STUDENT RETENTION

Presented by Advith Reddy Kunda ● Ajith Kumar Sukumar ● Narayan Rohith Reddy Pasham ● Zeina Kamal

Group G | University of South Florida

Overview

- Business Question
- Method & Purpose
- Dataset
- Descriptive Analysis
- Exploratory Analysis
- Data Prep
- Models
- Results
- Conclusion
- Thank You

Business Question

In today's educational landscape, student retention and success is of utmost importance for educational institutions. Identifying students who are at risk of dropping out and implementing timely interventions can significantly contribute to improving graduation rates and ensuring academic success.

What are certain factors that may affect students' retention or drop out in academic institutions?

Method & Purpose

This project aims to develop a predictive model using machine learning classification algorithms to identify students who are likely to drop out. By leveraging data on student demographics, academic performance, socio-economic factors, and other relevant variables, the aim is to build a robust predictive model that can effectively forecast the likelihood of students dropping out.

Predicting the likelihood of a student dropping out will enable universities to provide support and resources to those students to improve retention rates

Dataset

It encompasses a wide range of information about students in higher education institutions, such as their demographics, socioeconomic backgrounds, and academic performance information that can be used to analyze the possible predictors of student dropout and academic success.

Descriptive Analysis

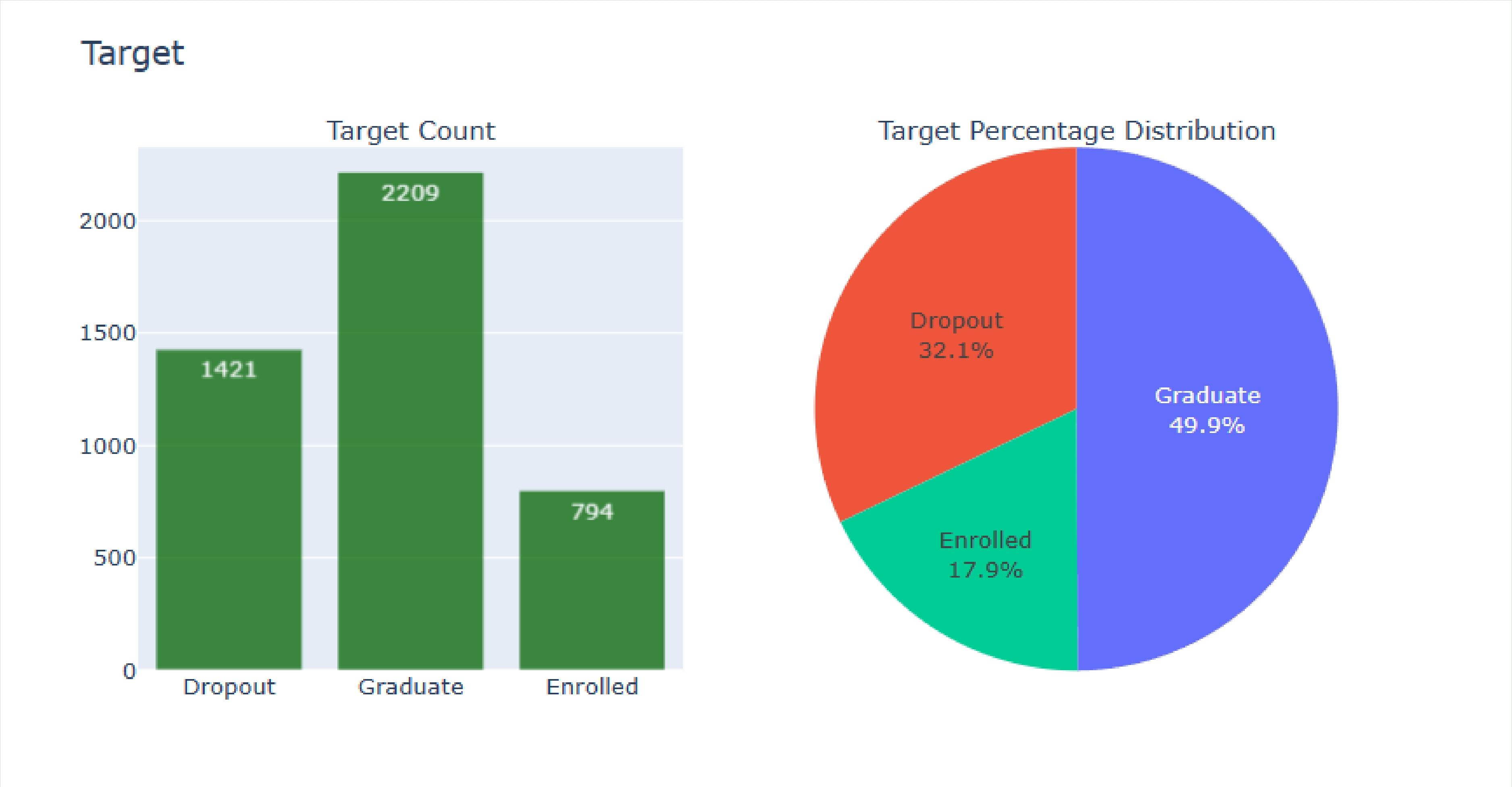
We have 4,424 observations (rows) and 35 features (Columns)

The majority of categorical variables in the downloaded dataset have already been converted to numerical format. However, for the purpose of exploratory data analysis, we will revert certain columns to their original categorical form.

```
In [44]:  # shape of data
          data.shape

Out[44]:  (4424, 35)
```

```
In [45]:  data.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   Nationality                                     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
```

Exploratory Analysis



Target Variable

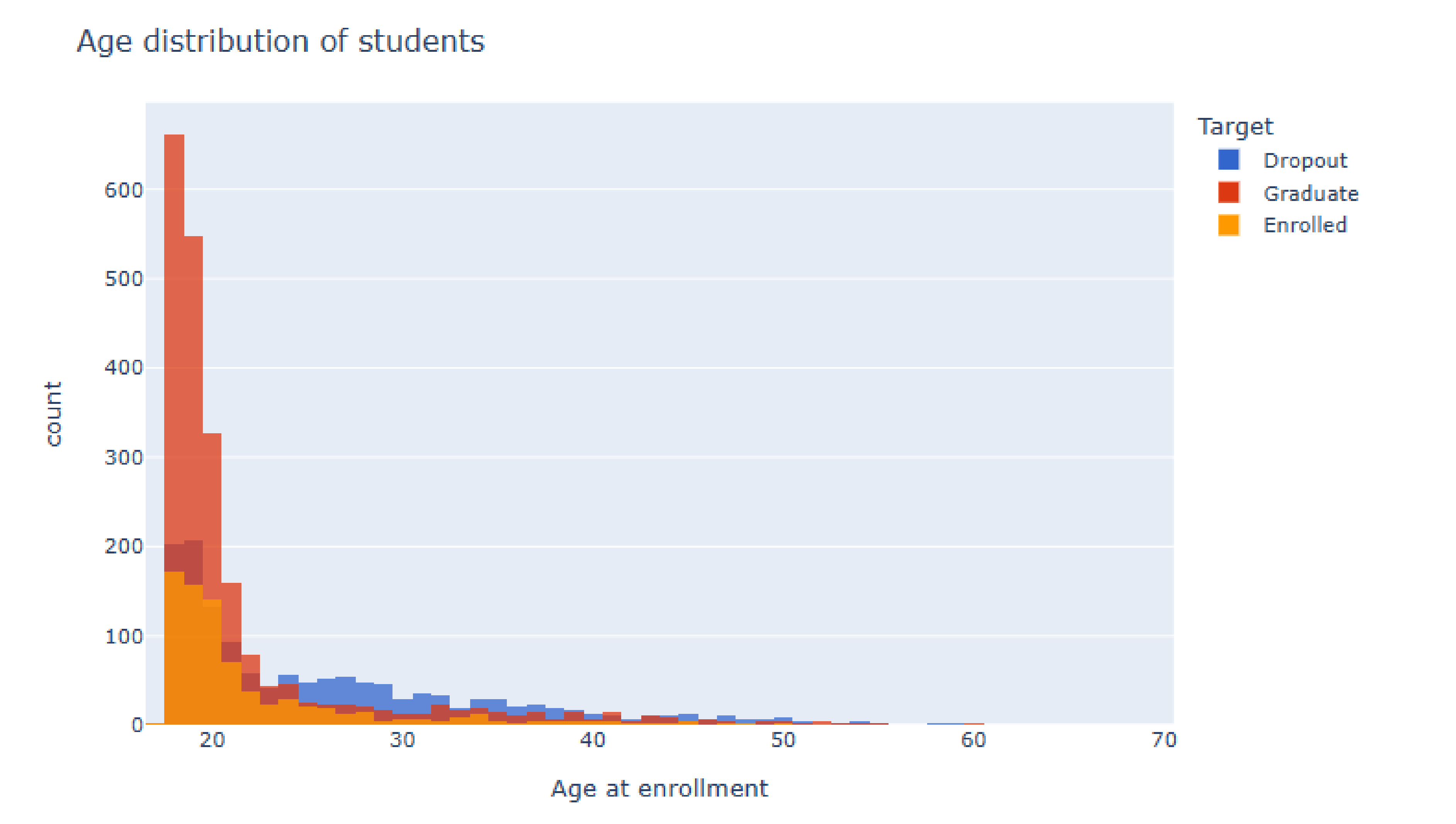
From the target column we can infer the following:

- Dropout: This means for that particular observation, the student dropped out
- Graduate: The student is a graduate
- Enrolled: The student is currently enrolled

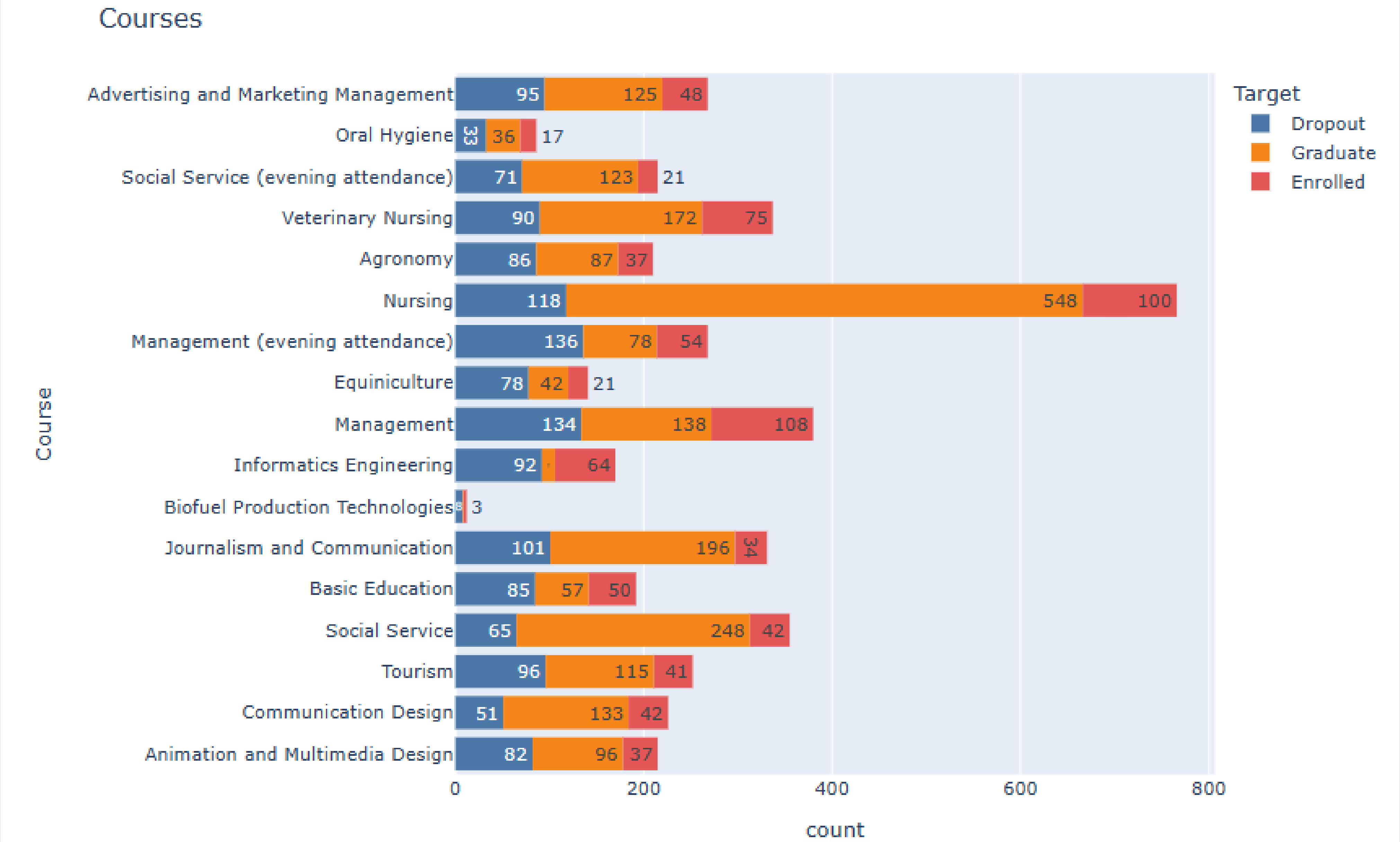
Exploratory Analysis

Age

- Distribution shows that majority of the students are in their late teen's to early 20's
- It is also observed that there was an increase in dropout rate from mid 20's to early 30's



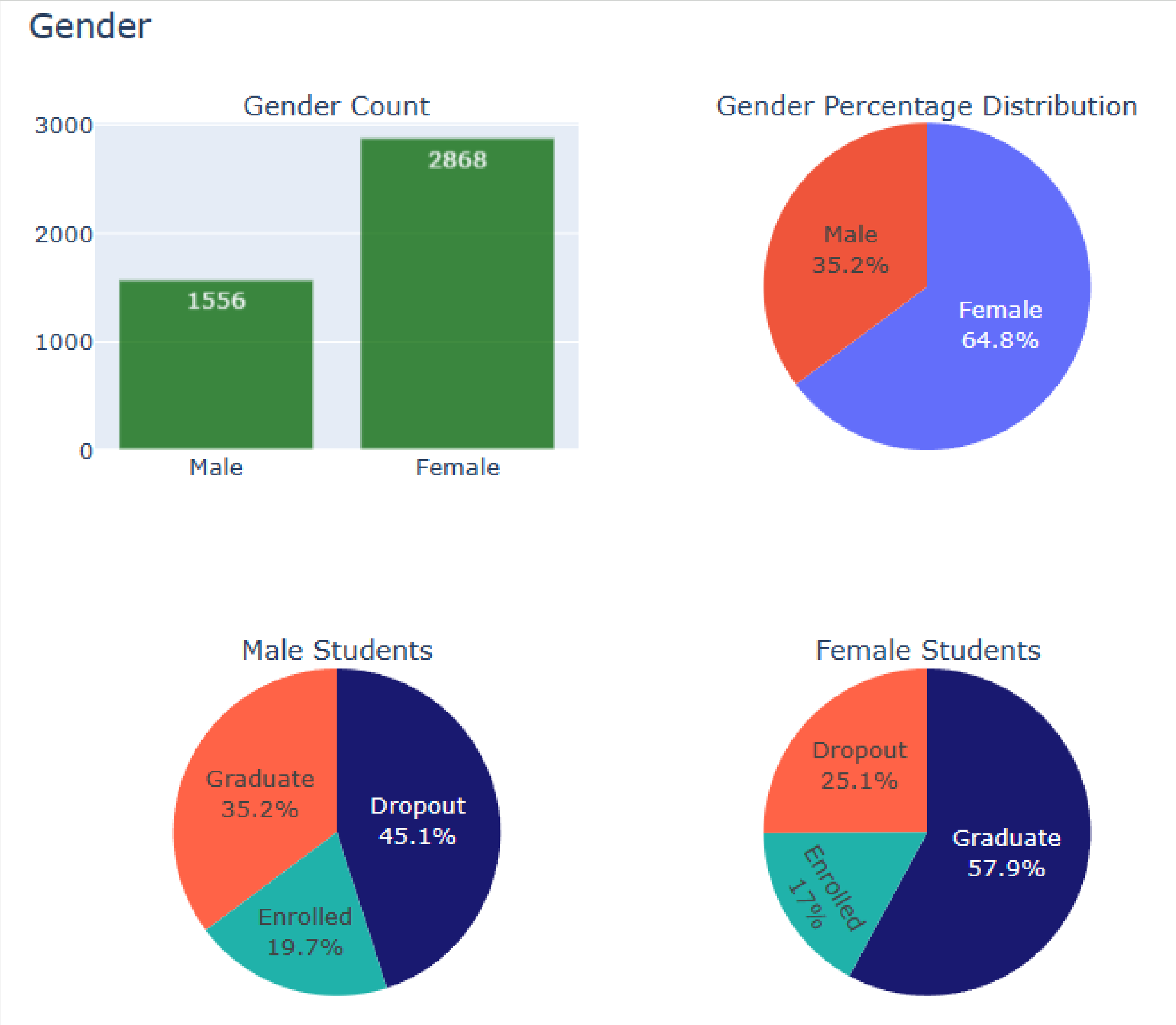
Exploratory Analysis



Courses

The course that had the highest number of dropouts was Management with evening attendance (136)

