7513 | 0.7647 | 0.7539 | 0.7835 |

1. Sklearn implementation of logistic regression – oversampled dataset

Train

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Metric | 1 | 2 | 3 | 4 | 5 |
| Accuracy | 0.7924 | 0.8049 | 0.7867 | 0.7955 | 0.7873 |
| Precision | 0.7954 | 0.8077 | 0.7890 | 0.7980 | 0.7896 |
| Recall | 0.7924 | 0.8049 | 0.7867 | 0.7955 | 0.7873 |
| F1 | 0.7927 | 0.8053 | 0.7869 | 0.7958 | 0.7876 |

Test

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Metric | 1 | 2 | 3 | 4 | 5 |
| Accuracy | 0.7446 | 0.7537 | 0.7480 | 0.7503 | 0.7839 |
| Precision | 0.7758 | 0.7604 | 0.7731 | 0.7699 | 0.8018 |
| Recall | 0.7446 | 0.7537 | 0.7480 | 0.7503 | 0.7839 |
| F1 | 0.7562 | 0.7566 | 0.7571 | 0.7368 | 0.7902 |

1. Sklearn implementation of logistic regression – undersampled dataset

Train

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Metric | 1 | 2 | 3 | 4 | 5 |
| Accuracy | 0.7786 | 0.7644 | 0.7762 | 0.7778 | 0.7708 |
| Precision | 0.7806 | 0.7673 | 0.7782 | 0.7780 | 0.7718 |
| Recall | 0.7786 | 0.7644 | 0.7762 | 0.7778 | 0.7708 |
| F1 | 0.7785 | 0.7645 | 0.7759 | 0.7773 | 0.7701 |

Test

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Metric | 1 | 2 | 3 | 4 | 5 |
| Accuracy | 0.7186 | 0.7356 | 0.7333 | 0.7288 | 0.7568 |
| Precision | 0.7653 | 0.7505 | 0.7660 | 0.7599 | 0.7923 |
| Recall | 0.7186 | 0.7356 | 0.7333 | 0.7288 | 0.7568 |
| F1 | 0.7349 | 0.7416 | 0.7453 | 0.7389 | 0.7683 |

1. Custom implementation of logistic regression – imbalanced set

Train

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Metric | 1 | 2 | 3 | 4 | 5 |
| Accuracy | 0.7784 | 0.7855 | 0.7824 | 0.7858 | 0.7774 |
| Precision | 0.7617 | 0.7723 | 0.7696 | 0.7703 | 0.7621 |
| Recall | 0.7785 | 0.7855 | 0.7824 | 0.7858 | 0.7774 |
| F1 | 0.7535 | 0.7636 | 0.7623 | 0.7637 | 0.7537 |

Test

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Metric | 1 | 2 | 3 | 4 | 5 |
| Accuracy | 0.7729 | 0.7627 | 0.7650 | 0.7638 | 0.7975 |
| Precision | 0.7583 | 0.7434 | 0.7389 | 0.7491 | 0.7872 |
| Recall | 0.7729 | 0.7627 | 0.7650 | 0.7638 | 0.7975 |
| F1 | 0.7508 | 0.7359 | 0.7428 | 0.7376 | 0.7773 |

1. Custom implementation of logistic regression – oversampled set

Train

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Metric | 1 | 2 | 3 | 4 | 5 |
| Accuracy | 0.7510 | 0.7560 | 0.7510 | 0.7555 | 0.7468 |
| Precision | 0.7543 | 0.7594 | 0.7543 | 0.7590 | 0.7494 |
| Recall | 0.7510 | 0.7560 | 0.7510 | 0.7555 | 0.7468 |
| F1 | 0.7504 | 0.7553 | 0.7507 | 0.7552 | 0.7460 |

Test

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Metric | 1 | 2 | 3 | 4 | 5 |
| Accuracy | 0.7401 | 0.7605 | 0.7390 | 0.7424 | 0.7828 |
| Precision | 0.7667 | 0.7633 | 0.7654 | 0.7614 | 0.8016 |
| Recall | 0.7401 | 0.7605 | 0.7390 | 0.7424 | 0.7828 |
| F1 | 0.7501 | 0.7611 | 0.7486 | 0.7485 | 0.7891 |

