```python
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

# Data Preprocessing and EDA
from sklearn.preprocessing import OrdinalEncoder
from scipy.stats import chi2_contingency

# Modeling
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from xgboost import XGBClassifier
from sklearn.utils.class_weight import compute_sample_weight
from sklearn.model_selection import RandomizedSearchCV
from sklearn.ensemble import VotingClassifier
from sklearn.inspection import permutation_importance
from sklearn.metrics import balanced_accuracy_score, f1_score,
roc_auc_score, confusion_matrix, ConfusionMatrixDisplay

students = pd.read_csv('/content/dataset(1).csv')
students.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4424 entries, 0 to 4423
Data columns (total 35 columns):
 #   Column                                          Non-Null Count
Dtype
---  ------                                          --------------
-----
 0   Marital status                                  4424 non-null
int64
 1   Application mode                                4424 non-null
int64
 2   Application order                               4424 non-null
int64
 3   Course                                          4424 non-null
int64
 4   Daytime/evening attendance                      4424 non-null
int64
 5   Previous qualification                          4424 non-null
int64
 6   Nacionality                                     4424 non-null
int64
 7   Mother's qualification                          4424 non-null
int64
 8   Father's qualification                          4424 non-null
int64
```

```
 9   Mother's occupation                             4424 non-null
int64
 10  Father's occupation                             4424 non-null
int64
 11  Displaced                                       4424 non-null
int64
 12  Educational special needs                       4424 non-null
int64
 13  Debtor                                          4424 non-null
int64
 14  Tuition fees up to date                         4424 non-null
int64
 15  Gender                                          4424 non-null
int64
 16  Scholarship holder                              4424 non-null
int64
 17  Age at enrollment                               4424 non-null
int64
 18  International                                   4424 non-null
int64
 19  Curricular units 1st sem (credited)             4424 non-null
int64
 20  Curricular units 1st sem (enrolled)             4424 non-null
int64
 21  Curricular units 1st sem (evaluations)          4424 non-null
int64
 22  Curricular units 1st sem (approved)             4424 non-null
int64
 23  Curricular units 1st sem (grade)                4424 non-null
float64
 24  Curricular units 1st sem (without evaluations)  4424 non-null
int64
 25  Curricular units 2nd sem (credited)             4424 non-null
int64
 26  Curricular units 2nd sem (enrolled)             4424 non-null
int64
 27  Curricular units 2nd sem (evaluations)          4424 non-null
int64
 28  Curricular units 2nd sem (approved)             4424 non-null
int64
 29  Curricular units 2nd sem (grade)                4424 non-null
float64
 30  Curricular units 2nd sem (without evaluations)  4424 non-null
int64
 31  Unemployment rate                               4424 non-null
float64
 32  Inflation rate                                  4424 non-null
float64
 33  GDP                                             4424 non-null
```

```
float64
 34  Target                                          4424 non-null
object
dtypes: float64(5), int64(29), object(1)
memory usage: 1.2+ MB
```

```
# Examine the shape of the DataFrame.
print("Shape of the DataFrame:", df.shape)

# Get a concise summary of the DataFrame.
print("\nDataFrame Info:")
display(df.info())

# Generate descriptive statistics for numerical features.
print("\nDescriptive Statistics for Numerical Features:")
display(df.describe())

# Calculate frequency counts for categorical features.
categorical_cols = df.select_dtypes(include=['object',
'category']).columns
print("\nFrequency Counts for Categorical Features:")
for col in categorical_cols:
    print(f"\nFrequency counts for '{col}':")
    display(df[col].value_counts())

# Identify potential outliers (example: using IQR for 'Age at
enrollment')
Q1 = df['Age at enrollment'].quantile(0.25)
Q3 = df['Age at enrollment'].quantile(0.75)
IQR = Q3 - Q1
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR
potential_outliers = df[(df['Age at enrollment'] < lower_bound) |
(df['Age at enrollment'] > upper_bound)]
print("\nPotential outliers in 'Age at enrollment':")
display(potential_outliers[['Age at enrollment']])

print("\nSummary of observations:")
print("The dataset contains information about student performance and
demographics, with a target variable indicating dropout or graduation.
Some numerical features, such as age at enrollment, show potential
outliers.  Further analysis is needed.")
```

```
Shape of the DataFrame: (4424, 35)

DataFrame Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4424 entries, 0 to 4423
Data columns (total 35 columns):
```

```
 #   Column                                          Non-Null Count
Dtype
---  ------                                          --------------
-----
 0   Marital status                                  4424 non-null
int64
 1   Application mode                                4424 non-null
int64
 2   Application order                               4424 non-null
int64
 3   Course                                          4424 non-null
int64
 4   Daytime/evening attendance                      4424 non-null
int64
 5   Previous qualification                          4424 non-null
int64
 6   Nacionality                                     4424 non-null
int64
 7   Mother's qualification                          4424 non-null
int64
 8   Father's qualification                          4424 non-null
int64
 9   Mother's occupation                             4424 non-null
int64
 10  Father's occupation                             4424 non-null
int64
 11  Displaced                                       4424 non-null
int64
 12  Educational special needs                       4424 non-null
int64
 13  Debtor                                          4424 non-null
int64
 14  Tuition fees up to date                         4424 non-null
int64
 15  Gender                                          4424 non-null
int64
 16  Scholarship holder                              4424 non-null
int64
 17  Age at enrollment                               4424 non-null
int64
 18  International                                   4424 non-null
int64
 19  Curricular units 1st sem (credited)             4424 non-null
int64
 20  Curricular units 1st sem (enrolled)             4424 non-null
int64
 21  Curricular units 1st sem (evaluations)          4424 non-null
int64
 22  Curricular units 1st sem (approved)             4424 non-null
```