Exploratory Analysis



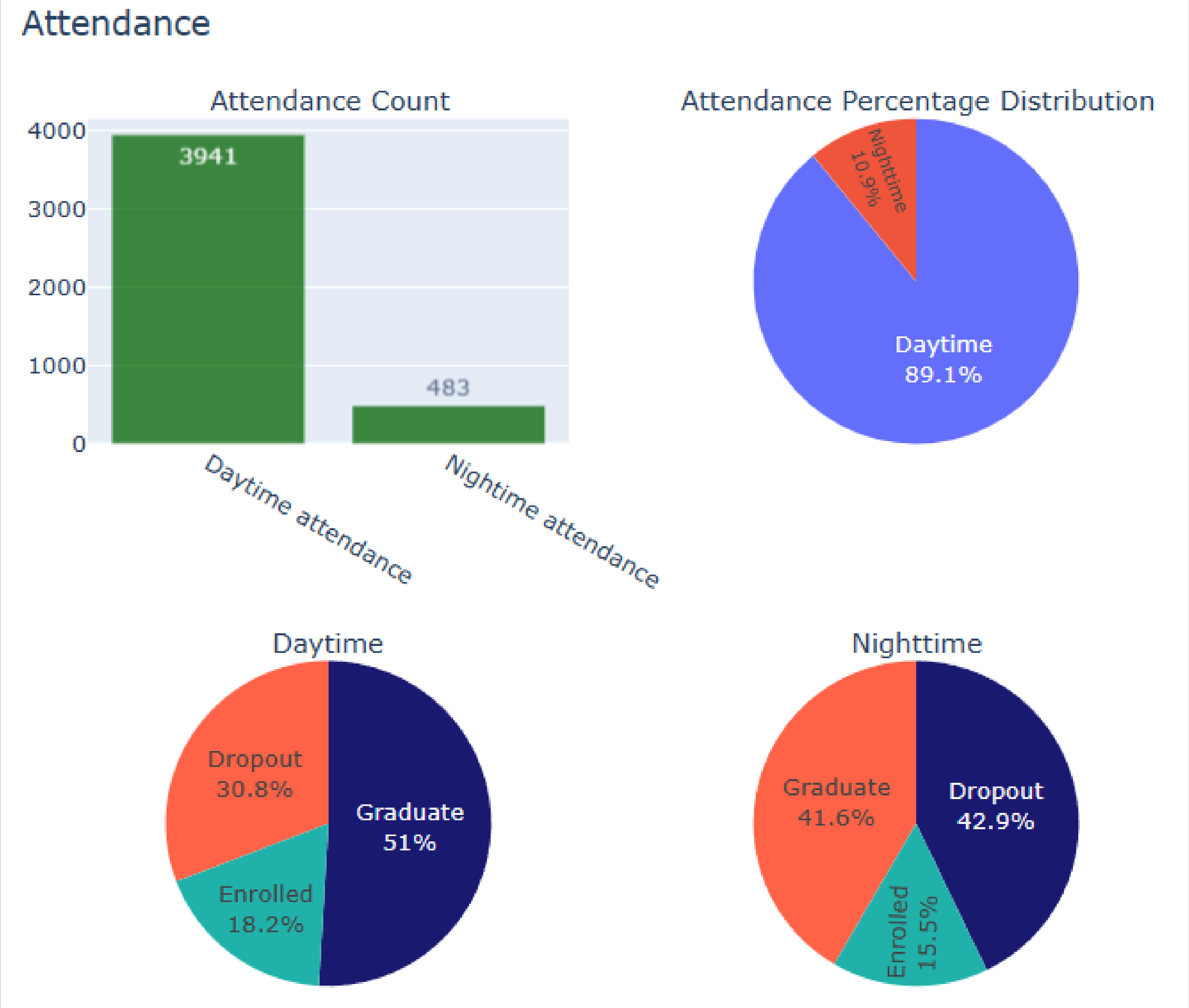
Gender

- There was a significant number of female students (64.8%) compared to the males (35.2%).
- Also it is observed that there was a higher rate of dropout students that were male (45.1%), compared to the females (25.1%).

Exploratory Analysis

Attendance

Vast Majority (89.1%) of the students attended daytime classes, however, there seems to be about 12.1% increase in the dropout rate for nighttime students compared to the day time students.

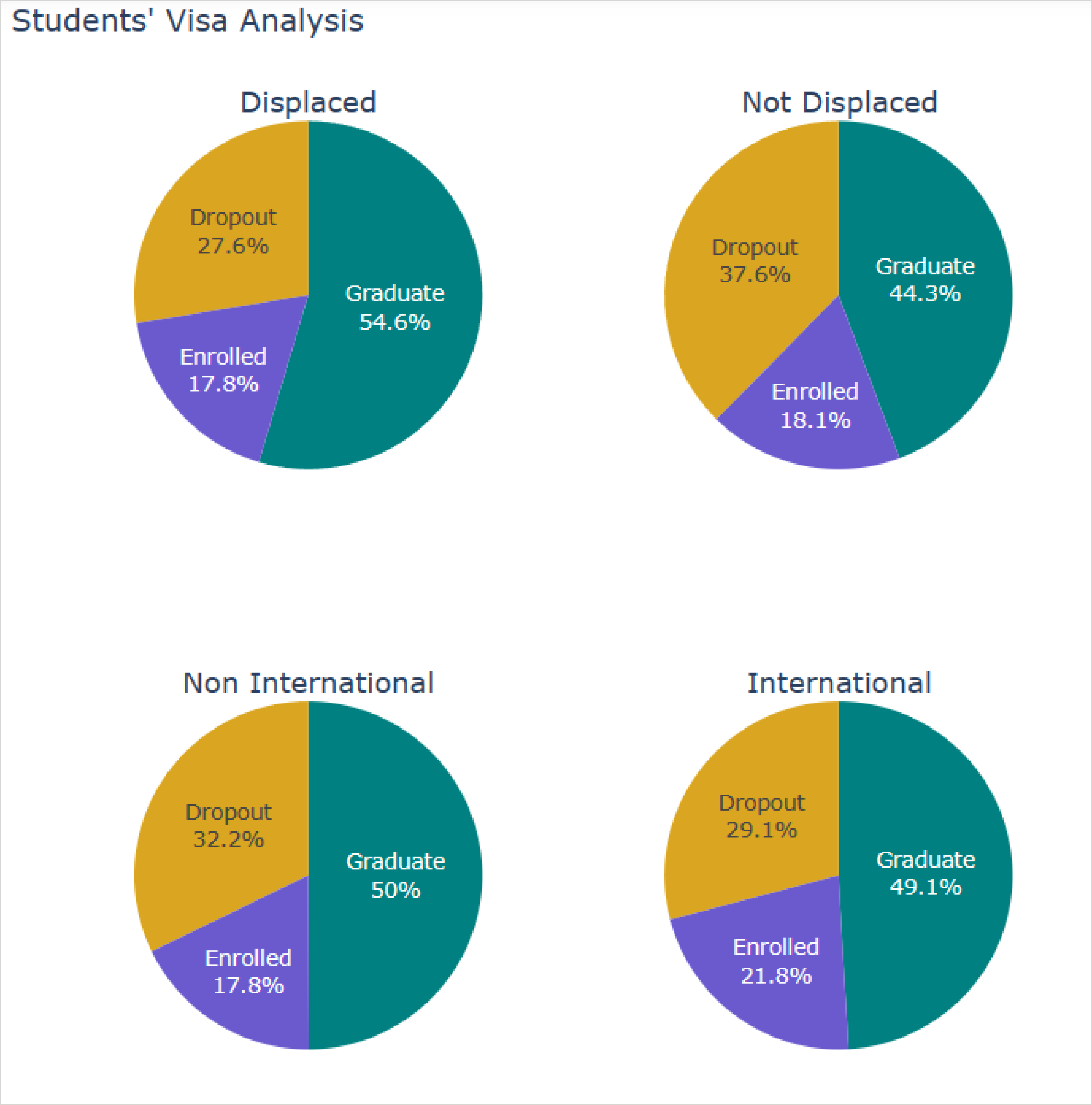


Exploratory Analysis

Marital Status

- Vast majority of the students are single, however 30.2% of single students dropout.
- Another thing to note is that legally separated students (66.7%) had the highest percentage of dropouts followed by Married students (47.2%).

