IMPORT LIBRARIES

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

```
from google.colab import drive
drive.mount('/content/drive')
```

```
    Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

IMPORT DATASET

```
df = pd.read_csv('/content/drive/MyDrive/data science/miniproject/collegePlace (1).csv')
df
```

	Age	Gender	Stream	Internships	CGPA	Hostel	HistoryOfBacklogs	PlacedOrNot
0	22	Male	Electronics And Communication	1	8	1	1	1
1	21	Female	Computer Science	0	7	1	1	1
2	22	Female	Information Technology	1	6	0	0	1
3	21	Male	Information Technology	0	8	0	1	1
4	22	Male	Mechanical	0	8	1	0	1
...	...	...	...	...	...	...	...	...
2961	23	Male	Information Technology	0	7	0	0	0
2962	23	Male	Mechanical	1	7	1	0	0
2963	22	Male	Information Technology	1	7	0	0	0
2964	22	Male	Computer Science	1	7	0	0	0
2965	23	Male	Civil	0	8	0	0	1

2966 rows × 8 columns

TOTAL NUMBER OF FEATURES AND ATTRIBUTES IN DATASET

```
df.shape
```

```
    (2966, 8)
```

```
df.size
```

```
    23728
```

```
df.head()
```

	Age	Gender	Stream	Internships	CGPA	Hostel	HistoryOfBacklogs	PlacedOrNot
0	22	Male	Electronics And Communication	1	8	1	1	1
1	21	Female	Computer Science	0	7	1	1	1
2	22	Female	Information Technology	1	6	0	0	1
3	21	Male	Information Technology	0	8	0	1	1
4	22	Male	Mechanical	0	8	1	0	1

```
df.tail()
```

	Age	Gender	Stream	Internships	CGPA	Hostel	HistoryOfBacklogs	PlacedOrNot
2961	23	Male	Information Technology	0	7	0	0	0
2962	23	Male	Mechanical	1	7	1	0	0
2963	22	Male	Information Technology	1	7	0	0	0
2964	22	Male	Computer Science	1	7	0	0	0
2965	23	Male	Civil	0	8	0	0	1

HOW DATA LOOKS

```
df.describe()
```

	Age	Internships	CGPA	Hostel	HistoryOfBacklogs	PlacedOrNot
count	2966.000000	2966.000000	2966.000000	2966.000000	2966.000000	2966.000000
mean	21.485840	0.703641	7.073837	0.269049	0.192178	0.552596
std	1.324933	0.740197	0.967748	0.443540	0.394079	0.497310
min	19.000000	0.000000	5.000000	0.000000	0.000000	0.000000
25%	21.000000	0.000000	6.000000	0.000000	0.000000	0.000000
50%	21.000000	1.000000	7.000000	0.000000	0.000000	1.000000
75%	22.000000	1.000000	8.000000	1.000000	0.000000	1.000000
max	30.000000	3.000000	9.000000	1.000000	1.000000	1.000000

SHOWS THE TYPE OF DATA

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2966 entries, 0 to 2965
Data columns (total 8 columns):
 #   Column             Non-Null Count  Dtype 
---  ------             --------------  ----- 
 0   Age                2966 non-null   int64 
 1   Gender             2966 non-null   object
 2   Stream             2966 non-null   object
 3   Internships        2966 non-null   int64 
 4   CGPA               2966 non-null   int64 
 5   Hostel             2966 non-null   int64 
 6   HistoryOfBacklogs  2966 non-null   int64 
 7   PlacedOrNot        2966 non-null   int64 
dtypes: int64(6), object(2)
memory usage: 185.5+ KB
```