1. Custom implementation of logistic regression – undersampled set

Train

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Metric | 1 | 2 | 3 | 4 | 5 |
| Accuracy | 0.7198 | 0.7158 | 0.7230 | 0.7273 | 0.7286 |
| Precision | 0.7224 | 0.7183 | 0.7259 | 0.7284 | 0.7304 |
| Recall | 0.7198 | 0.7158 | 0.7230 | 0.7273 | 0.7286 |
| F1 | 0.7181 | 0.7145 | 0.7218 | 0.7250 | 0.7260 |

Test

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Metric | 1 | 2 | 3 | 4 | 5 |
| Accuracy | 0.7390 | 0.7513 | 0.7379 | 0.7243 | 0.7647 |
| Precision | 0.7731 | 0.7560 | 0.7615 | 0.7462 | 0.7897 |
| Recall | 0.7390 | 0.7514 | 0.7379 | 0.7243 | 0.7647 |
| F1 | 0.7511 | 0.7528 | 0.7466 | 0.7320 | 0.7731 |

**Interpretation**

In this particular case, both oversampling and undersampling negatively impact the model’s ability to generalize to unseen data. Notably, undersampling yields the weakest performance, likely due to the loss of valuable information from the majority class.

1. **Parameters optimization**

To enhance model performance, hyperparameter tuning was conducted using GridSearchCV, a tool from scikit-learn that performs an exhaustive search over specified parameter values for a given model. It evaluates all possible combinations using cross-validation, helping to identify the set of hyperparameters that yields the best performance.

1. Random Forest Classifier – best hyperparameters

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 |
| max\_depth | 30 | 30 | None | 30 | 30 |
| min\_samples\_split | 2 | 3 | 3 | 2 | 2 |
| n\_estimators | 100 | 500 | 200 | 500 | 500 |
| Accuracy | 0.7638 | 0.7706 | 0.7763 | 0.7763 | 0.7964 |
| Precision | 0.8103 | 0.8430 | 0.8283 | 0.8518 | 0.8514 |
| Recall | 0.7638 | 0.7706 | 0.7763 | 0.7763 | 0.7964 |
| F1 | 0.7801 | 0.7947 | 0.7952 | 0.8003 | 0.8147 |

1. K Neighbors Classifier – best hyperparameters

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 |
| n\_neighbors | 7 | 7 | 9 | 9 | 15 |
| p | 2 | 2 | 2 | 2 | 2 |
| Accuracy | 0.6949 | 0.7164 | 0.7141 | 0.7085 | 0.7319 |
| Precision | 0.7369 | 0.7843 | 0.7843 | 0.7812 | 0.8164 |
| Recall | 0.6949 | 0.7164 | 0.7141 | 0.7085 | 0.7319 |
| F1 | 0.7097 | 0.7393 | 0.7369 | 0.7302 | 0.7564 |

1. Logistic Regression – best hyperparameters

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 |
| C | 1 | 1 | 1 | 0.1 | 0.1 |
| max\_iter | 200 | 200 | 200 | 200 | 200 |
| penalty | l2 | l1 | l1 | l2 | l2 |
| solver | liblinear | liblinear | liblinear | liblinear | liblinear |
| Accuracy | 0.7740 | 0.7605 | 0.7706 | 0.7627 | 0.7907 |
| Precision | 0.8174 | 0.8222 | 0.8198 | 0.8361 | 0.8471 |
| Recall | 0.7740 | 0.7605 | 0.7706 | 0.7627 | 0.7907 |
| F1 | 0.7893 | 0.7826 | 0.7890 | 0.7867 | 0.8102 |

1. Custom implementation of Logistic Regression – best hyperparameters

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 |
| batch\_size | 128 | 128 | 128 | 128 | 128 |
| learning\_rate | 0.01 | 0.01 | 0.01 | 0.01 | 0.01 |
| Accuracy | 0.7808 | 0.7650 | 0.7695 | 0.7593 | 0.7919 |
| Precision | 0.8294 | 0.8377 | 0.8251 | 0.8303 | 0.8434 |
| Recall | 0.7808 | 0.7650 | 0.7695 | 0.7593 | 0.7919 |
| F1 | 0.7976 | 0.7908 | 0.7901 | 0.7829 | 0.8095 |

**Interpretation**

The Random Forest Classifier achieved the highest overall performance, particularly in F1-score and precision. Logistic Regression showed stable results, with L2 regularization performing slightly better. The K-Nearest Neighbors model had lower accuracy overall, though tuning the number of neighbors significantly influenced performance. Custom logistic regression closely matched the sklearn implementation, confirming its reliability.

