

```
In [1]: import pandas as pd
import seaborn as sns

In [2]: df=pd.read_csv("D:/Admission_Predict.csv")

In [3]: df.columns

Out[3]: Index(['Serial No.', 'GRE Score', 'TOEFL Score', 'University Rating', 'SOP',
'LOR ', 'CGPA', 'Research', 'Chance of Admit '],
dtype='object')

In [4]: df.shape

Out[4]: (400, 9)

In [5]: df.head()

Out[5]:
```

	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

```


In [6]: from sklearn.preprocessing import Binarizer
bi=Binarizer(threshold=0.75)
df['Chance of Admit ']=bi.fit_transform(df[['Chance of Admit ']])

In [7]: df.head()

Out[7]:
```

	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

```


In [8]: x=df.drop('Chance of Admit ',axis=1)
y=df['Chance of Admit ']

In [9]: x

Out[9]:
```

	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

```


In [10]: y

Out[10]:
```

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

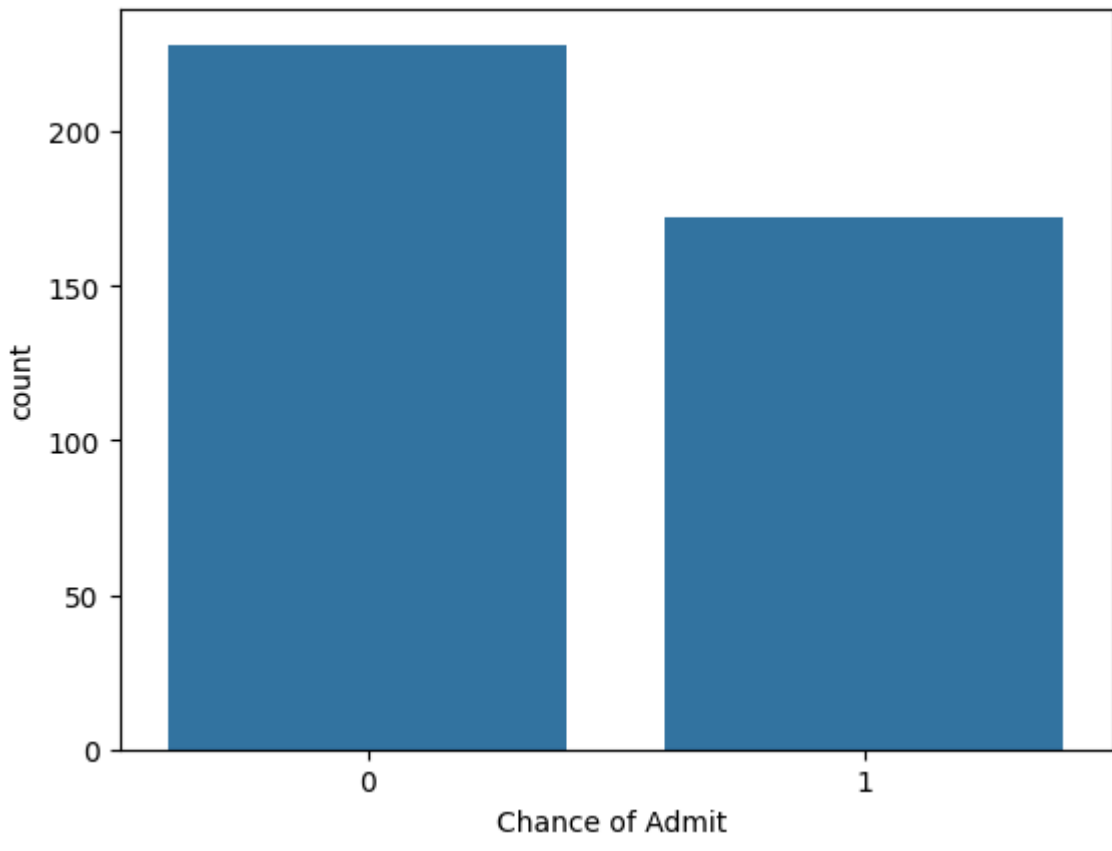
Name: Chance of Admit , Length: 400, dtype: float64

```


In [11]: y=y.astype('int')

In [12]: sns.countplot(x = y)

Out[12]: <Axes: xlabel='Chance of Admit ', ylabel='count'>
```



```


In [13]: y.value_counts()

Out[13]:
```

