Importing important libraries for analysis

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

In [2]:
```
# Reading the file and displaying
df = pd.read_csv('train.csv')
df.head(2)
```

Out[2]:

	id	Marital status	Application mode	Application order	Course	Daytime/evening attendance	Previous qualification	Previous qualification (grade)	Nacion
0	0	1	1	1	9238	1	1	126.0	
1	1	1	17	1	9238	1	1	125.0	

2 rows × 38 columns

In [3]:
```
# Understanding the dimensions
df.shape
```

Out[3]:
```
(76518, 38)
```

In [4]:
```
# Checking for Null values
df.isnull().sum()
```

Out[4]:
```
id                                              0
Marital status                                  0
Application mode                                0
Application order                               0
Course                                          0
Daytime/evening attendance                      0
Previous qualification                          0
Previous qualification (grade)                  0
Nacionality                                     0
Mother's qualification                          0
Father's qualification                          0
Mother's occupation                             0
Father's occupation                             0
Admission grade                                 0
Displaced                                       0
Educational special needs                       0
Debtor                                          0
Tuition fees up to date                         0
Gender                                          0
Scholarship holder                              0
Age at enrollment                               0
International                                   0
Curricular units 1st sem (credited)             0
Curricular units 1st sem (enrolled)             0
Curricular units 1st sem (evaluations)          0
Curricular units 1st sem (approved)             0
Curricular units 1st sem (grade)                0
Curricular units 1st sem (without evaluations)  0
Curricular units 2nd sem (credited)             0
Curricular units 2nd sem (enrolled)             0
Curricular units 2nd sem (evaluations)          0
Curricular units 2nd sem (approved)             0
Curricular units 2nd sem (grade)                0
Curricular units 2nd sem (without evaluations)  0
Unemployment rate                               0
Inflation rate                                  0
GDP                                             0
Target                                          0
dtype: int64
```

In [5]:
```
#checking dupliactes
df.duplicated().sum()
```

Out[5]:
```
0
```

In [6]:
```
# Knowing the count of unique values
df.nunique()
```

Out[6]:
```
id                                              76518
Marital status                                      6
Application mode                                   22
Application order                                   8
Course                                             19
Daytime/evening attendance                          2
Previous qualification                             21
Previous qualification (grade)                    110
Nacionality                                        18
Mother's qualification                             35
Father's qualification                             39
Mother's occupation                                40
Father's occupation                                56
Admission grade                                   668
Displaced                                           2
Educational special needs                           2
Debtor                                              2
Tuition fees up to date                             2
Gender                                              2
Scholarship holder                                  2
Age at enrollment                                  46
International                                       2
Curricular units 1st sem (credited)                21
Curricular units 1st sem (enrolled)                24
Curricular units 1st sem (evaluations)             36
Curricular units 1st sem (approved)                23
Curricular units 1st sem (grade)                 1206
Curricular units 1st sem (without evaluations)     12
Curricular units 2nd sem (credited)                20
Curricular units 2nd sem (enrolled)                22
Curricular units 2nd sem (evaluations)             31
Curricular units 2nd sem (approved)                21
Curricular units 2nd sem (grade)                 1234
Curricular units 2nd sem (without evaluations)     11
Unemployment rate                                  11
Inflation rate                                     13
GDP                                                11
Target                                              3
dtype: int64
```

In [7]:
```
# basic info of datatypes and columns
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 76518 entries, 0 to 76517
Data columns (total 38 columns):
 #   Column                                          Non-Null Count  Dtype  
---  ------                                          --------------  -----  
 0   id                                              76518 non-null  int64  
 1   Marital status                                  76518 non-null  int64  
 2   Application mode                                76518 non-null  int64  
 3   Application order                               76518 non-null  int64  
 4   Course                                          76518 non-null  int64  
 5   Daytime/evening attendance                      76518 non-null  int64  
 6   Previous qualification                          76518 non-null  int64  
 7   Previous qualification (grade)                  76518 non-null  float64
 8   Nacionality                                     76518 non-null  int64  
 9   Mother's qualification                          76518 non-null  int64  
 10  Father's qualification                          76518 non-null  int64  
 11  Mother's occupation                             76518 non-null  int64  
 12  Father's occupation                             76518 non-null  int64  
 13  Admission grade                                 76518 non-null  float64
 14  Displaced                                       76518 non-null  int64  
 15  Educational special needs                       76518 non-null  int64  
 16  Debtor                                          76518 non-null  int64  
 17  Tuition fees up to date                         76518 non-null  int64  
 18  Gender                                          76518 non-null  int64  
 19  Scholarship holder                              76518 non-null  int64  
 20  Age at enrollment                               76518 non-null  int64  
 21  International                                   76518 non-null  int64  
 22  Curricular units 1st sem (credited)             76518 non-null  int64  
 23  Curricular units 1st sem (enrolled)             76518 non-null  int64  
 24  Curricular units 1st sem (evaluations)          76518 non-null  int64  
 25  Curricular units 1st sem (approved)             76518 non-null  int64  
 26  Curricular units 1st sem (grade)                76518 non-null  float64
 27  Curricular units 1st sem (without evaluations)  76518 non-null  int64  
 28  Curricular units 2nd sem (credited)             76518 non-null  int64  
 29  Curricular units 2nd sem (enrolled)             76518 non-null  int64  
 30  Curricular units 2nd sem (evaluations)          76518 non-null  int64  
 31  Curricular units 2nd sem (approved)             76518 non-null  int64  
 32  Curricular units 2nd sem (grade)                76518 non-null  float64
 33  Curricular units 2nd sem (without evaluations)  76518 non-null  int64  
 34  Unemployment rate                               76518 non-null  float64
 35  Inflation rate                                  76518 non-null  float64
 36  GDP                                             76518 non-null  float64
 37  Target                                          76518 non-null  object 
dtypes: float64(7), int64(30), object(1)
memory usage: 22.2+ MB
```