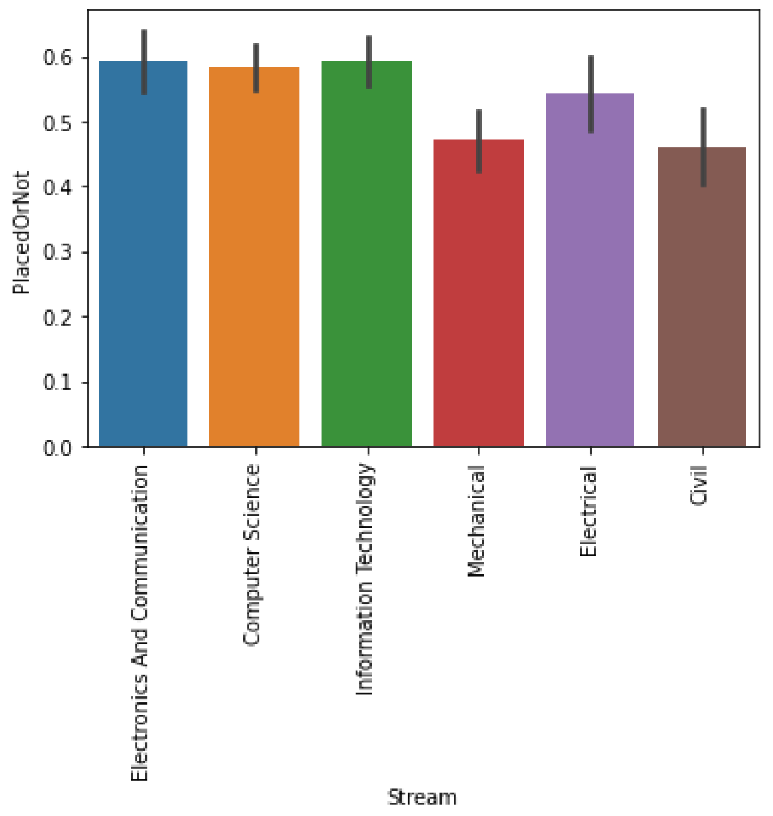
```
df.corr()
```

	Age	Internships	CGPA	Hostel	HistoryOfBacklogs	PlacedOrNot
Age	1.000000	0.006552	-0.119787	0.003042	-0.042586	0.046943
Internships	0.006552	1.000000	0.023496	0.004617	-0.015118	0.179334
CGPA	-0.119787	0.023496	1.000000	0.014991	0.002576	0.588648
Hostel	0.003042	0.004617	0.014991	1.000000	0.103506	-0.038182
HistoryOfBacklogs	-0.042586	-0.015118	0.002576	0.103506	1.000000	-0.022337
PlacedOrNot	0.046943	0.179334	0.588648	-0.038182	-0.022337	1.000000

GRAPHS

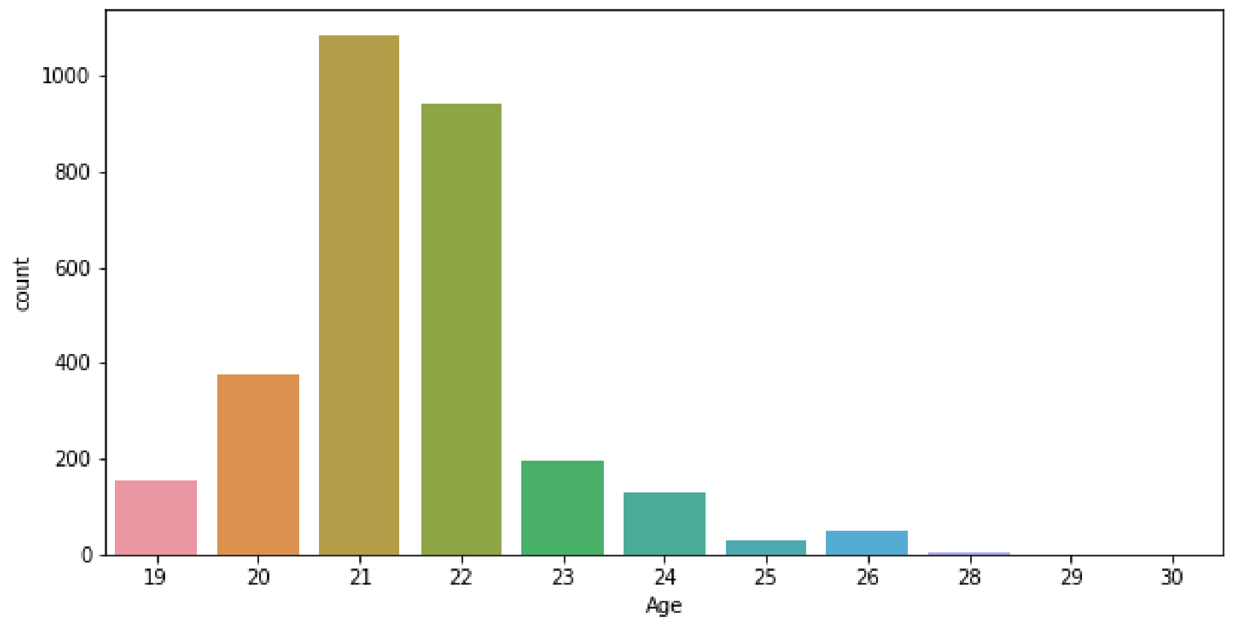
```
plt.xticks(rotation = 90)
sns.barplot(x = df.Stream, y = df.PlacedOrNot)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f123338fdd0>
```