1. **Ensemble methods**

Ensemble methods are machine learning techniques that combine predictions from multiple models to improve overall performance and robustness. By aggregating the strengths of diverse learners, ensembles often outperform individual models. Two commonly used ensemble approaches are the VotingClassifier, which makes predictions based on majority voting (hard) or averaged probabilities (soft), and the StackingClassifier, which combines base model outputs using a meta-model trained to optimize final predictions.

The used combination of estimators include 3 models:

* Random Forest Classifier(300 estimators)
* Logistic Regression (1000 iterations)
* SVC

1. Voting Classifier

* Accuracy: 0.7710 (+/- 0.0022) [Random Forest Classifier]
* Accuracy: 0.7570 (+/- 0.0045) [Logisitic Regression]
* Accuracy: 0.7615 (+/- 0.0032) [SVC]
* Accuracy: 0.7715 (+/- 0.0069) [Voting Classifier]

1. Stacking Classifier

* Accuracy: 0.7684

**Interpretation**

The VotingClassifier achieved a slightly better accuracy than the individual models, though the improvement over the strongest base learner was minimal. The StackingClassifier performed comparably but did not surpass the voting ensemble. While ensemble methods helped stabilize performance, the overall gain in this setup was modest.

**Implementation of Mixture of Experts**

The Mixture of Experts (MoE) is an ensemble technique that divides the input space into regions and assigns different models—called *experts*—to specialize in each region. A gating mechanism determines which expert is responsible for making a prediction on a given input. In this implementation, K-Means clustering is used as the gating model to partition the data, and each expert is trained only on the subset of data assigned to its cluster. During prediction, new samples are routed to the corresponding expert based on the cluster assignment.

1. Mixture of experts

* Logistic Regression (max iter = 300)
* SVC
* RandomForestClassifier

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 |
| Accuracy | 0.7661 | 0.7797 | 0.7672 | 0.7480 | 0.7613 |

1. Mixture of experts – 5x Logistic Regression (max iter = 250)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 |
| Accuracy | 0.7661 | 0.7797 | 0.7672 | 0.7480 | 0.7613 |

1. Mixture of experts – 5x Random Forest Classifier

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 |
| Accuracy | 0.7638 | 0.7774 | 0.7774 | 0.7333 | 0.7670 |

**Interpretation**

The Mixture of Experts models achieved stable but not noticeably better results compared to earlier techniques. The slight gains from using diverse experts suggest potential, though the lack of hyperparameter tuning likely limited performance.

**Final model**

For the final model, I employed an ensemble approach combining Logistic Regression, Support Vector Classifier (SVC), and Random Forest Classifier, each configured with optimized hyperparameters. Hyperparameter tuning was performed using GridSearchCV in conjunction with K-Fold cross-validation to ensure robust model selection and generalization.

1. SVC
2. Parameters grid 1

* C [0.1 / 1 / 10]
* Kernel [linear / rbf]
* Gamma [scale / auto]

1. Parameters grid 2

* C [0.1 / 1 / 1]
* Kernel [poly]
* Gamma [scale / auto]
* Degree [2 /3 /4]

Best models

1. Model 1

* C [1]
* Gamma [scale]
* Kernel [rbf]

1. Model 2

* C [10]
* Gamma [auto]
* Kernel [rbf]

1. Model 3

* C [10]
* Gamma [auto]
* Kernel [rbf]

1. Random Forest Classifier

Parameters grid

* Number of estimators [100 / 200 / 300 / 500]
* Max depth [30 / 50 / None]
* Min samples split [2 / 3]
* Min samples leaf [1 / 2]
* Bootstrap [True / False]

Best models

1. Model 1

* Bootstrap [False]
* Max depth [None]
* Min samples leaf [2]
* Min samples split [3]
* Number of estimators [200]

1. Model 2

* Bootstrap [False]
* Max depth [50]
* Min samples leaf [1]
* Min samples split [2]
* Number of estimators [500]

1. Model 3

* Bootstrap [False]
* Max depth [30]
* Min samples leaf [1]
* Min samples split [2]
* Number of estimators [500]

1. Logistic Regression
2. Parameters grid 1

* Degree [1 / 2]
* Penalty [l1]
* C [0.1 / 1]
* Solver [liblinear]
* Max iterations [40 / 100 / 200]

1. Parameters grid 2

* Degree [1 / 2]
* Penalty [None, l2]
* C [0.1 / 1]
* Solver [lbfgs]
* Max iterations [40 / 100 / 200]