In [8]:
```
# Understanding the datasets by statistics
df.describe().T
```

Out[8]:

	count	mean	std	min	25%	50%	75%
id	76518.0	38258.500000	22088.988286	0.00	19129.250000	38258.500000	57387.75000
Marital status	76518.0	1.111934	0.441669	1.00	1.000000	1.000000	1.00000
Application mode	76518.0	16.054419	16.682337	1.00	1.000000	17.000000	39.00000
Application order	76518.0	1.644410	1.229645	0.00	1.000000	1.000000	2.00000
Course	76518.0	9001.286377	1803.438531	33.00	9119.000000	9254.000000	9670.00000
Daytime/evening attendance	76518.0	0.915314	0.278416	0.00	1.000000	1.000000	1.00000
Previous qualification	76518.0	3.658760	8.623774	1.00	1.000000	1.000000	1.00000
Previous qualification (grade)	76518.0	132.378766	10.995328	95.00	125.000000	133.100000	140.00000
Nacionality	76518.0	1.226600	3.392183	1.00	1.000000	1.000000	1.00000
Mother's qualification	76518.0	19.837633	15.399456	1.00	1.000000	19.000000	37.00000
Father's qualification	76518.0	23.425076	14.921164	1.00	4.000000	19.000000	37.00000
Mother's occupation	76518.0	8.583196	17.471591	0.00	4.000000	7.000000	9.00000
Father's occupation	76518.0	8.882172	16.803940	0.00	5.000000	7.000000	9.00000
Admission grade	76518.0	125.363971	12.562328	95.00	118.000000	124.600000	132.00000
Displaced	76518.0	0.569265	0.495182	0.00	0.000000	1.000000	1.00000
Educational special needs	76518.0	0.003738	0.061023	0.00	0.000000	0.000000	0.00000
Debtor	76518.0	0.071382	0.257463	0.00	0.000000	0.000000	0.00000
Tuition fees up to date	76518.0	0.893646	0.308292	0.00	1.000000	1.000000	1.00000
Gender	76518.0	0.315821	0.464845	0.00	0.000000	0.000000	1.00000
Scholarship holder	76518.0	0.247393	0.431500	0.00	0.000000	0.000000	0.00000
Age at enrollment	76518.0	22.278653	6.889241	17.00	18.000000	19.000000	23.00000
International	76518.0	0.006626	0.081130	0.00	0.000000	0.000000	0.00000
Curricular units 1st sem (credited)	76518.0	0.188871	1.175296	0.00	0.000000	0.000000	0.00000
Curricular units 1st sem	76518.0	5.891516	1.671776	0.00	5.000000	6.000000	6.00000

	count	mean	std	min	25%	50%	75%
(enrolled)							
Curricular units 1st sem (evaluations)	76518.0	7.352362	3.508292	0.00	6.000000	7.000000	9.00000
Curricular units 1st sem (approved)	76518.0	4.178520	2.687995	0.00	2.000000	5.000000	6.00000
Curricular units 1st sem (grade)	76518.0	9.995862	5.264224	0.00	10.666667	12.166667	13.31428
Curricular units 1st sem (without evaluations)	76518.0	0.057960	0.408490	0.00	0.000000	0.000000	0.00000
Curricular units 2nd sem (credited)	76518.0	0.137053	0.933830	0.00	0.000000	0.000000	0.00000
Curricular units 2nd sem (enrolled)	76518.0	5.933414	1.627182	0.00	5.000000	6.000000	6.00000
Curricular units 2nd sem (evaluations)	76518.0	7.234468	3.503040	0.00	6.000000	7.000000	9.00000
Curricular units 2nd sem (approved)	76518.0	4.007201	2.772956	0.00	1.000000	5.000000	6.00000
Curricular units 2nd sem (grade)	76518.0	9.626085	5.546035	0.00	10.000000	12.142857	13.24404
Curricular units 2nd sem (without evaluations)	76518.0	0.062443	0.462107	0.00	0.000000	0.000000	0.00000
Unemployment rate	76518.0	11.520340	2.653375	7.60	9.400000	11.100000	12.70000
Inflation rate	76518.0	1.228218	1.398816	-0.80	0.300000	1.400000	2.60000
GDP	76518.0	-0.080921	2.251382	-4.06	-1.700000	0.320000	1.79000

Label encoding the Target variable

In [9]:
```
from sklearn.preprocessing import LabelEncoder

le = LabelEncoder()

df['Target'] = le.fit_transform(df['Target'])
```

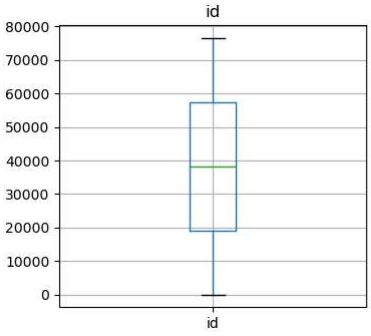
In [10]:
```
# COrelation by target variable
correlation_table = df.corr()['Target'][:-1]
print(correlation_table)
```

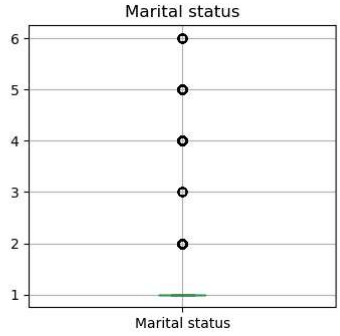
```
id                                               0.001411
Marital status                                  -0.123093
Application mode                                -0.324830
Application order                                0.128394
Course                                           0.154208
Daytime/evening attendance                       0.124484
Previous qualification                          -0.092319
Previous qualification (grade)                   0.138119
Nacionality                                     -0.004722
Mother's qualification                          -0.085056
Father's qualification                          -0.012449
Mother's occupation                             -0.043157
Father's occupation                             -0.032646
Admission grade                                  0.172880
Displaced                                        0.150066
Educational special needs                       -0.000499
Debtor                                          -0.248391
Tuition fees up to date                          0.415691
Gender                                          -0.330181
Scholarship holder                               0.394124
Age at enrollment                               -0.320679
International                                   -0.000326
Curricular units 1st sem (credited)              0.035764
Curricular units 1st sem (enrolled)              0.263657
Curricular units 1st sem (evaluations)           0.152398
Curricular units 1st sem (approved)              0.725490
Curricular units 1st sem (grade)                 0.661355
Curricular units 1st sem (without evaluations)  -0.060941
Curricular units 2nd sem (credited)              0.038062
Curricular units 2nd sem (enrolled)              0.289165
Curricular units 2nd sem (evaluations)           0.214951
Curricular units 2nd sem (approved)              0.781452
Curricular units 2nd sem (grade)                 0.719036
Curricular units 2nd sem (without evaluations)  -0.072690
Unemployment rate                                0.015731
Inflation rate                                  -0.037344
GDP                                              0.106462
Name: Target, dtype: float64
```

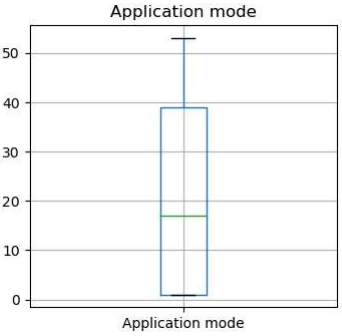
- This show the realationship between other variables to Target variable
- For example Gender is inversely proportional to the target variable has the value in (-) represents negative relationship.
- 0 represents no relation . -1 to +1 show the strength of relationship.