Exploratory Analysis



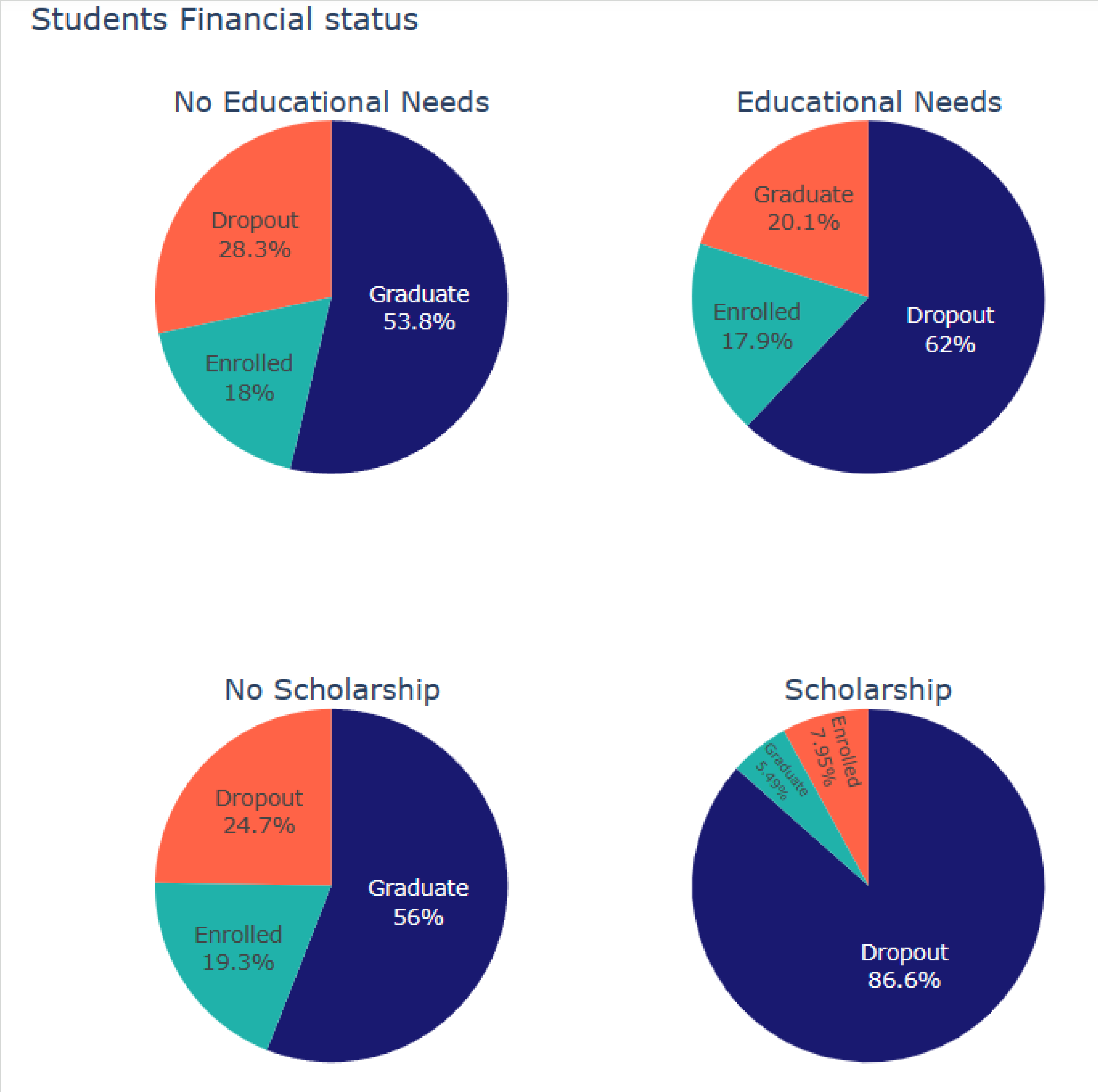
Visa Status

From the pie chart, we can see students who were not displaced had a higher dropout rate (37.6%) compared to students who were displaced (27.6%) Also, non-international students had a higher dropout rate of 32.2% compared to international students who had 29.1%.

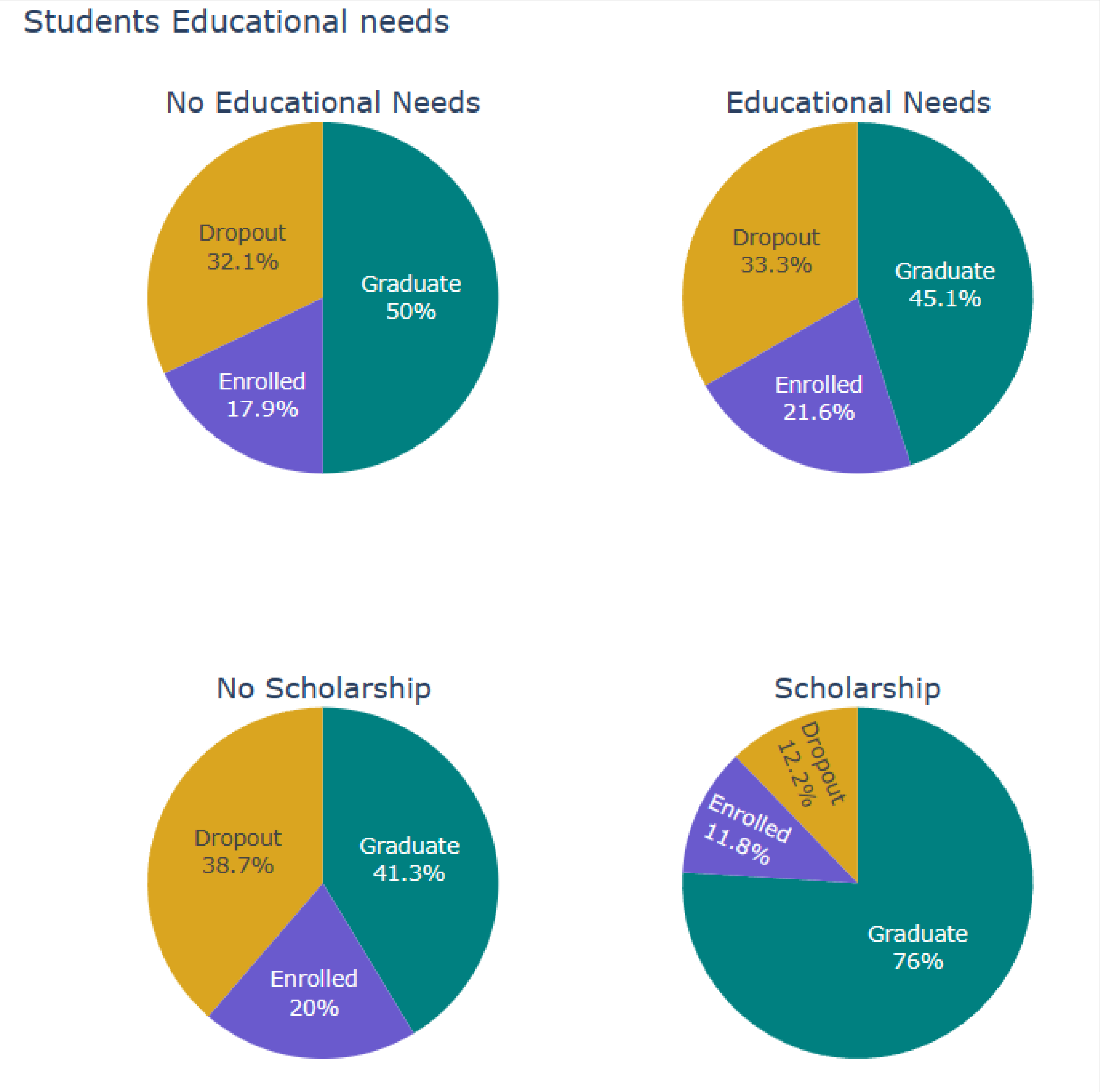
Exploratory Analysis

Financial Status

Unsurprisingly, students who were in debt and had not completed payment for tuition had a higher dropout rate of 62% and 86.6% respectively.



Exploratory Analysis



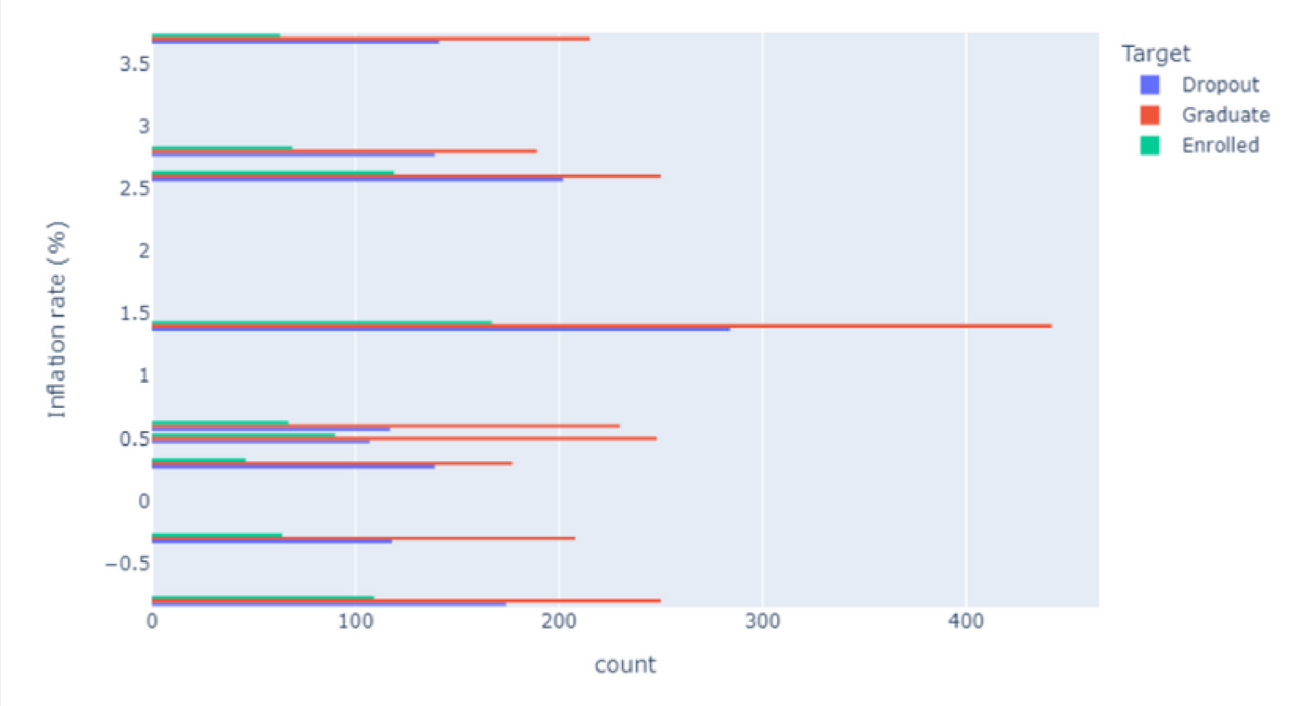
Educational Needs

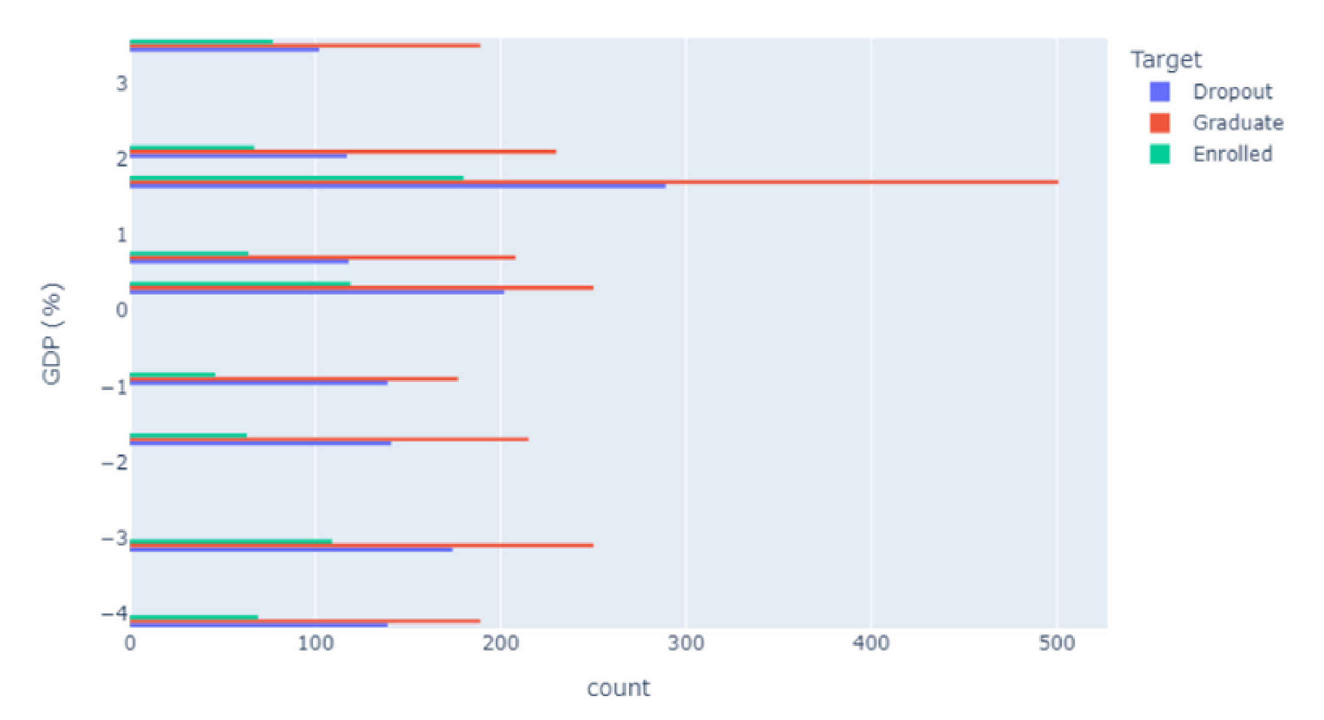
Similarly, students who were granted scholarships had a low dropout rate of 12.2% compared to those who were not given (38.7%). The educational needs of the students didn't seem to be a significant factor because students with and without educational needs had a 33.3% and 32.1% dropout rate respectively. The column would be dropped before training the model

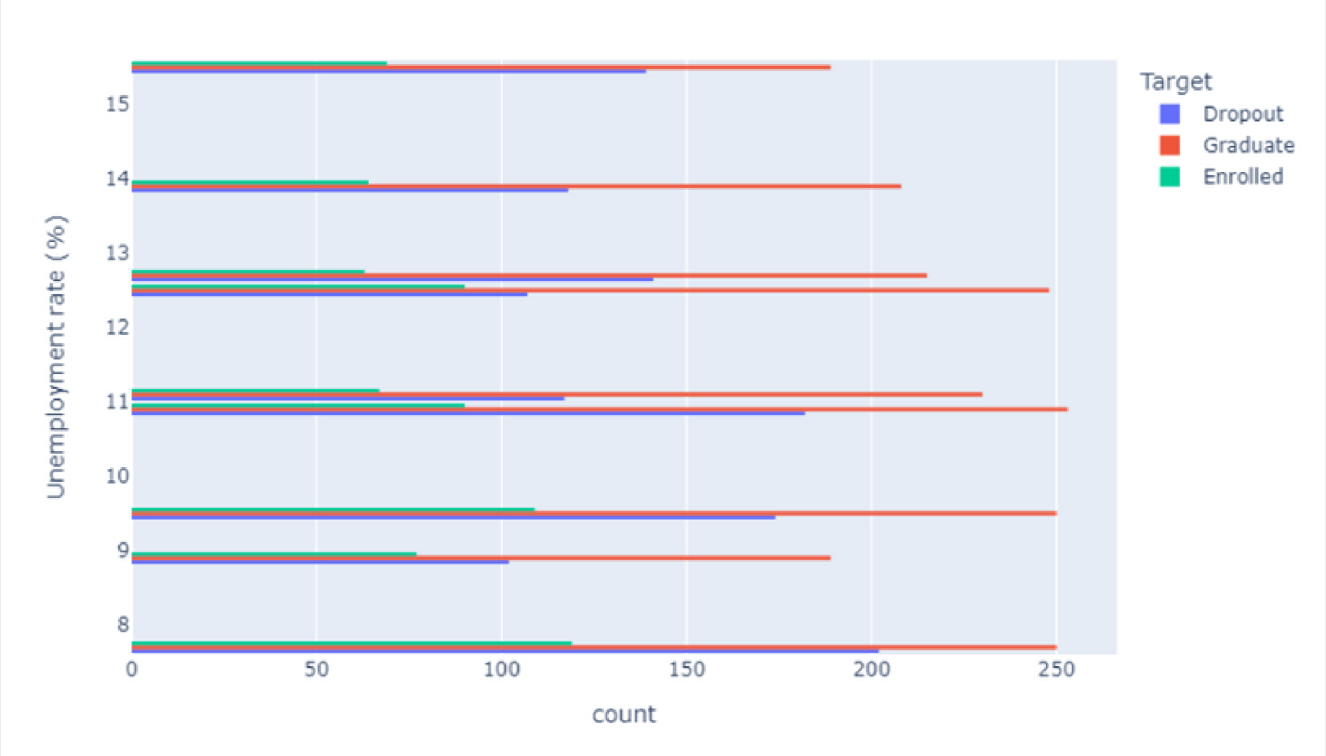
Exploratory Analysis

Economic Factors

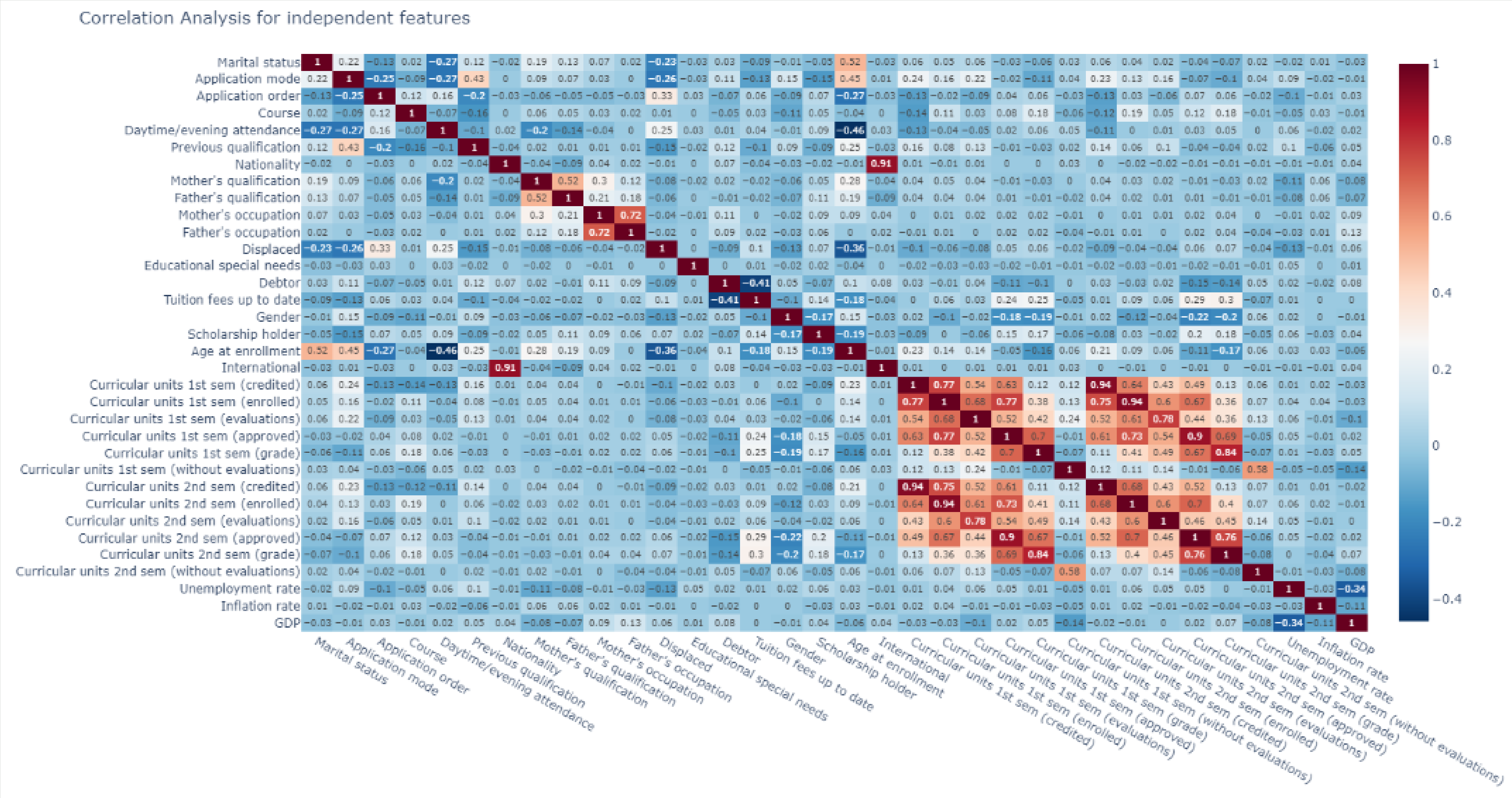
The economic factors does not seem to have an effect on the dropout rate revealing no pattern or meaningful insight.







Exploratory Analysis



Features Selection

From the results we can see some features have strong correlation with each other:

- Nationality and International
- Mother's qualification and Father's qualification
- Mother's occupation and Father's occupation
- Curricular Units 1st Sem. and Curricular Units 2nd Sem.

Data Prep

Data Processing

In [194]:

```python
#Get dummies for Target columns
dummies = pd.get_dummies(data['Target'])

#Drop all columns except that for Dropout
dummies.drop(['Enrolled',	'Graduate'], axis = 1, inplace= True)
data['Target'] = dummies
data.head()
```

Out[194]:

	Marital status	Application mode	Application order	Course	Daytime/evening attendance	Previous qualification	Nationality	Mother's qualification	Father's qualification	Mother's occupation	Father's occupation	Displaced	Educ
0	1	8	5	2	1	1	1	13	10	6	10	1	
1	1	6	1	11	1	1	1	1	3	4	4	1	
2	1	1	5	5	1	1	1	22	27	10	10	1	
3	1	8	2	15	1	1	1	23	27	6	4	1	
4	2	12	1	3	0	1	1	22	28	10	10	0	

Data Prep

Normalizing Data

```
In [195]: Y = np.array(data['Target'])
          X_features = data.drop('Target', axis = 1)
          X_features.head()
```

Out[195]:

	Marital status	Application mode	Application order	Course	Daytime/evening attendance	Previous qualification	Nationality	Mother's qualification	Father's qualification	Mother's occupation	Father's occupation	Displaced	Educ
0	1	8	5	2	1	1	1	13	10	6	10	1	
1	1	6	1	11	1	1	1	1	3	4	4	1	
2	1	1	5	5	1	1	1	22	27	10	10	1	
3	1	8	2	15	1	1	1	23	27	6	4	1	
4	2	12	1	3	0	1	1	22	28	10	10	0	

```
In [197]: Y[:5]

Out[197]: array([1, 0, 1, 0, 0], dtype=uint8)

In [198]: scaler =  StandardScaler()
          X = scaler.fit_transform(X_features)
          X

Out[198]: array([[-0.29482875,  0.21006857,  2.49089589, ..., -0.28763846,
                   0.12438647,  0.76576084],
                 [-0.29482875, -0.16740639, -0.55406775, ...,  0.87622207,
                  -1.10522155,  0.34719942],
                 [-0.29482875, -1.11109377,  2.49089589, ..., -0.28763846,
                   0.12438647,  0.76576084],
                 ...,
                 [-0.29482875, -1.11109377, -0.55406775, ...,  0.87622207,
                  -1.10522155,  0.34719942],
                 [-0.29482875, -1.11109377, -0.55406775, ..., -0.81325289,
                  -1.46687097, -1.37551124],
                 [-0.29482875, -0.35614386, -0.55406775, ...,  0.42569541,
                   1.7879738 , -0.74987207]])
```