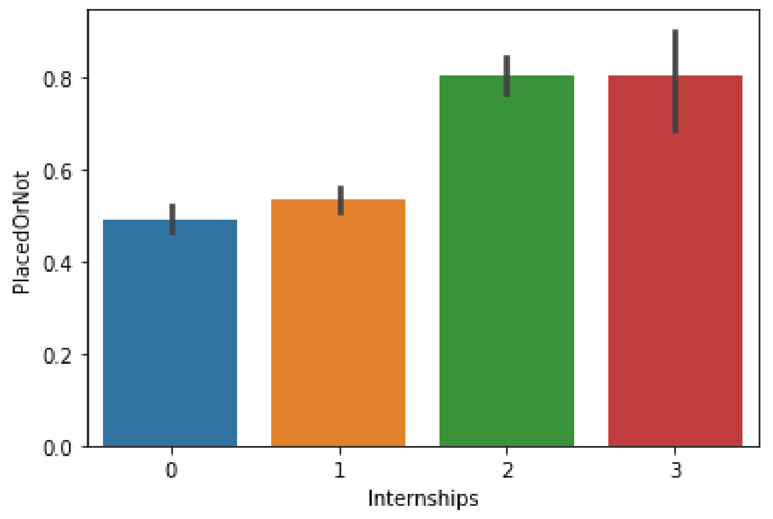
```
plt.figure(figsize = (10,5))
sns.countplot(x = df.Age)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f123322a2d0>
```



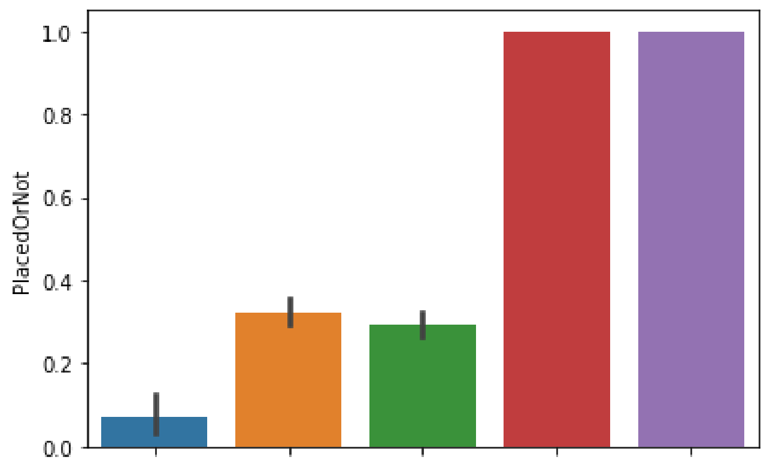
```
sns.barplot(x = df.Internships, y = df.PlacedOrNot)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f1232d69990>
```