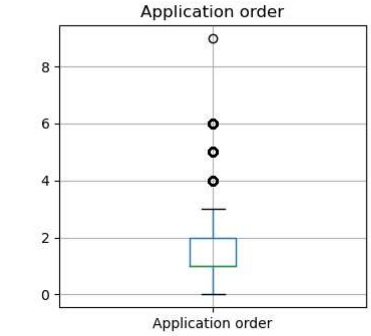
In [11]:
```
# Determine the number of columns and create a subplot grid
n_cols = len(df.columns)
n_rows = (n_cols // 3) + (n_cols % 3 > 0)

fig, axes = plt.subplots(nrows=n_rows, ncols=3, figsize=(15, 5 * n_rows))
fig.subplots_adjust(hspace=0.4, wspace=0.4)

# Flatten axes array for easy iteration
axes = axes.flatten()

# Plot each column's box plot
for idx, column in enumerate(df.columns):
    df.boxplot(column, ax=axes[idx])
```

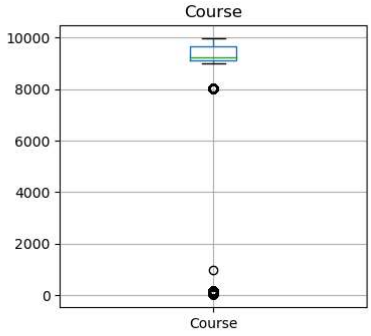
```
    axes[idx].set_title(column)

# Remove any unused subplots
for j in range(idx + 1, len(axes)):
    fig.delaxes(axes[j])

plt.show()
```

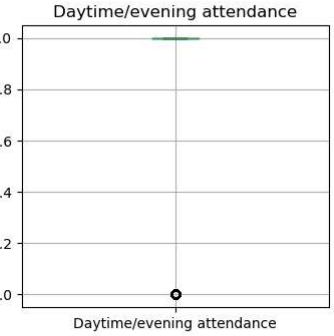

id
id

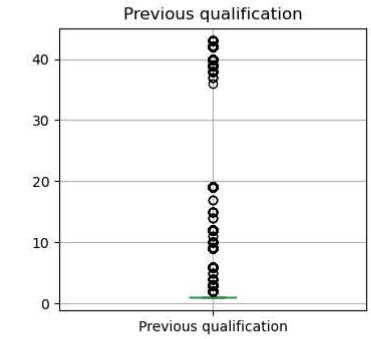

Marital status
Marital status

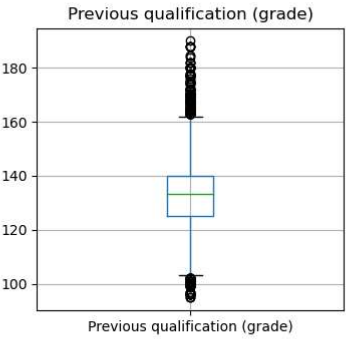

Application mode
Application mode

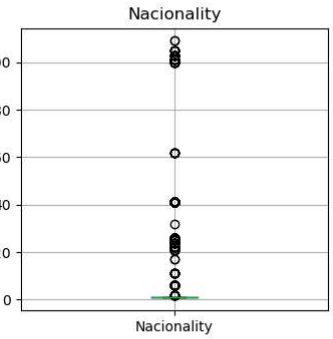

Application order
Application order

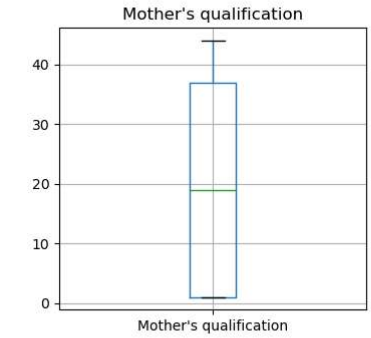

Course
Course


Daytime/evening attendance
Daytime/evening attendance

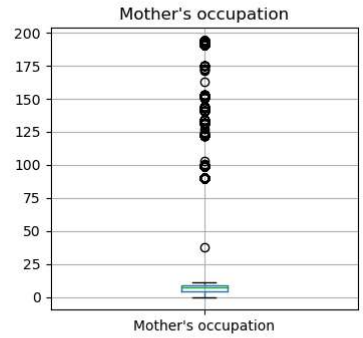

Previous qualification
Previous qualification

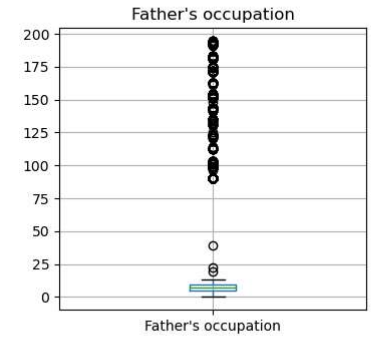

Previous qualification (grade)
Previous qualification (grade)