Models

Logistic Regression 01

```
In [257]:
# Train a logistic regression model
lr_model = LogisticRegression()
lr_model.fit(X_train, y_train)

# Predict target values for test data
y_pred = lr_model.predict(X_test)

# Confusion Matrix
lr_matrix = confusion_matrix(y_test, y_pred)

# Evaluate the model's accuracy
lr_acc = round(accuracy_score(y_test, y_pred), 3)
print(f'Accuracy of logistic regression model is {lr_acc * 100}%')

Accuracy of logistic regression model is 85.6%
```

Decision Trees 02

```
In [258]:
# Train a Decision tree model
tree_model = DecisionTreeClassifier()
tree_model.fit(X_train, y_train)

# Predict target values for test data
y_pred = tree_model.predict(X_test)

# Confusion Matrix
tree_matrix = confusion_matrix(y_test, y_pred)

# Evaluate the model's accuracy
tree_acc = round(accuracy_score(y_test, y_pred), 3)
print(f'Accuracy of Decision tree model is {tree_acc * 100}%')

Accuracy of Decision tree model is 77.2%
```

Models

Support Vector Machines 03

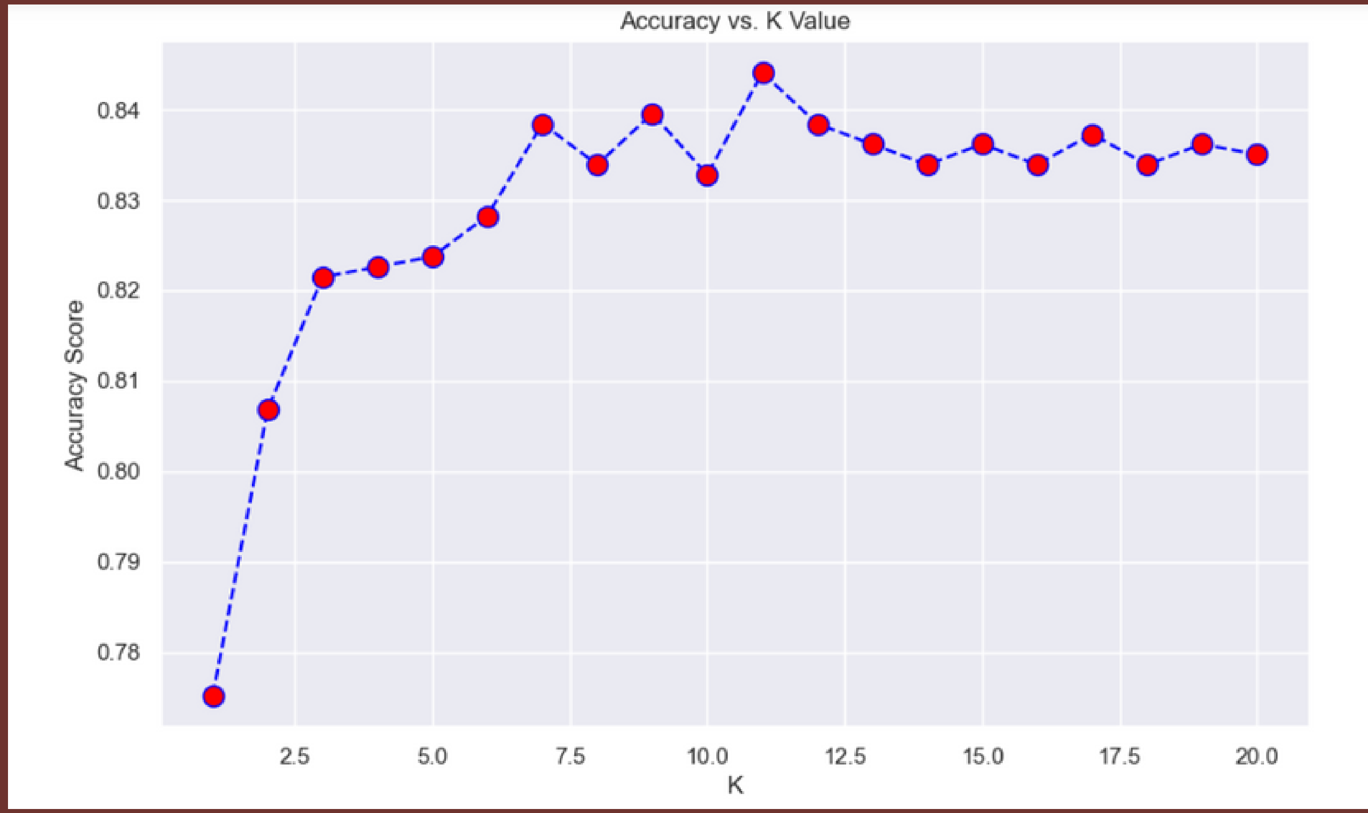
```
In [260]:
# Train a SVC model
svm_model = SVC(kernel = 'rbf')
svm_model.fit(X_train, y_train)

# Predict target values for test data
y_pred = svm_model.predict(X_test)

# Confusion Matrix
svm_matrix = confusion_matrix(y_test, y_pred)

# Evaluate the model's accuracy
svm_acc = round(accuracy_score(y_test, y_pred),3)
print(f'Accuracy of Support vector classifier model is {svm_acc * 100}%')

Accuracy of Support vector classifier model is 86.2%
```

Random Forest 04

```
In [261]:
# Train a Random forest model
rf_model = RandomForestClassifier()
rf_model.fit(X_train, y_train)

# Predict target values for test data
y_pred = rf_model.predict(X_test)

# Confusion Matrix
rf_matrix = confusion_matrix(y_test, y_pred)

# Evaluate the model's accuracy
rf_acc = round(accuracy_score(y_test, y_pred), 3)
print(f'Accuracy of Random forest classifier model is {svm_acc * 100}%')

Accuracy of Random forest classifier model is 86.2%
```

Models

K- Nearest Neighbours 05

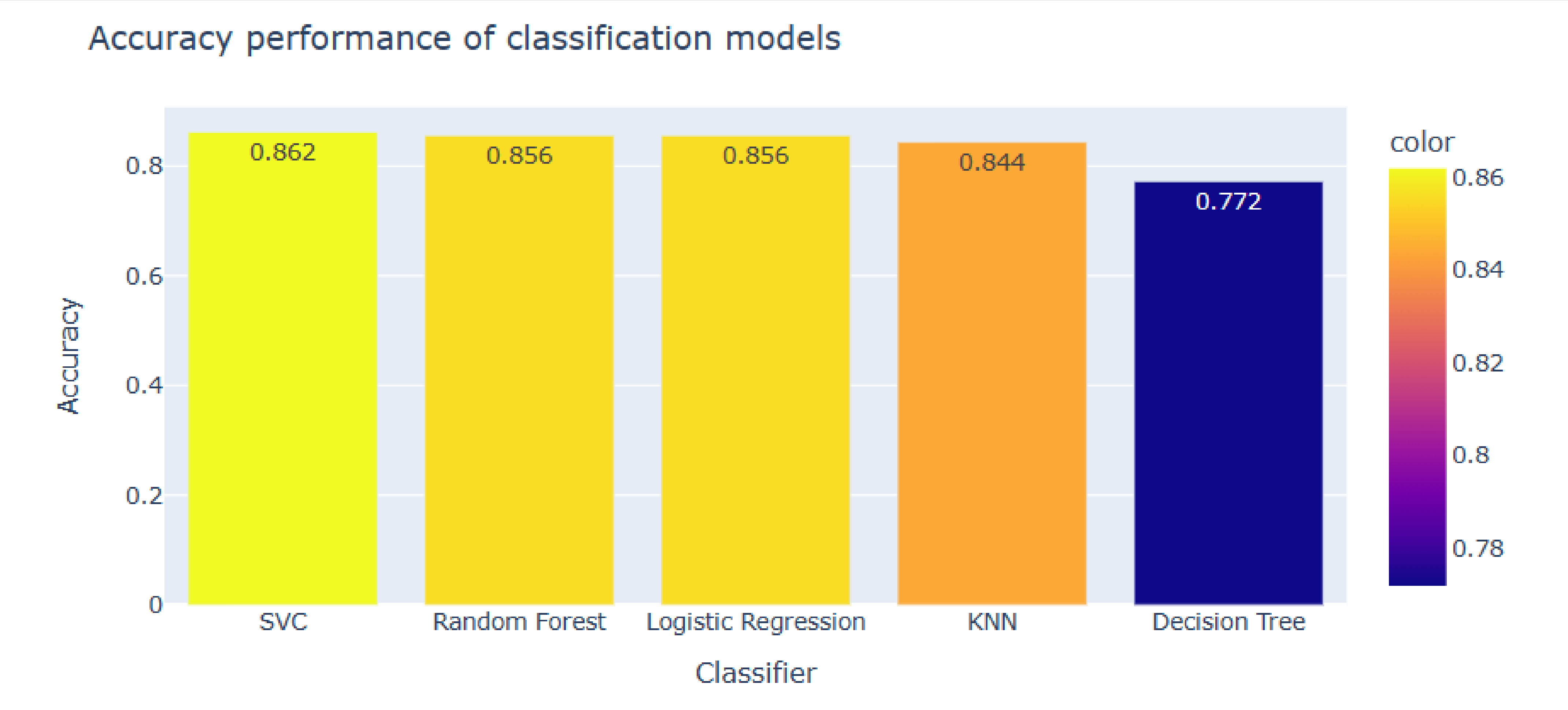



```
In [266]:
# Train a KNN model
knn_model = KNeighborsClassifier(n_neighbors=k)
knn_model.fit(X_train, y_train)

# Predict target values for test data
y_pred = knn_model.predict(X_test)

# Confusion Matrix
knn_matrix = confusion_matrix(y_test, y_pred)

# Evaluate the model's accuracy
knn_acc = round(accuracy_score(y_test, y_pred), 3)
print(f'Accuracy of KNN model is {knn_acc * 100}%')

Accuracy of KNN model is 84.39999999999999%
```