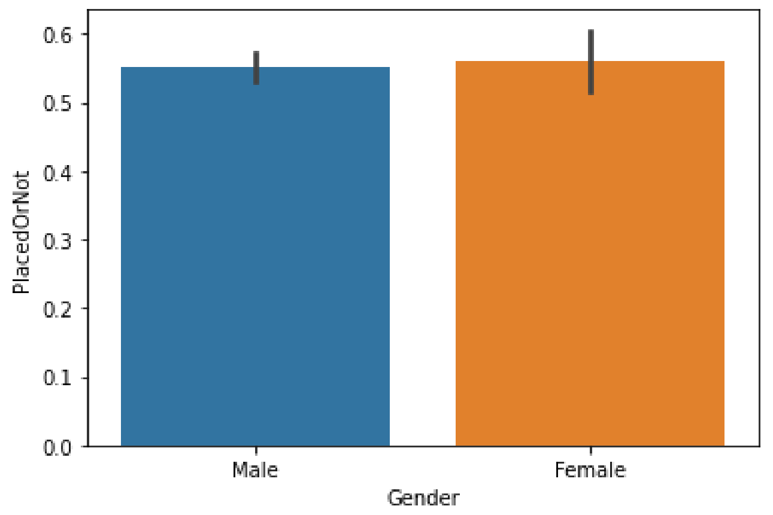
```
sns.barplot(x = df.CGPA, y = df.PlacedOrNot)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f1232d01450>
```



```
sns.barplot(x = df.Gender, y = df.PlacedOrNot)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f1232c794d0>
```



```
df.Stream.unique()
```

```
array(['Electronics And Communication', 'Computer Science',
       'Information Technology', 'Mechanical', 'Electrical', 'Civil'],
      dtype=object)
```

PRE-PROCESSING

```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
```

```
df.Gender = le.fit_transform(df.Gender)
df.Stream = le.fit_transform(df.Stream)
```

```
df.head()
```

	Age	Gender	Stream	Internships	CGPA	Hostel	HistoryOfBacklogs	PlacedOrNot
0	22	1	3	1	8	1	1	1
1	21	0	1	0	7	1	1	1
2	22	0	4	1	6	0	0	1
3	21	1	4	0	8	0	1	1
4	22	1	5	0	8	1	0	1

```
df.tail()
```

	Age	Gender	Stream	Internships	CGPA	Hostel	HistoryOfBacklogs	PlacedOrNot
2961	23	1	4	0	7	0	0	0
2962	23	1	5	1	7	1	0	0
2963	22	1	4	1	7	0	0	0
2964	22	1	1	1	7	0	0	0
2965	23	1	0	0	8	0	0	1

```
from sklearn.svm import SVC
from sklearn.tree import DecisionTreeClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
```

```
from sklearn.model_selection import cross_val_score
```

Define variable

```
x = df.drop(['PlacedOrNot'], axis = 1)
```

```
y = df.PlacedOrNot
```

```
cross_val_score(SVC(), x, y, cv = 3)
```

```
array([0.73609707, 0.76238625, 0.84817814])
```

```
cross_val_score(DecisionTreeClassifier(), x, y, cv = 3)
```

```
array([0.84428716, 0.84327604, 0.90789474])
```

```
cross_val_score(LogisticRegression(), x, y, cv = 3)
```

```
array([0.71991911, 0.74823054, 0.83704453])
```

```
cross_val_score(RandomForestClassifier(n_estimators=50), x, y, cv = 3)
```

```
array([0.84327604, 0.85338726, 0.90789474])
```

```
cross_val_score(KNeighborsClassifier(),x, y ,cv = 3)

    array([0.83013145, 0.81496461, 0.87246964])
```

TRAING AND TESTING

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size = 0.2)
```

```
X_train
```

	Age	Gender	Stream	Internships	CGPA	Hostel	HistoryOfBacklogs
1405	22	1	0	0	8	1	1
2479	19	1	0	0	6	1	0
1210	21	1	4	0	9	0	0
1971	20	1	4	1	8	1	0
593	22	1	1	1	6	0	0
...	...	...	...	...	...	...	...
1463	22	1	4	0	7	0	0
2272	22	1	5	0	8	0	0
688	22	1	5	0	8	1	0
324	21	1	5	1	6	1	0
2695	20	1	4	2	8	0	0

2372 rows × 7 columns

```
X_test
```

	Age	Gender	Stream	Internships	CGPA	Hostel	HistoryOfBacklogs
1247	22	0	1	0	8	1	0
2806	23	1	1	0	6	0	0
1889	22	0	5	1	6	1	0
233	22	1	4	1	7	0	0
1808	20	0	5	1	6	1	0
...	...	...	...	...	...	...	...
1203	21	1	3	0	8	0	0
2531	20	1	1	0	8	0	0
624	22	0	3	2	8	0	0
1827	22	1	3	1	8	0	0
115	23	1	2	0	6	0	0

594 rows × 7 columns