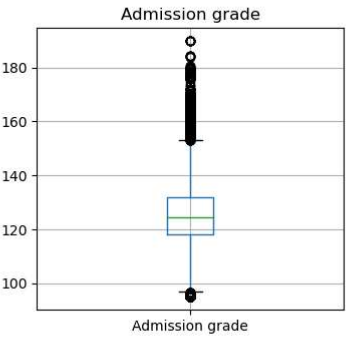

Nacionality
Nacionality

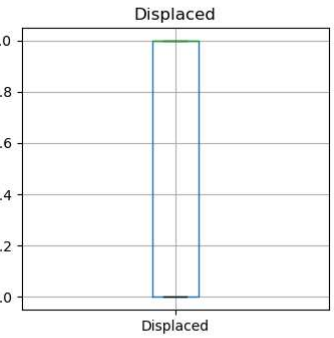

Mother's qualification
Mother's qualification

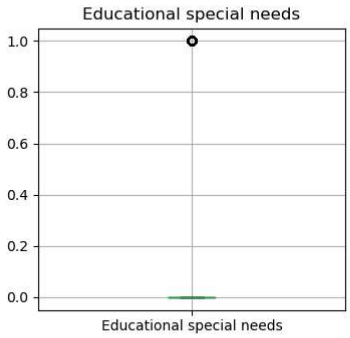

Father's qualification
Father's qualification

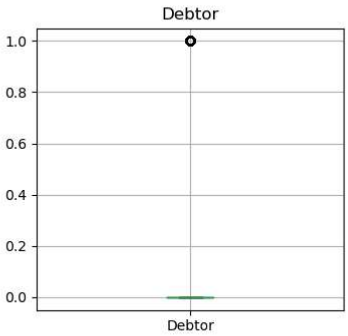

Mother's occupation
Mother's occupation

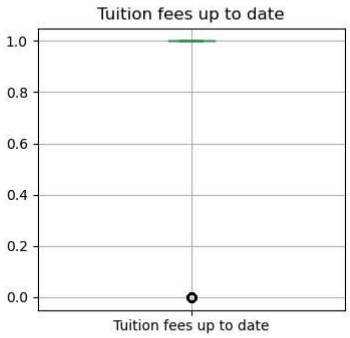

Father's occupation
Father's occupation

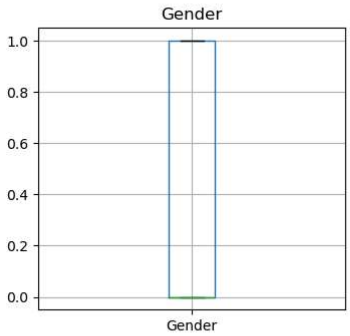

Admission grade
Admission grade

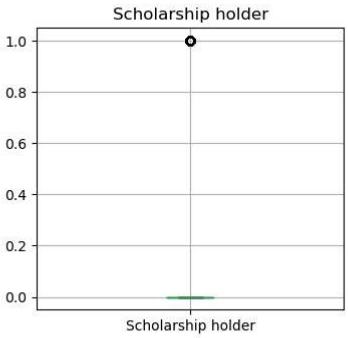

Displaced
Displaced

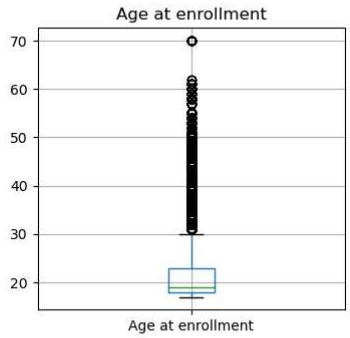

Educational special needs
Educational special needs

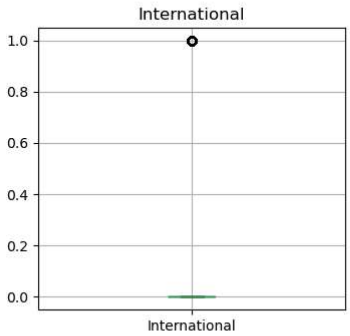

Debtor
Debtor

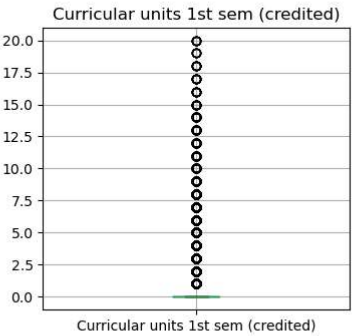

Tuition fees up to date
Tuition fees up to date

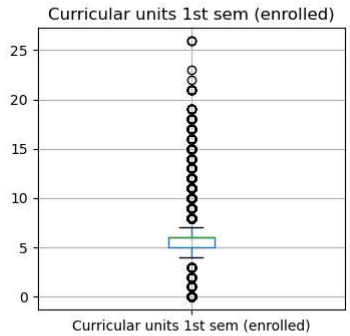

Gender
Gender

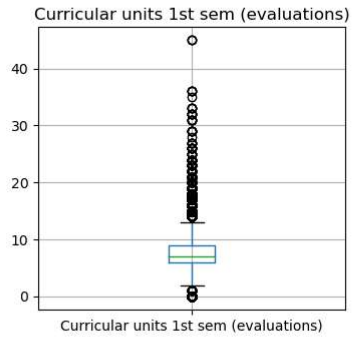

Scholarship holder
Scholarship holder

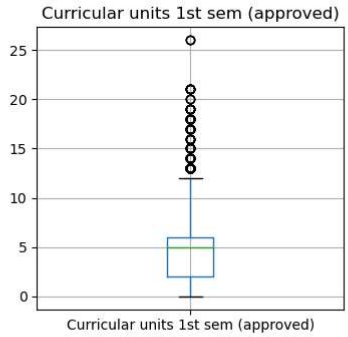

Age at enrollment
Age at enrollment

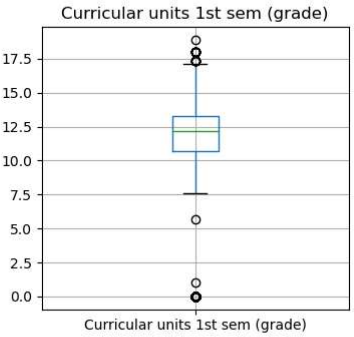

International
International

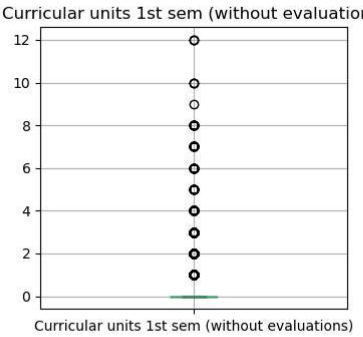

Curricular units 1st sem (credited)
Curricular units 1st sem (credited)

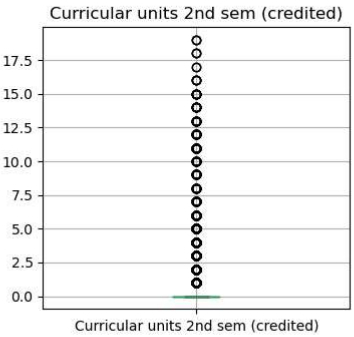

Curricular units 1st sem (enrolled)
Curricular units 1st sem (enrolled)

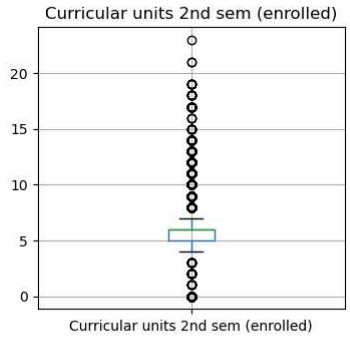

Curricular units 1st sem (evaluations)
Curricular units 1st sem (evaluations)

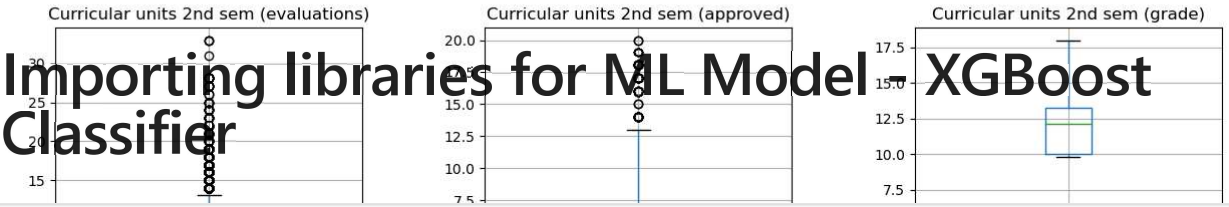

Curricular units 1st sem (approved)
Curricular units 1st sem (approved)