Chance of Admit	count
0	228
1	172

Name: count, dtype: int64

```


In [14]: 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)

In [15]: x_train.shape

Out[15]: (300, 8)

In [16]: x_test.shape

Out[16]: (100, 8)

In [17]: x_test

Out[17]:
```

	Serial No.	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Research
132	133	309	105	5	3.5	3.5	8.56	0
309	310	308	110	4	3.5	3.0	8.60	0
341	342	326	110	3	3.5	3.5	8.76	1
196	197	306	105	2	3.0	2.5	8.26	0
246	247	316	105	3	3.0	3.5	8.73	0
...
146	147	315	105	3	2.0	2.5	8.48	0
135	136	314	109	4	3.5	4.0	8.77	1
390	391	314	102	2	2.0	2.5	8.24	0
264	265	325	110	2	3.0	2.5	8.76	1
364	365	313	102	3	3.5	4.0	8.90	1

100 rows × 8 columns

```


In [18]: from sklearn.tree import DecisionTreeClassifier
classifier=DecisionTreeClassifier(random_state=0)

In [19]: classifier.fit(x_train,y_train)

Out[19]:
```

DecisionTreeClassifier

DecisionTreeClassifier(random_state=0)

```

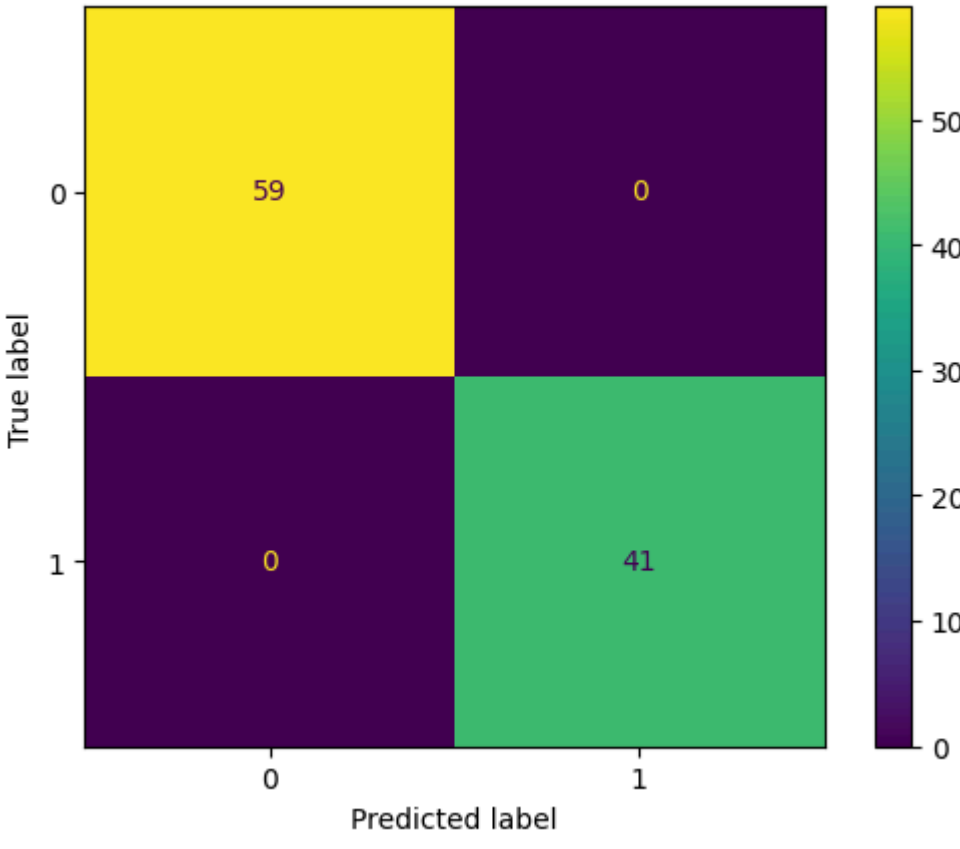

In [20]: y_pred=classifier.predict(x_test)

result=pd.DataFrame({"actual": y_test,"predicted"y_pred)) result

In [21]: from sklearn.metrics import ConfusionMatrixDisplay,accuracy_score
from sklearn.metrics import classification_report

In [22]: ConfusionMatrixDisplay.from_predictions(y_test,y_pred)

Out[22]: <sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x221a1a82020>
```



```


In [23]: accuracy_score(y_test,y_pred)

Out[23]: 0.9

In [24]: 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