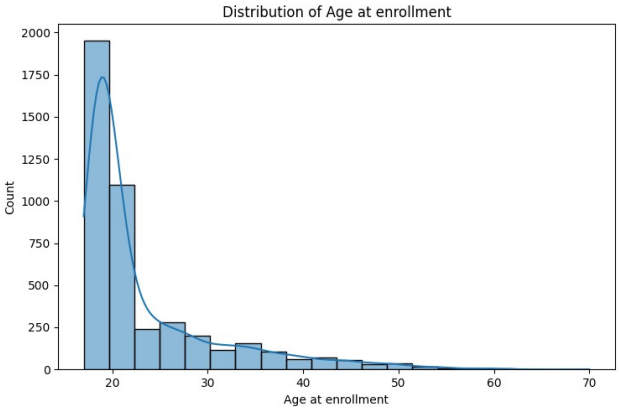
```
int64
 23  Curricular units 1st sem (grade)                4424 non-null
float64
 24  Curricular units 1st sem (without evaluations)  4424 non-null
int64
 25  Curricular units 2nd sem (credited)             4424 non-null
int64
 26  Curricular units 2nd sem (enrolled)             4424 non-null
int64
 27  Curricular units 2nd sem (evaluations)          4424 non-null
int64
 28  Curricular units 2nd sem (approved)             4424 non-null
int64
 29  Curricular units 2nd sem (grade)                4424 non-null
float64
 30  Curricular units 2nd sem (without evaluations)  4424 non-null
int64
 31  Unemployment rate                               4424 non-null
float64
 32  Inflation rate                                  4424 non-null
float64
 33  GDP                                             4424 non-null
float64
 34  Target                                          4424 non-null
object
dtypes: float64(5), int64(29), object(1)
memory usage: 1.2+ MB

None


Descriptive Statistics for Numerical Features:

{"type":"dataframe"}


Frequency Counts for Categorical Features:

Frequency counts for 'Target':

Target
Graduate    2209
Dropout     1421
Enrolled     794
Name: count, dtype: int64


Potential outliers in 'Age at enrollment':

{"summary":"{\n  \"name\": \"print(\\\"The dataset contains
information about student performance and demographics, with a target
```