Curricular units 1st sem (grade)
Curricular units 1st sem (grade)


Curricular units 1st sem (without evaluations)
Curricular units 1st sem (without evaluations)


Curricular units 2nd sem (credited)
Curricular units 2nd sem (credited)


Curricular units 2nd sem (enrolled)
Curricular units 2nd sem (enrolled)



Importing libraries for ML Model - XGBoost Classifier

In [12]:
```
from sklearn.model_selection import train_test_split
import xgboost as xgb
from sklearn.metrics import accuracy_score,classification_report
from sklearn.metrics import confusion_matrix
from sklearn.metrics import ConfusionMatrixDisplay
from sklearn.metrics import confusion_matrix, roc_auc_score, roc_curve, RocCurveDispla
from sklearn.preprocessing import label_binarize
```

In [13]:
```
# making the tain and test split
X = df.drop('Target',axis=1)
y = df[['Target']]

X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.30,random_state=1)

print("X_train shape",X_train.shape)
print("X_test shape",X_test.shape)
print("y_train shape",y_train.shape)
print("y_test shape",y_test.shape)
```

```
X_train shape (53562, 37)
X_test shape (22956, 37)
y_train shape (53562, 1)
y_test shape (22956, 1)
```

In [14]:
```
# Making XG boost model and tuning to get best result
model = xgb.XGBClassifier(max_depth = 9,
                          subsample = 0.8,
                          n_estimators = 410,
                          learning_rate = 0.03,
                          min_child_weight = 3,
                          reg_alpha = 0,
                          re_lambda = 0
)


model.fit(X_train,y_train)
y_predict = model.predict(X_test)
y_pred_train = model.predict(X_train)
```

In [15]:
```
# Checking the accuracy of the model
print('Train Accuracy: ',accuracy_score(y_train,y_pred_train))

print('Test Accuracy: ',accuracy_score(y_test,y_predict))
```

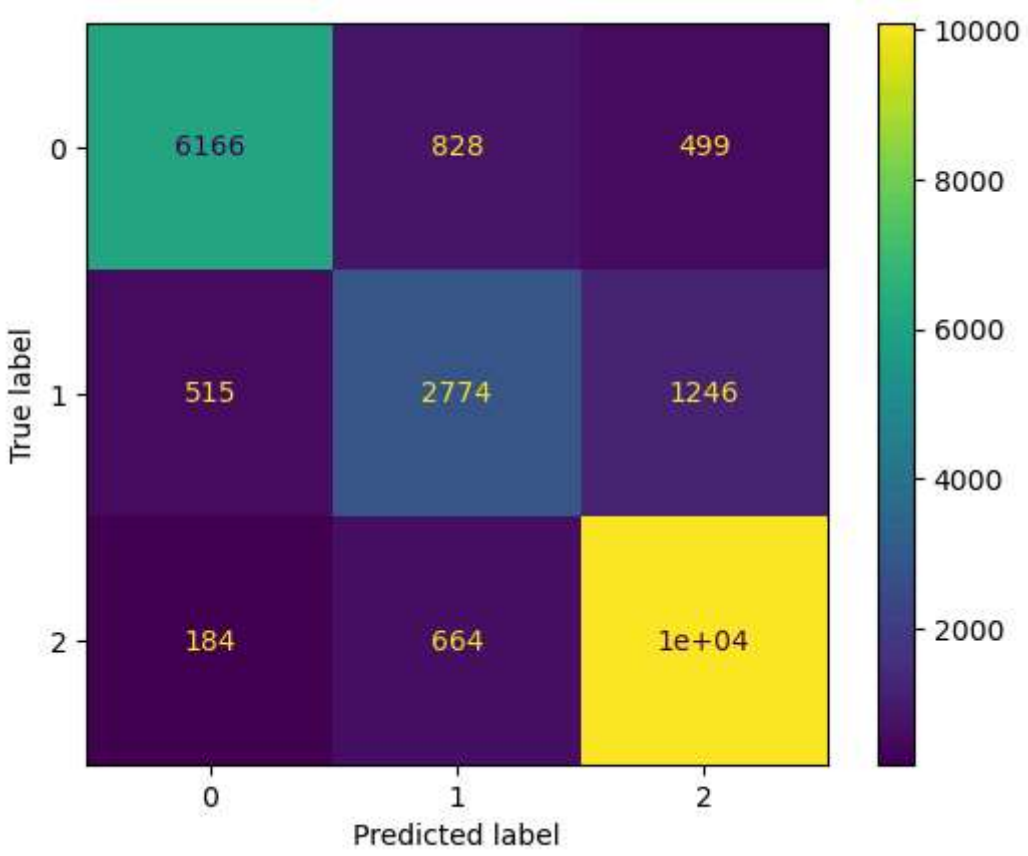
```
Train Accuracy:  0.9224076770845002
Test Accuracy:  0.8285415577626765
```

In [16]:
```
# CHecking the percentage of True positives and Accuracy
print(classification_report(y_test,y_predict))
```

	precision	recall	f1-score	support
0	0.90	0.82	0.86	7493
1	0.65	0.61	0.63	4535
2	0.85	0.92	0.89	10928
accuracy			0.83	22956
macro avg	0.80	0.79	0.79	22956
weighted avg	0.83	0.83	0.83	22956

In [17]:
```
# Visualizing the confusion matrix
cm = confusion_matrix(y_test,y_predict)

plt.figure(figsize=(15,8))
disp = ConfusionMatrixDisplay(confusion_matrix=cm)
disp.plot()
plt.show()
```

```
<Figure size 1500x800 with 0 Axes>
```