Best models

1. Model 1

* Degree [1]
* Penalty [2]
* C [0.1]
* Solver [lbfgs]
* Max iterations [100]

1. Model 2

* Degree [1]
* Penalty [l2]
* C [1]
* Solver [lbfgs]
* Max iterations [100]

1. Model 3

* Degree [1]
* Penalty [l2]
* C [0.1]
* Solver [lbfgs]
* Max iterations [100]

Moreover, a convergence plot with the loss function across iterations was generated for a few configuration with different hyperparameters

1. Configuration 1

* Degree [1]
* Penalty [l1]
* C [1]
* Solver [liblinear]

Obraz zawierający tekst, linia, Wykres, diagram

Zawartość wygenerowana przez AI może być niepoprawna.

1. Configuration 2

* Degree [2]
* Penalty [l1]
* C [1]
* Solver [liblinear]

Obraz zawierający tekst, Wykres, diagram, linia

Zawartość wygenerowana przez AI może być niepoprawna.

1. Configuration 3

* Degree [1]
* Penalty [l2]
* C [1]
* Solver [lbfgs]

Obraz zawierający tekst, linia, Wykres, diagram

Zawartość wygenerowana przez AI może być niepoprawna.

1. Configuration 4

* Degree [2]
* Penalty [l2]
* C [0.1]
* Solver [lbfgs]

Obraz zawierający tekst, linia, Wykres, diagram

Zawartość wygenerowana przez AI może być niepoprawna.

Therefore, the final estimators remain following

1. Linear Regression

* Degree [1]
* Max iterations [200]
* C [1]
* Penalty [l1]
* Solver [Liblinear]

1. Linear Regression

* Degree [1]
* Max iterations [200]
* C [0.1]
* Penalty [l2]
* Solver [lbgfs]

1. Linear Regression

* Degree [2]
* Max iterations [200]
* C [1]
* Penalty [l1]
* Solver [Liblinear]

1. Linear Regression

* Degree [2]
* Max iterations [200]
* C [0.1]
* Penalty [l2]
* Solver [lbgfs]

1. Random Forest Classifier

* Number of estimators [200]
* Max depth [None]
* Min samples leaf [2]
* Min samples split [3]
* Bootstrap [False]

1. Random Forest Classifier

* Number of estimators [500]
* Max depth [50]
* Min samples leaf [1]
* Min samples split [2]
* Bootstrap [False]

1. Random Forest Classifier

* Number of estimators [500]
* Max depth [30]
* Min samples leaf [1]
* Min samples split [2]
* Bootstrap [False]

1. SVC

* C [1]
* Gamma [scale]

1. SVC

* C [10]
* Gamma [scale]

1. SVC

* C [10]
* Gamma [auto]

Now this is how results for ensemble methods present

1. Voting Classifier

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 |
| Accuracy | 0.7672 | 0.7887 | 0.7889 | 0.7853 | 0.7794 |

1. Stacking Classifier

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 |
| Accuracy | 0.7616 | 0.7898 | 0.7819 | 0.7876 | 0.7794 |

**Ablation study**

To evaluate the impact of individual techniques on model performance, each method was tested independently before integrating the most effective ones into the final solution. The observed contributions are summarized below

1. **Cross validation** (3 folds) provided a robust framework for model evaluation, significantly reducing variance caused by random data splits and ensuring consistent performance assessment
2. **Convergence diagrams** allowed for detailed tracking of model training progression, exposing trends such as overfitting, the impact of feature reduction, and the effect of increasing polynomial degree
3. **Regularization (L1 / L2)** improved training stability and reduced overfitting slightly, especially at moderate values of the regularization parameter C
4. **Class balancing** using SMOTE and undersampling negatively affected performance. Oversampling led to inconsistency, while undersampling degraded accuracy due to loss of informative samples
5. **Hyperparameter optimization** (via GridSearchCV) had the greatest impact on improving model accuracy and generalization across classifiers
6. **Ensemble methods** (Voting and Stacking) let to modest accuracy improvements and enhanced robustness, outperforming most individual models
7. **Custom Mixture of Experts** demonstrated conceptual value but underperformed due to lack of parameter tuning and simplistic clustering

The most significant improvements came from using cross-validation, hyperparameter tuning, and insights from convergence diagrams, all of which helped build more stable and accurate models. Regularization and ensemble methods added small but consistent gains in generalization. In contrast, class balancing techniques had a negative effect in this case.