```
import pandas as pd
import numpy as np
import matplotlib as plt
import seaborn as sns
```

```
df=pd.read_csv('/content/Admission_Predict.csv')
```

```
df
```

	Serial No.	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Research	Chance of Admit
0	1	337	118	4	4.5	4.5	9.65	1	0.92
1	2	324	107	4	4.0	4.5	8.87	1	0.76
2	3	316	104	3	3.0	3.5	8.00	1	0.72
3	4	322	110	3	3.5	2.5	8.67	1	0.80
4	5	314	103	2	2.0	3.0	8.21	0	0.65
...	...	...	...	...	...	...	...	...	...
395	396	324	110	3	3.5	3.5	9.04	1	0.82
396	397	325	107	3	3.0	3.5	9.11	1	0.84
397	398	330	116	4	5.0	4.5	9.45	1	0.91
398	399	312	103	3	3.5	4.0	8.78	0	0.67
399	400	333	117	4	5.0	4.0	9.66	1	0.95

400 rows × 9 columns

```
df.columns
```

```
Index(['Serial No.', 'GRE Score', 'TOEFL Score', 'University Rating', 'SOP',
       'LOR ', 'CGPA', 'Research', 'Chance of Admit '],
      dtype='object')
```

```
df.shape
```

```
(400, 9)
```

```
df.head()
```

	Serial No.	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Research	Chance of Admit
0	1	337	118	4	4.5	4.5	9.65	1	0.92
1	2	324	107	4	4.0	4.5	8.87	1	0.76
2	3	316	104	3	3.0	3.5	8.00	1	0.72
3	4	322	110	3	3.5	2.5	8.67	1	0.80
4	5	314	103	2	2.0	3.0	8.21	0	0.65

```
from sklearn.preprocessing import Binarizer
bi = Binarizer( threshold=0.75 )  #0.75> then 1 else 0
df['Chance of Admit '] = bi.fit_transform(df[['Chance of Admit ']])
```

```
df.head()
```

	Serial No.	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Research	Chance of Admit
0	1	337	118	4	4.5	4.5	9.65	1	1.0
1	2	324	107	4	4.0	4.5	8.87	1	1.0
2	3	316	104	3	3.0	3.5	8.00	1	0.0
3	4	322	110	3	3.5	2.5	8.67	1	1.0
4	5	314	103	2	2.0	3.0	8.21	0	0.0

```
x = df.drop('Chance of Admit ',axis = 1)
y = df['Chance of Admit ']
```

```
x
```

	Serial No.	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Research
0	1	337	118	4	4.5	4.5	9.65	1
1	2	324	107	4	4.0	4.5	8.87	1
2	3	316	104	3	3.0	3.5	8.00	1
3	4	322	110	3	3.5	2.5	8.67	1
4	5	314	103	2	2.0	3.0	8.21	0
...	...	...	...	...	...	...	...	...
395	396	324	110	3	3.5	3.5	9.04	1
396	397	325	107	3	3.0	3.5	9.11	1
397	398	330	116	4	5.0	4.5	9.45	1
398	399	312	103	3	3.5	4.0	8.78	0
399	400	333	117	4	5.0	4.0	9.66	1

400 rows × 8 columns

```
y
```

```
0      1.0
1      1.0
2      0.0
3      1.0
4      0.0
      ...
395    1.0
396    1.0
397    1.0
398    0.0
399    1.0
Name: Chance of Admit , Length: 400, dtype: float64
```

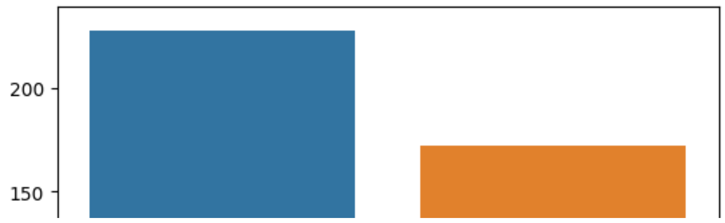
```
y = y.astype('int')
```

```
y
```

```
0      1
1      1
2      0
3      1
4      0
      ..
395    1
396    1
397    1
398    0
399    1
Name: Chance of Admit , Length: 400, dtype: int64
```

```
sns.countplot(x=y)
```

```
<Axes: xlabel='Chance of Admit ', ylabel='count'>
```



```
y.value_counts()
```

```
0    228
1    172
Name: Chance of Admit , dtype: int64
```

```
#Q.2
#cross validation
from sklearn.model_selection import train_test_split
x_train , x_test , y_train , y_test = train_test_split(x,y,random_state=0,test_size=0.25)
```

```
x_train.shape
```

```
(300, 8)
```

```
x_test.shape
```

```
(100, 8)
```

```
#import the class
from sklearn.tree import DecisionTreeClassifier
```

```
classifer = DecisionTreeClassifier(random_state=0)
```

```
classifer.fit(x_train,y_train)
```

```
▾        DecisionTreeClassifier
DecisionTreeClassifier(random_state=0)
```

```
y_pred = classifer.predict(x_test)
```

```
result = pd.DataFrame({
 'actual': y_test,
 'predicted':y_pred
})
```

```
result
```

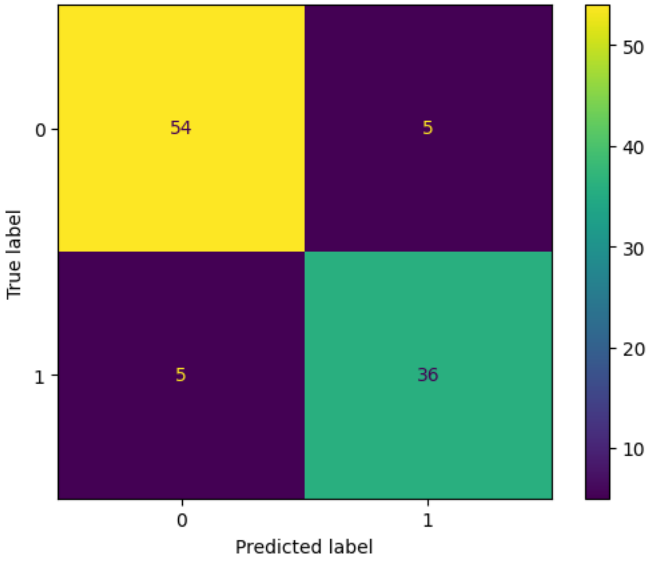
	actual	predicted
132	0	0
309	0	0
341	1	1
196	0	0
246	0	1
...	...	...
146	0	0
135	1	1
390	0	0
264	0	0
364	1	1

100 rows × 2 columns

```
from sklearn.metrics import ConfusionMatrixDisplay, accuracy_score, classification_report
```

```
ConfusionMatrixDisplay.from_predictions(y_test,y_pred)
```

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x797962a4ded0>
```



```
accuracy_score(y_test,y_pred)
```

```
0.9
```

```
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.92      0.92      0.92        59
           1       0.88      0.88      0.88        41

    accuracy                           0.90       100
   macro avg       0.90      0.90      0.90       100
weighted avg       0.90      0.90      0.90       100
```