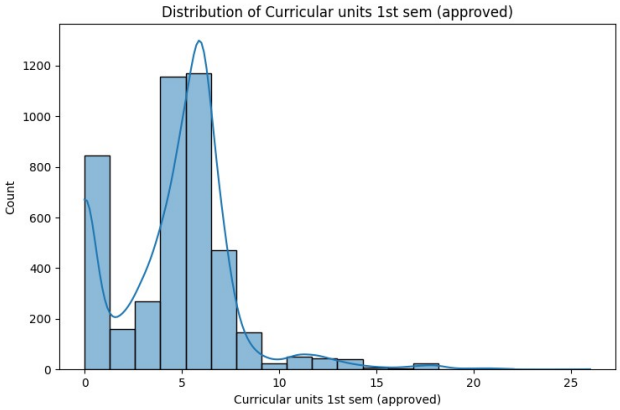
```
variable indicating dropout or graduation\",\n  \"rows\": 441,\n
\"fields\": [\n    {\n      \"column\": \"Age at enrollment\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
6,\n        \"min\": 35,\n        \"max\": 70,\n
\"num_unique_values\": 28,\n        \"samples\": [\n          47,\n
59,\n          36\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    }\n  ]\n}","type":"dataframe"}
```

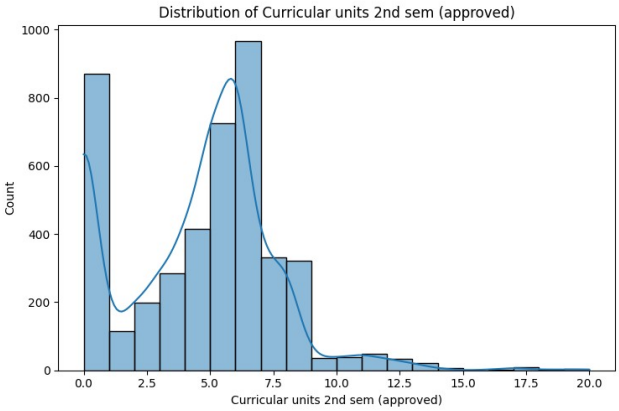

Summary of observations:
The dataset contains information about student performance and demographics, with a target variable indicating dropout or graduation. Some numerical features, such as age at enrollment, show potential outliers. Further analysis is needed.

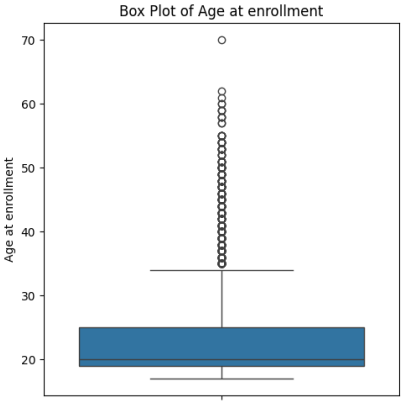
```
import matplotlib.pyplot as plt
import seaborn as sns

# Histograms for numerical features
plt.figure(figsize=(15, 10))
numerical_features = ['Age at enrollment', 'Curricular units 1st sem
(approved)', 'Curricular units 2nd sem (approved)']
for i, col in enumerate(numerical_features):
    plt.subplot(2, 2, i + 1)
    sns.histplot(df[col], bins=20, kde=True)
    plt.title(f'Distribution of {col}')
plt.tight_layout()
plt.show()

# Box plots for numerical features
plt.figure(figsize=(15, 5))
for i, col in enumerate(numerical_features):
  plt.subplot(1, 3, i + 1)
  sns.boxplot(y=df[col])
  plt.title(f'Box Plot of {col}')
plt.tight_layout()
plt.show()


# Bar chart for 'Target'
plt.figure(figsize=(8, 6))
sns.countplot(x='Target', data=df)
plt.title('Distribution of Target Variable')
plt.show()

# Scatter plots for numerical features
plt.figure(figsize=(10, 6))
```

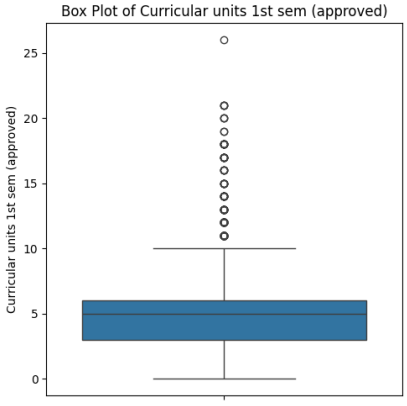
```
sns.scatterplot(x='Curricular units 1st sem (approved)', y='Curricular units 2nd sem (approved)', hue='Target', data=df)
plt.title('Relationship between Curricular Units Approved in 1st and 2nd Semesters')
plt.show()

plt.figure(figsize=(10, 6))
sns.scatterplot(x='Age at enrollment', y='Curricular units 1st sem (approved)', hue='Target', data=df)
plt.title('Relationship between Age at Enrollment and Curricular Units Approved in 1st Semester')
plt.show()
```