From the result we can determine that the optimal k- value with the highest score 11

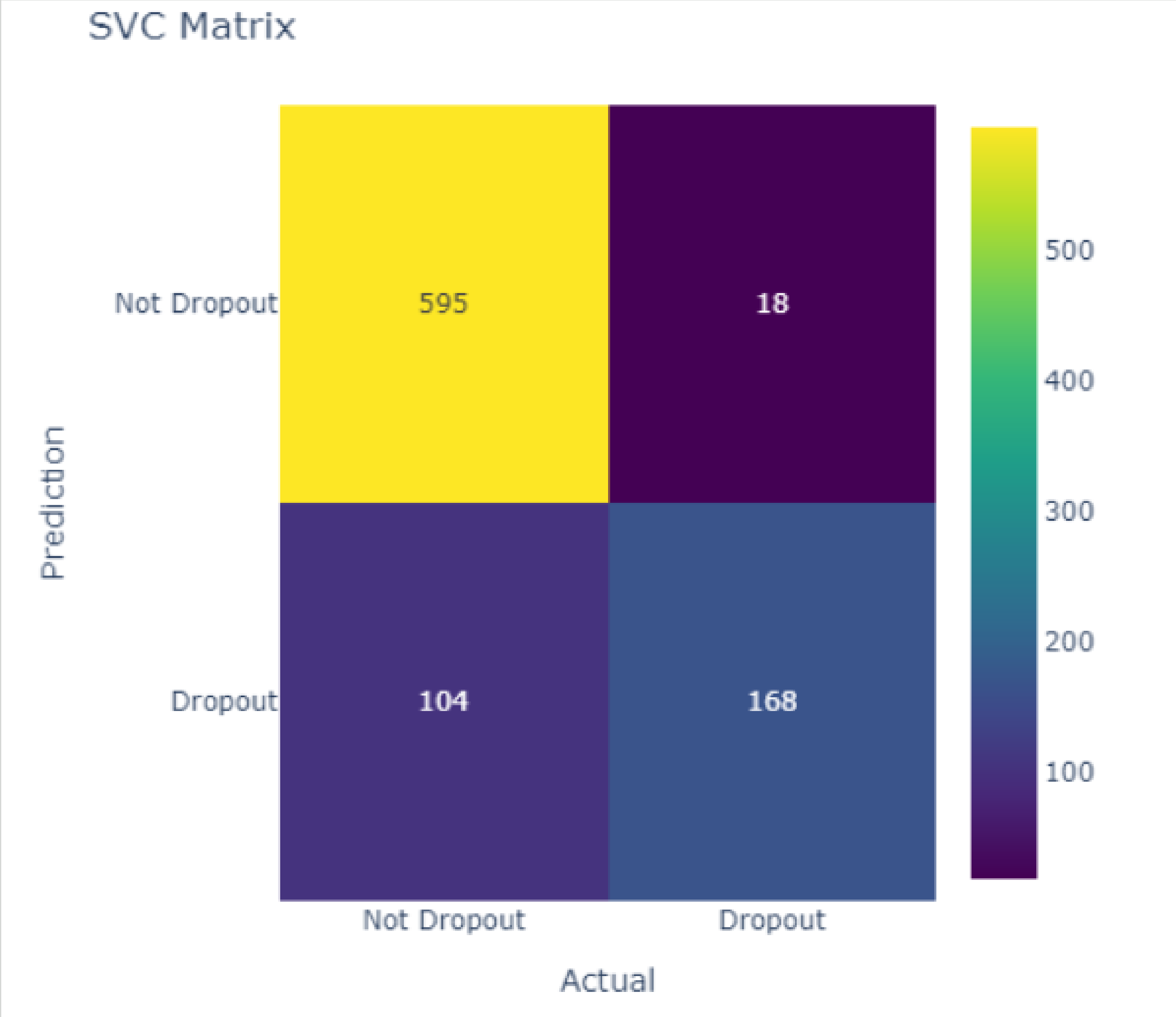
Results

Confusion Matrix



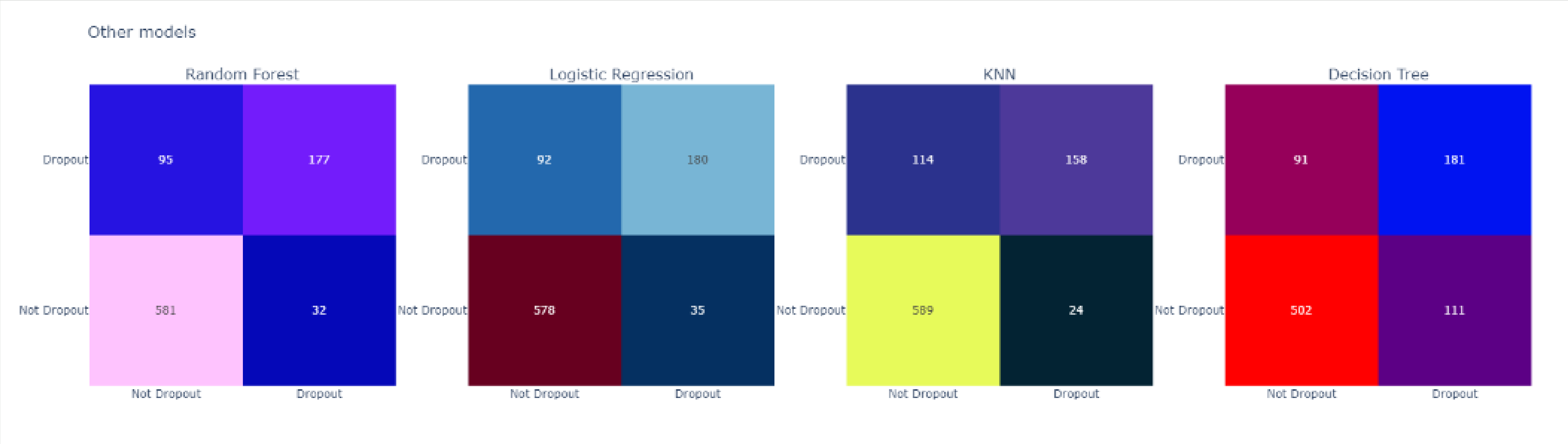
Results

Confusion Matrix



Results

Confusion Matrix



Conclusion

In [276]:
```
from IPython.display import Markdown
Markdown(f"""
#### From the results above we can see that {best_model} perfoms best with the highest accuracy of {round(best_score * 100, 2
```
Out[276]:

From the results above we can see that SVC perfoms best with the highest accuracy of 86.2%

The choice of the best model ultimately hinges on the dataset's unique needs and priorities. In this context, the models are competitive, with the decision leaning toward the Support Vector Classifier for its adaptability to non-linear data and strong accuracy. However, further assessment, such as cross-validation and considering the implications of false positives and false negatives, would provide a more comprehensive basis for selecting the ideal model for the specific application.

THANK YOU