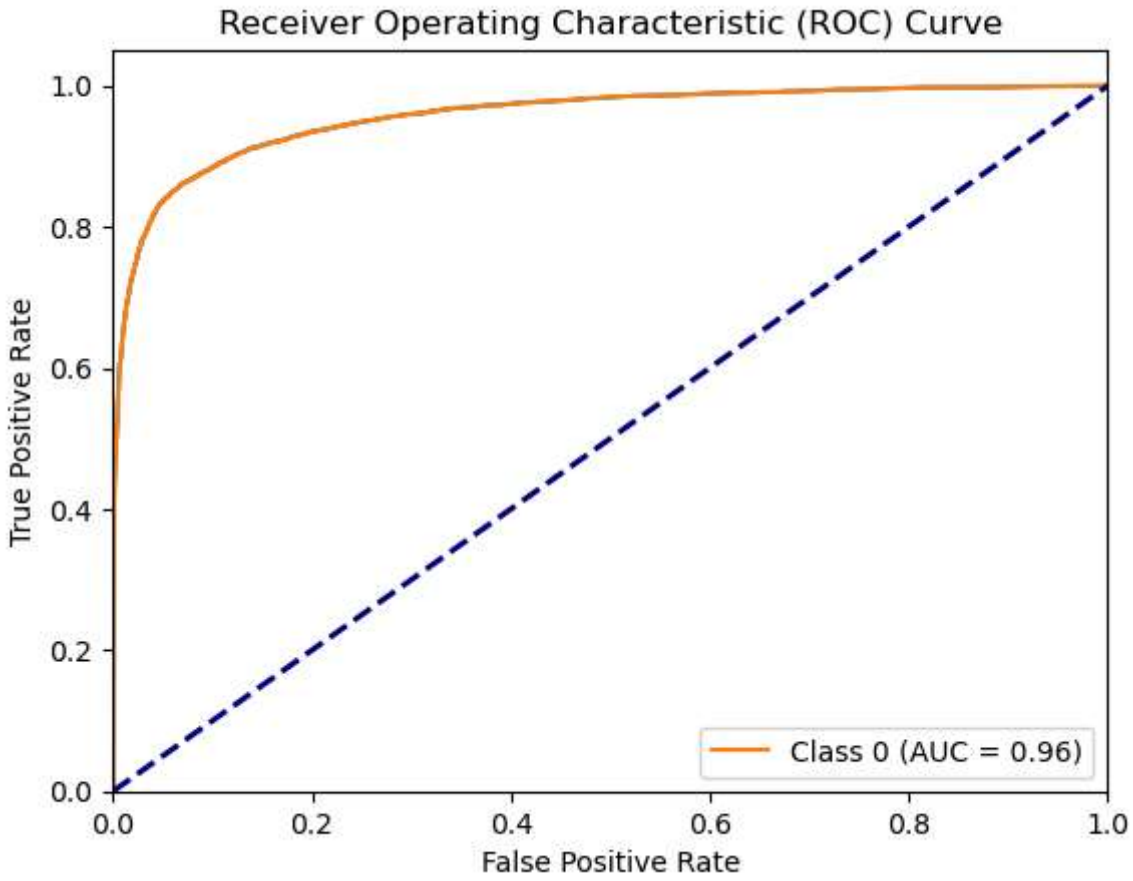


AUC-ROC Curve

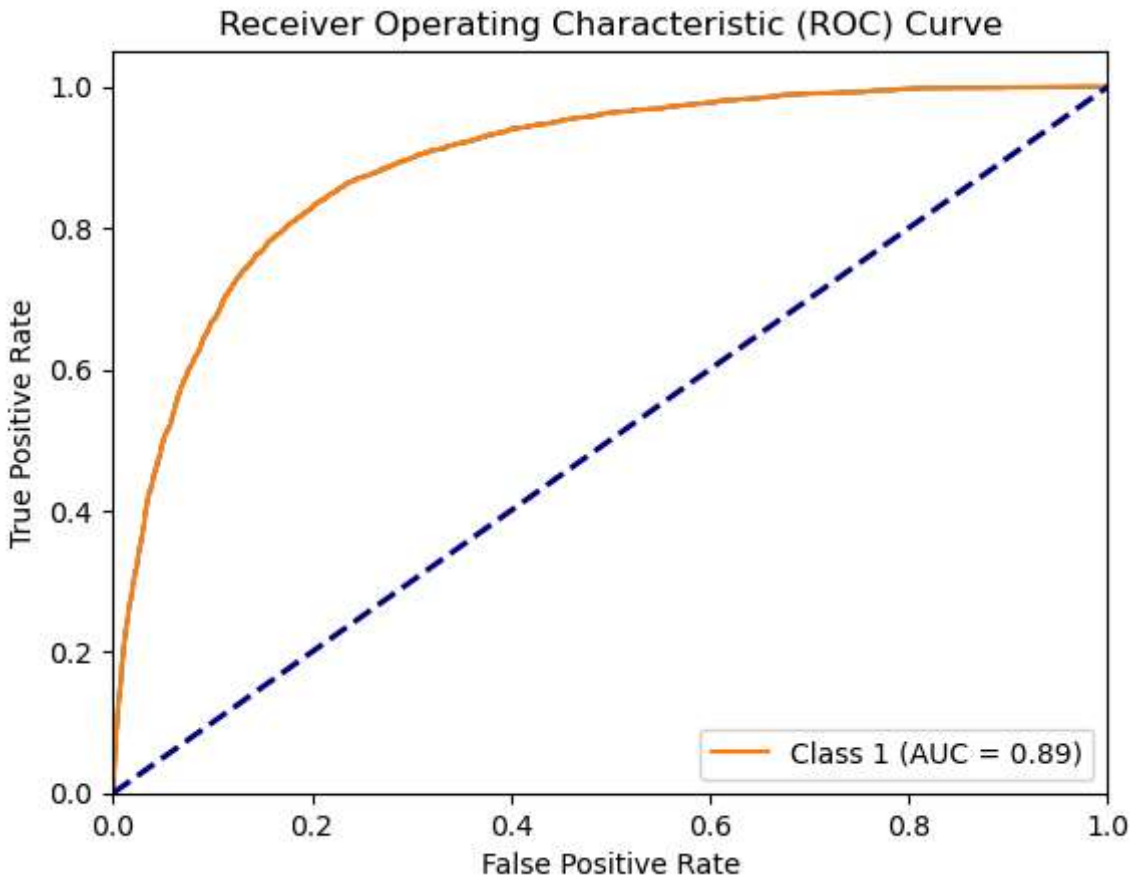
In [18]:
```
# Use the model's predict_proba method to get the predicted probabilities
y_predict_proba = model.predict_proba(X_test)

# Binarize the output
n_classes = y_predict_proba.shape[1]
y_test_binarized = label_binarize(y_test, classes=[*range(n_classes)])

# Calculate AUC-ROC score using the predicted probabilities
```

```
auc_roc = roc_auc_score(y_test, y_predict_proba, multi_class='ovr')
print(f'AUC-ROC Score: {auc_roc}')

# Plot ROC curve for each class
plt.figure(figsize=(12, 8))
for i in range(n_classes):
    fpr, tpr, _ = roc_curve(y_test_binarized[:, i], y_predict_proba[:, i])
    RocCurveDisplay(fpr=fpr, tpr=tpr).plot()
    plt.plot(fpr, tpr, label=f'Class {i} (AUC = {roc_auc_score(y_test_binarized[:, i],
    plt.plot([0, 1], [0, 1],color='navy',lw=2,linestyle='--')
    plt.xlim([0.0, 1.0])
    plt.ylim([0.0, 1.05])
    plt.xlabel('False Positive Rate')
    plt.ylabel('True Positive Rate')
    plt.title('Receiver Operating Characteristic (ROC) Curve')
    plt.legend(loc='lower right')
    plt.show()
```