```
y_train

    1405    1
    2479    0
    1210    1
    1971    1
    593     0
           ..
    1463    0
    2272    1
    688     1
    324     0
    2695    1
    Name: PlacedOrNot, Length: 2372, dtype: int64
```

```
y_test

    1247    1
    2806    0
    1889    0
    233     0
    1808    0
           ..
    1203    1
    2531    1
    624     1
    1827    1
    115     0
    Name: PlacedOrNot, Length: 594, dtype: int64
```

MODEL BUILDING using RANDOMFOREST Algorithm

```
model = RandomForestClassifier()
model.fit(X_train, y_train)

    RandomForestClassifier()
```

```
y_pred = model.predict(X_test)
y_pred

    array([1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1,
           1, 0, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1, 1, 0, 1, 1, 1, 0, 0, 0, 0, 0,
           1, 1, 0, 1, 1, 1, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0,
           1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0,
           0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 1, 0,
           1, 1, 0, 1, 1, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0,
           0, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1,
           0, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1,
           0, 1, 0, 1, 0, 0, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 0, 1, 1, 0, 0, 0,
           1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1, 0, 0,
           0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 1, 1,
           1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0,
           1, 0, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1,
           1, 0, 1, 0, 0, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 0, 1, 0,
           0, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0,
           0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1,
           1, 0, 0, 0, 1, 0, 0, 0, 1, 1, 1, 0, 1, 1, 1, 0, 0, 1, 0, 1, 0, 1,
           0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0,
           0, 1, 1, 0, 0, 1, 0, 1, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0,
           0, 0, 1, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 1, 1, 1,
           0, 1, 1, 1, 1, 0, 0, 1, 1, 1, 0, 0, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1,
           1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0, 0, 0,
           1, 0, 1, 1, 0, 1, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 0, 1, 0, 0, 1,
           1, 0, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0,
           0, 0, 1, 0, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0,
           1, 1, 1, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 1, 0, 1, 1, 1, 0, 0, 1, 0,
           1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0])
```

```
from sklearn.metrics import confusion_matrix,accuracy_score
cm = confusion_matrix(y_test, y_pred)
print(cm)
accuracy_score(y_test, y_pred)
```

```
[[260  23]
 [ 37 274]]
0.898989898989899
```

MODEL BUILDING using DECISIONTREE Algorithm

```
model1 = DecisionTreeClassifier()
model1.fit(X_train, y_train)
```

```
DecisionTreeClassifier()
```

```
y_pred = model1.predict(X_test)
y_pred
```

```
array([1, 0, 0, 0, 0, 0, 1, 0, 1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1,
       1, 0, 1, 1, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 0, 0, 0, 0,
       1, 1, 0, 1, 1, 1, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0,
       1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0,
       0, 0, 1, 1, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 1, 0,
       1, 1, 0, 0, 1, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0,
       0, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0, 1, 1, 1, 1, 1, 0, 1, 0, 0,
       0, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1,
       0, 1, 0, 1, 0, 0, 1, 1, 0, 0, 0, 1, 1, 1, 0, 1, 0, 1, 1, 0, 0, 0,
       1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 0, 1, 0, 0, 1, 1, 1, 0, 0,
       0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 0, 1, 1,
       0, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0,
       1, 0, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1,
       0, 0, 1, 0, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 1, 0, 1, 1, 0, 1, 0,
       0, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0,
       0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1,
       1, 0, 0, 0, 1, 0, 0, 0, 1, 1, 1, 0, 1, 1, 1, 0, 0, 1, 0, 1, 0, 1,
       0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0,
       0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 1, 1,
       0, 1, 1, 1, 1, 0, 0, 1, 1, 1, 0, 0, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1,
       1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 0, 1, 0, 0, 0,
       1, 0, 1, 1, 0, 1, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 0, 1, 0, 0, 1,
       1, 0, 1, 0, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0,
       0, 0, 1, 0, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0,
       1, 1, 1, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 1, 0, 1, 1, 1, 0, 0, 1, 0,
       1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0])
```

```
from sklearn.metrics import confusion_matrix,accuracy_score
cm = confusion_matrix(y_test, y_pred)
print(cm)
accuracy_score(y_test, y_pred)
```

```
[[272  11]
 [ 38 273]]
0.9175084175084175
```

For this problem **DecisionTree** is the best model.

✓ 0s completed at 10:48 AM