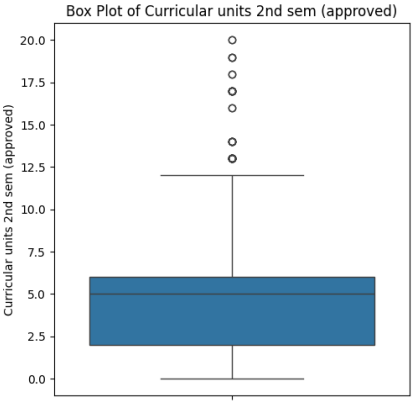


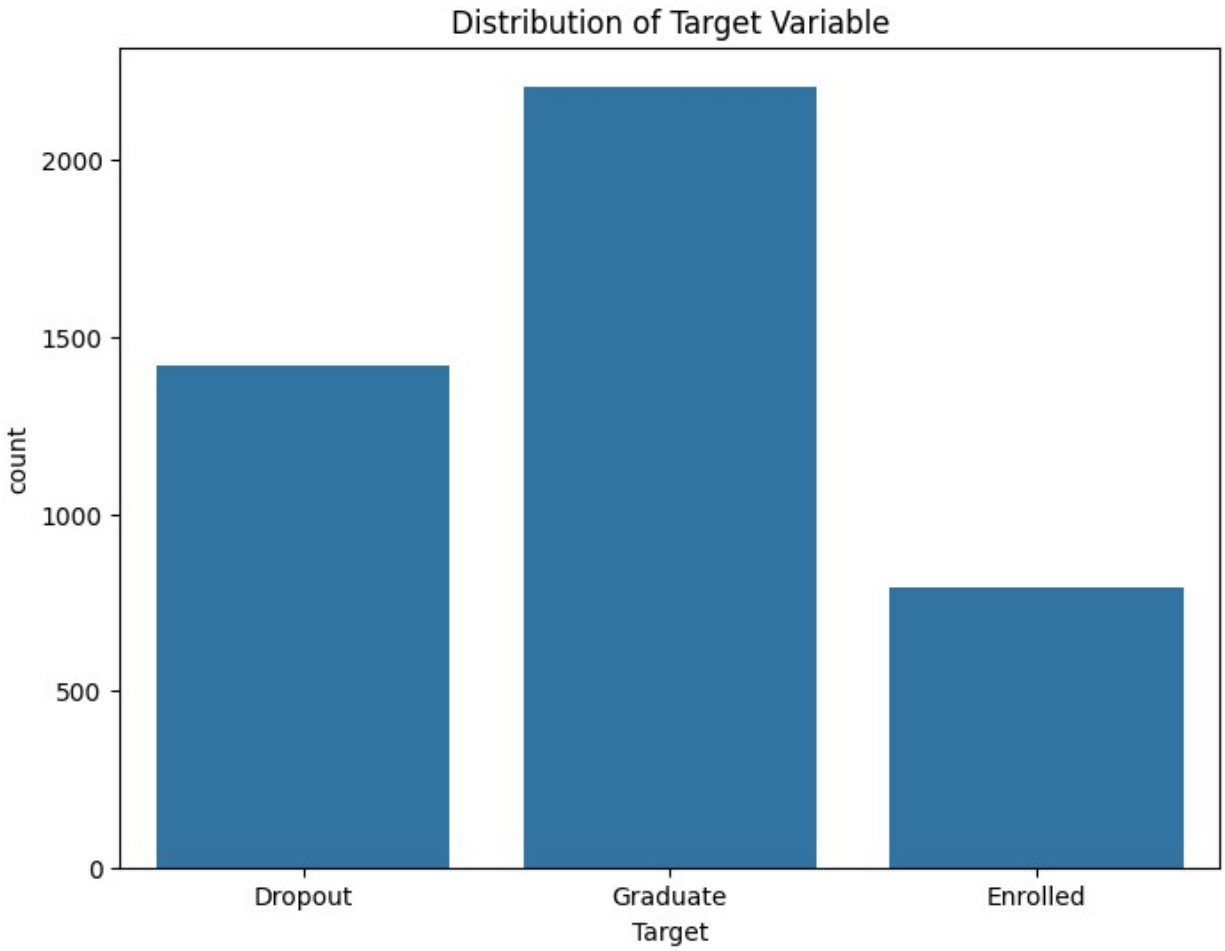


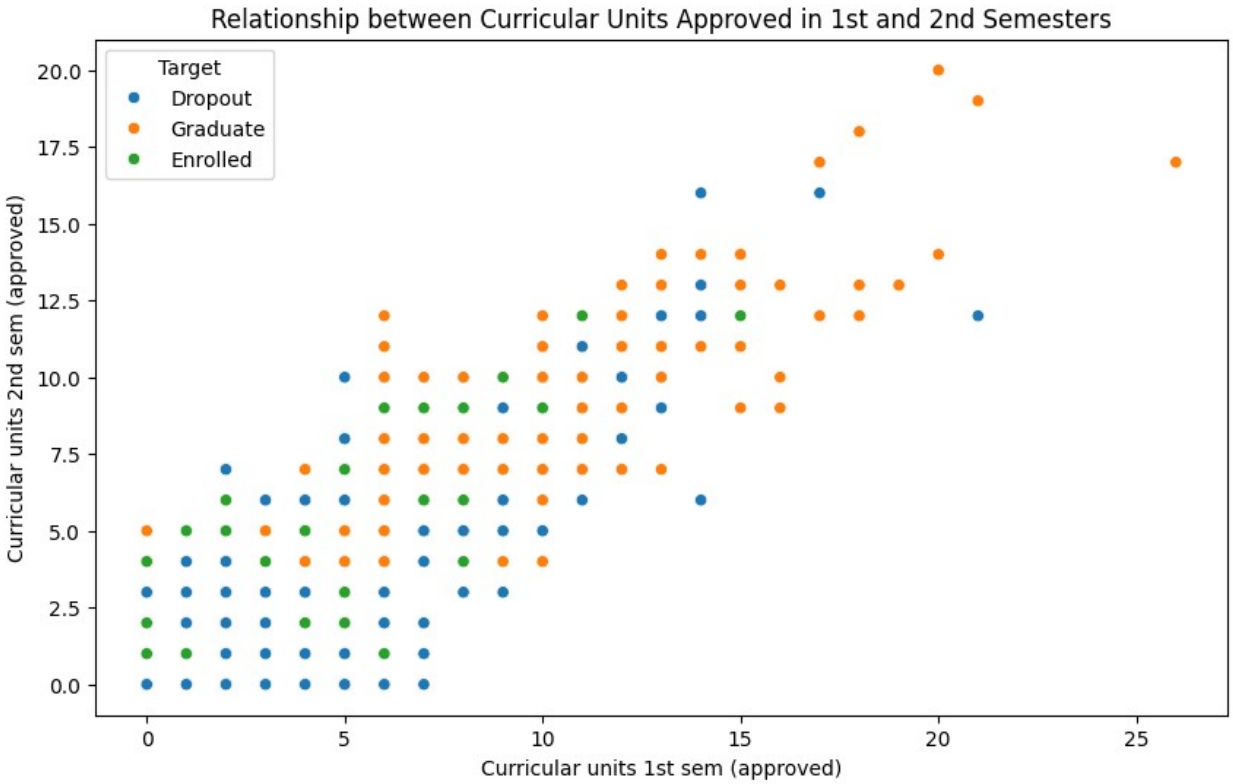


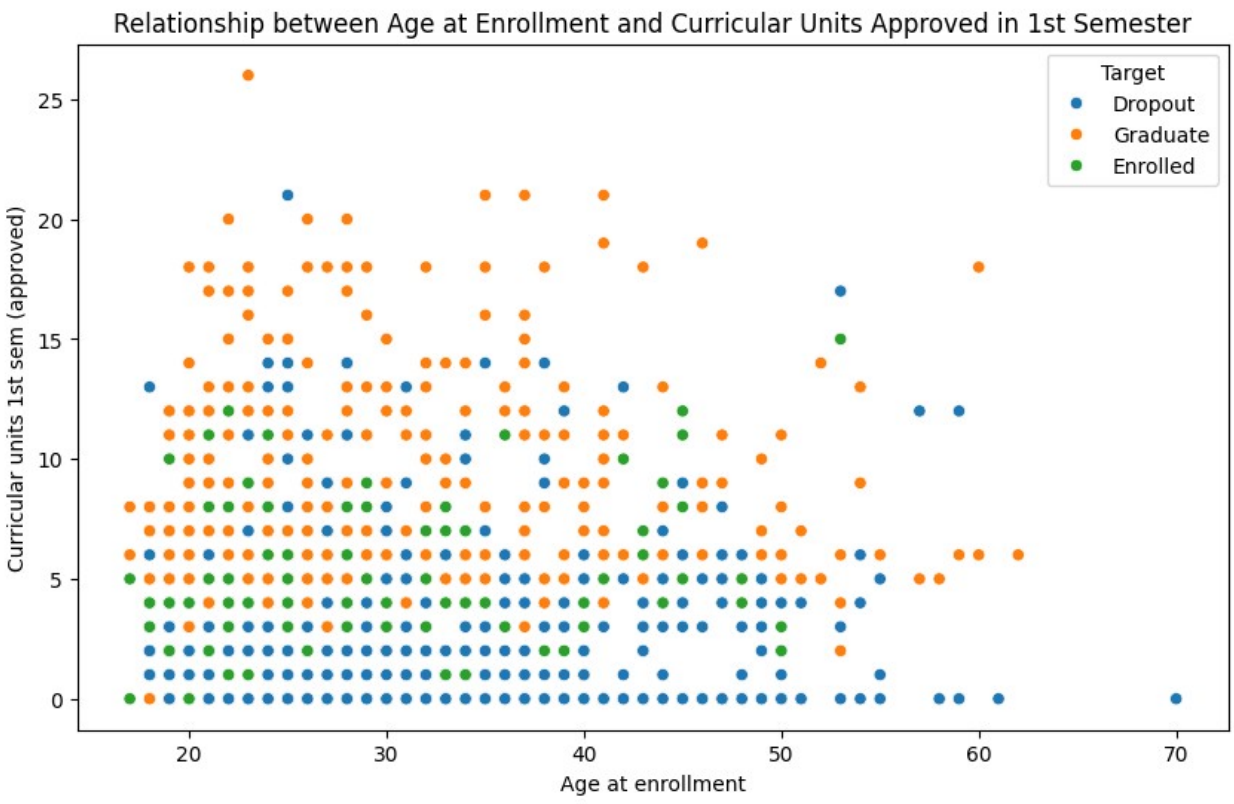




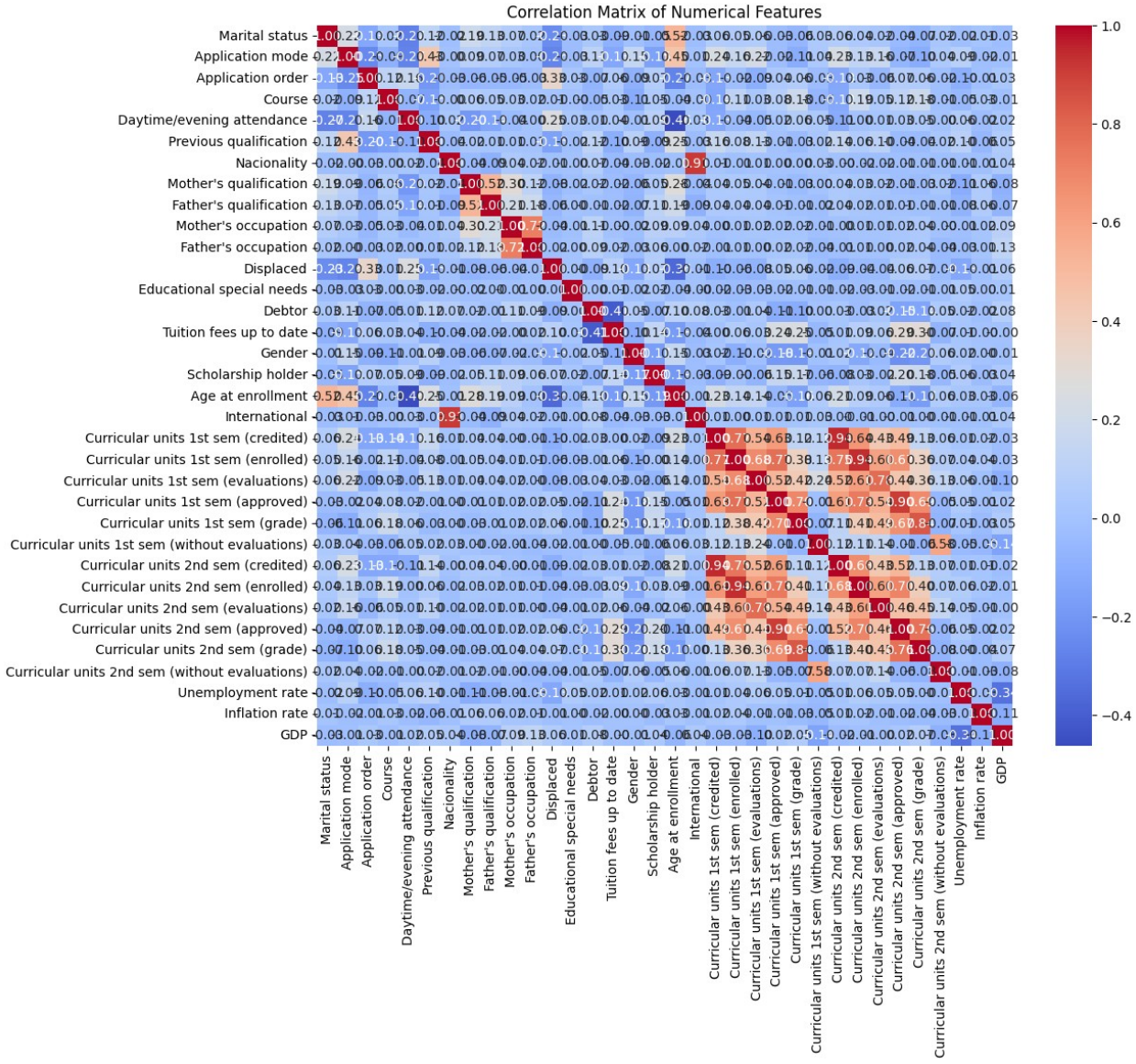



Distribution of Target Variable
count
2000
1500
1000
500
0
Dropout
Graduate
Enrolled
Target

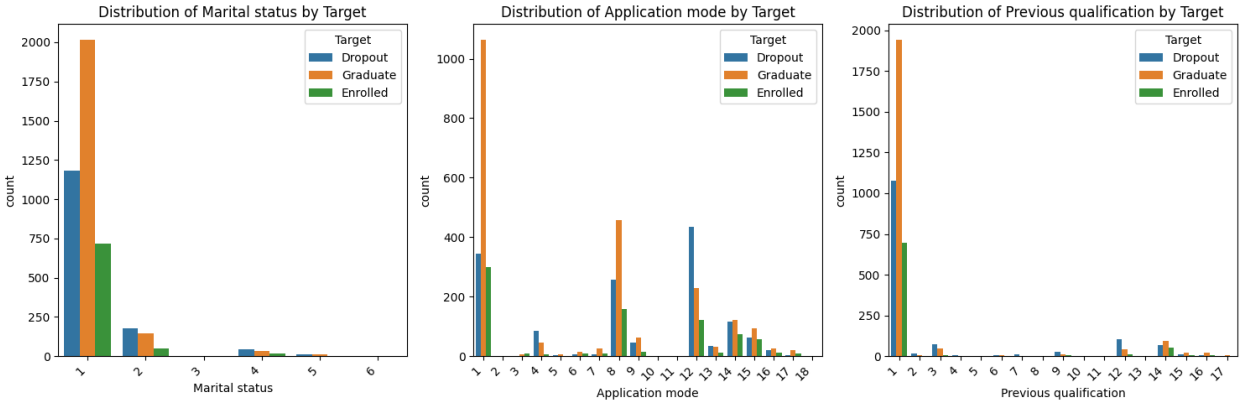

Relationship between Curricular Units Approved in 1st and 2nd Semesters
Target
Dropout
Graduate
Enrolled
Curricular units 2nd sem (approved)
Curricular units 1st sem (approved)
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
0
5
10
15
20
25


Relationship between Age at Enrollment and Curricular Units Approved in 1st Semester
Target
Dropout
Graduate
Enrolled
Curricular units 1st sem (approved)
Age at enrollment
0
5
10
15
20
25
20
30
40
50
60
70

```
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd

# Correlation Analysis
numerical_features = df.select_dtypes(include=['number']).columns
correlation_matrix = df[numerical_features].corr()
plt.figure(figsize=(12, 10))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm',
fmt=".2f")
plt.title('Correlation Matrix of Numerical Features')
plt.show()

# Grouped Analysis
grouped_data = df.groupby('Target')[numerical_features].agg(['mean',
'median', 'std'])
display(grouped_data)


# Categorical Feature Analysis
categorical_cols = ['Marital status', 'Application mode', 'Previous
qualification'] # Example categorical features
plt.figure(figsize=(15, 5))
for i, col in enumerate(categorical_cols):
    plt.subplot(1, len(categorical_cols), i + 1)
    sns.countplot(x=col, hue='Target', data=df)
    plt.xticks(rotation=45, ha='right')
    plt.title(f'Distribution of {col} by Target')
plt.tight_layout()
plt.show()

# Additional Analysis (example: Unemployment rate vs. Dropout rate)
# Create a new column indicating whether a student dropped out or not
df['Dropped_Out'] = df['Target'] == 'Dropout'
# Calculate the mean unemployment rate for each group
unemployment_by_dropout = df.groupby('Dropped_Out')['Unemployment
rate'].mean()
print(unemployment_by_dropout)
```



{"type":"dataframe","variable_name":"grouped_data"}



```
Dropped_Out
False    11.542358
True     11.616397
Name: Unemployment rate, dtype: float64
```

```
import scipy.stats as stats
from scipy.stats import chi2_contingency

# Additional Analysis: Unemployment Rate and Dropout Rate (already
done in the previous step)

# Additional Analysis: Qualification and Final Result
# Example: 'Previous qualification' vs 'Target'
contingency_table = pd.crosstab(df['Previous qualification'],
df['Target'])
chi2, p, dof, expected = chi2_contingency(contingency_table)
print(f"\nChi-squared test for 'Previous qualification' vs 'Target':")
print(f"Chi2 statistic: {chi2}")
print(f"P-value: {p}")
print(f"Degrees of freedom: {dof}")

#More detailed analysis on previous qualification
qualifications_dropout_rates = df.groupby('Previous qualification')
['Dropped_Out'].mean()
print("\nDropout rates by previous qualification:\n",
qualifications_dropout_rates)

# Further analysis can be performed based on the above results.
# Example:  Plot a bar chart of dropout rates by previous
qualification.
plt.figure(figsize=(10, 6))
qualifications_dropout_rates.plot(kind='bar')
plt.title('Dropout Rates by Previous Qualification')
plt.xlabel('Previous Qualification')
plt.ylabel('Dropout Rate')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```

```
Chi-squared test for 'Previous qualification' vs 'Target':
Chi2 statistic: 219.68070897587953
P-value: 7.160305160682533e-30
Degrees of freedom: 32

Dropout rates by previous qualification:
 Previous qualification
1     0.290019
2     0.695652
```

```
3     0.595238
4     0.500000
5     1.000000
6     0.437500
7     1.000000
8     0.750000
9     0.577778
10    1.000000
11    0.500000
12    0.641975
13    0.428571
14    0.315068
15    0.350000
16    0.166667
17    0.333333
Name: Dropped_Out, dtype: float64
```