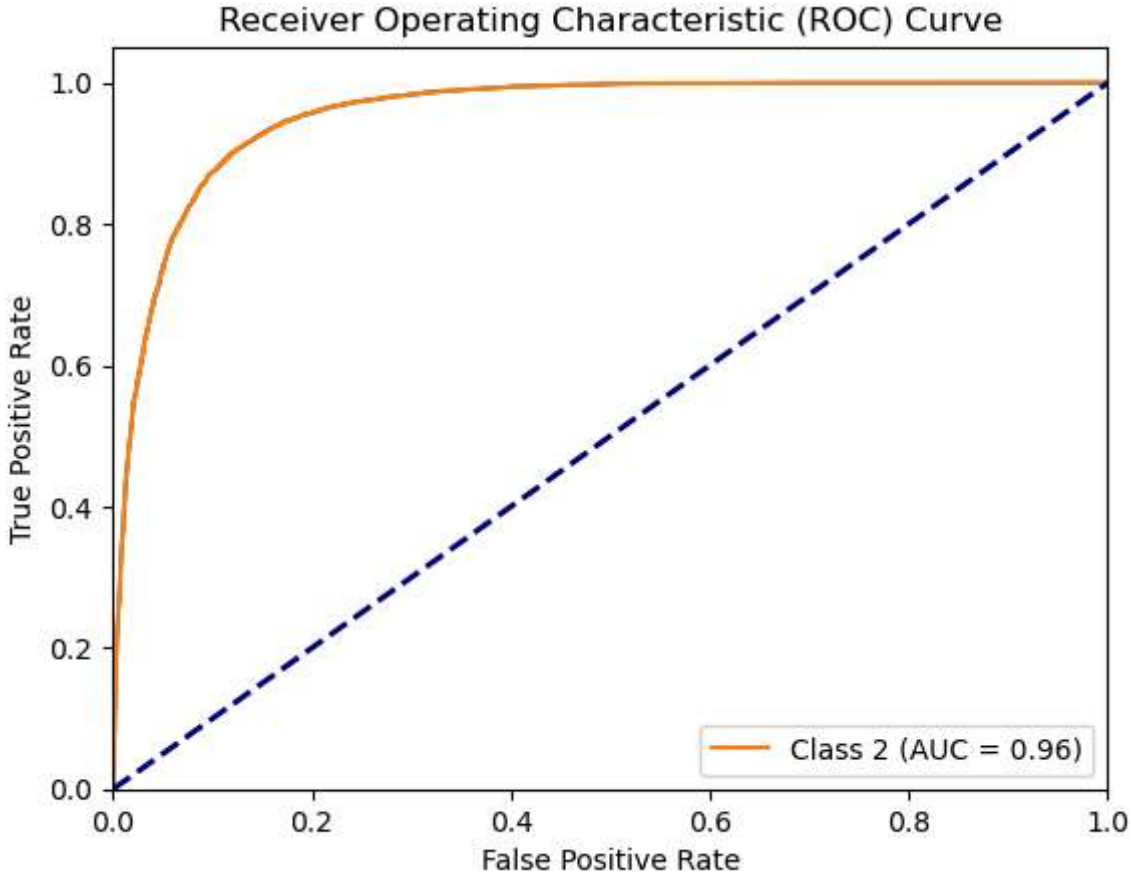
```
No artists with labels found to put in legend.  Note that artists whose label start w
ith an underscore are ignored when legend() is called with no argument.
AUC-ROC Score: 0.9341586963791775
<Figure size 1200x800 with 0 Axes>
```




```
No artists with labels found to put in legend.  Note that artists whose label start w
ith an underscore are ignored when legend() is called with no argument.
```




```
No artists with labels found to put in legend.  Note that artists whose label start w
ith an underscore are ignored when legend() is called with no argument.
```



The Accuracy and AUC-ROC curve shows the significance and Percentage of True Predictions

Now we are going to implement this model to test dataset to predict the Target variable

In [19]:
```
test = pd.read_csv('test.csv')
test.head(2)
```

Out[19]:

	id	Marital status	Application mode	Application order	Course	Daytime/evening attendance	Previous qualification	Previous qualification (grade)	Na
0	76518	1	1	1	9500	1	1	141.0	
1	76519	1	1	1	9238	1	1	128.0	

2 rows × 37 columns

Assigning the prediction to target variable

In [20]:
```
test['Target'] = model.predict(test)
```

In [21]:
```
test.head(5)
```

Out[21]:

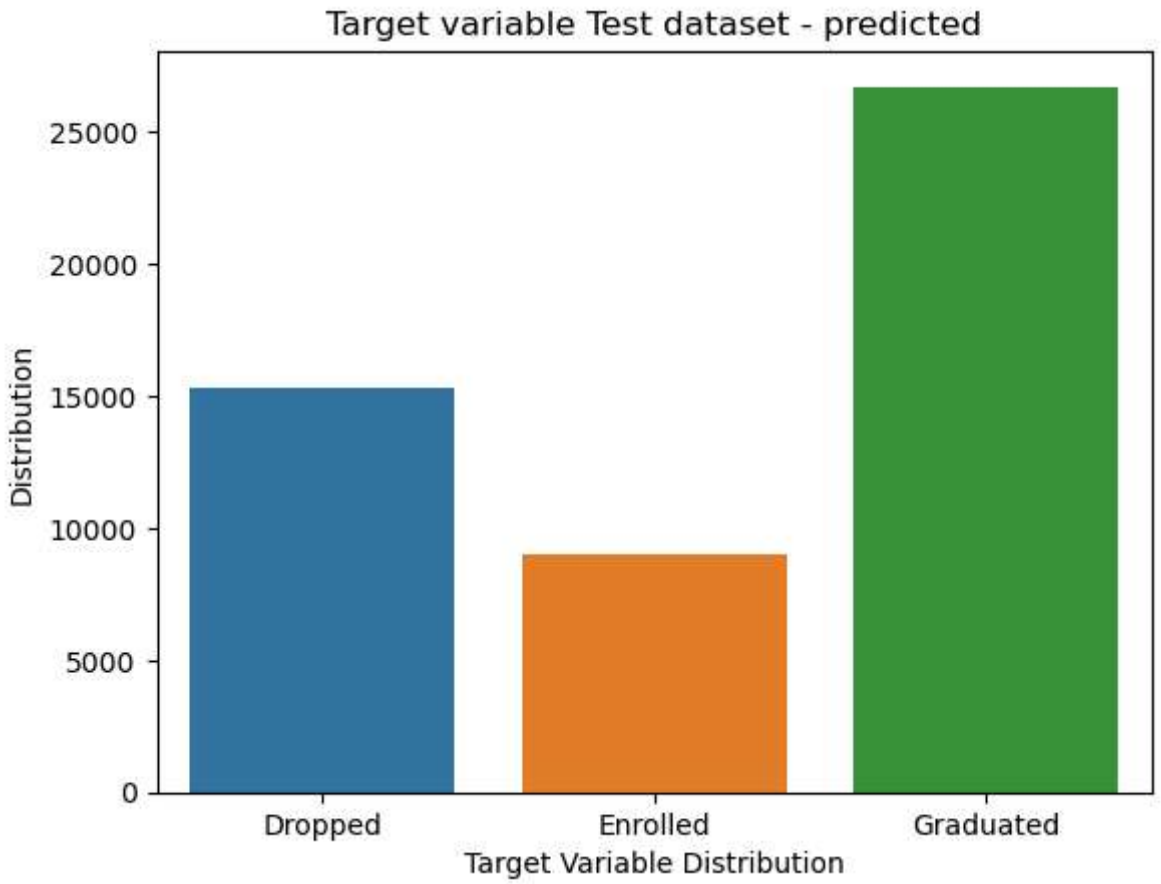
	id	Marital status	Application mode	Application order	Course	Daytime/evening attendance	Previous qualification	Previous qualification (grade)	Na
0	76518	1	1	1	9500	1	1	141.0	
1	76519	1	1	1	9238	1	1	128.0	
2	76520	1	1	1	9238	1	1	118.0	
3	76521	1	44	1	9147	1	39	130.0	
4	76522	1	39	1	9670	1	1	110.0	

5 rows × 38 columns

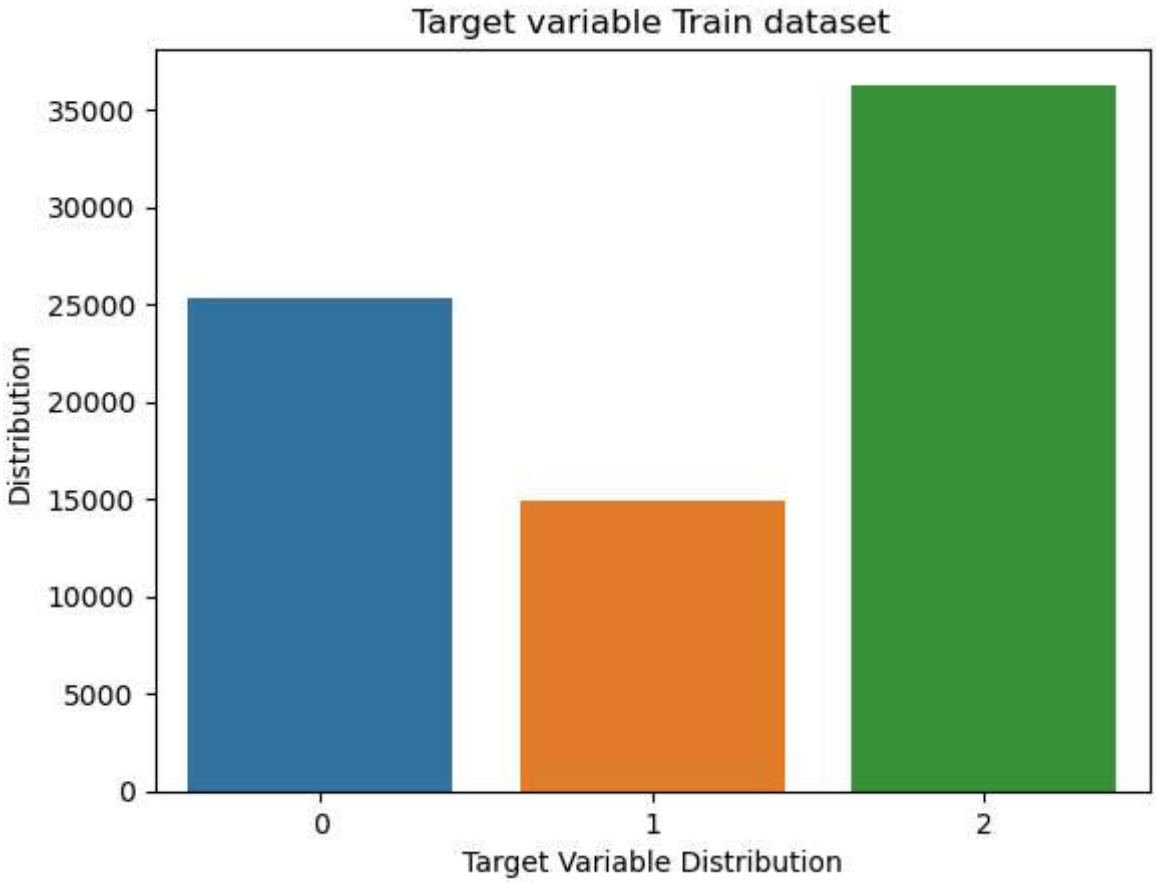
In [22]:
```
# creating the categorical cat variable as in datasets
val = {0:'Dropped',1:'Enrolled',2:'Graduated'}
test['Target_cat'] = test['Target'].replace(val)
```

Visualizing the Target variables of Test and Train Datasets

In [23]:
```
counts = test['Target_cat'].value_counts()
sns.barplot(x=counts.index, y=counts,order = counts.index.sort_values())
plt.xlabel('Target Variable Distribution')
plt.ylabel('Distribution')
plt.title("Target variable Test dataset - predicted")
plt.show()
```



In [24]:
```
counts = df['Target'].value_counts()
sns.barplot(x=counts.index, y=counts,order = counts.index.sort_values())
plt.xlabel('Target Variable Distribution')
plt.ylabel('Distribution')
plt.title("Target variable Train dataset")
plt.show()
```


Target variable Train dataset
Distribution
35000
30000
25000
20000
15000
10000
5000
0
0
1
2
Target Variable Distribution

In [ ]:

In [ ]: