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

```
df=pd.read_csv("/content/Admission_Predict.csv")
df=pd.read_csv("/content/Admission_Predict_Ver1.1.csv")
```

```
df.columns
```

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

Default title text

```
# @title Default title text
df.shape
```

```
(500, 9)
```

```
from sklearn.preprocessing import Binarizer
bi=Binarizer(threshold=0.75)
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
...	...	...	...	...	...	...	...	...
495	496	332	108	5	4.5	4.0	9.02	1
496	497	337	117	5	5.0	5.0	9.87	1
497	498	330	120	5	4.5	5.0	9.56	1
498	499	312	103	4	4.0	5.0	8.43	0
499	500	327	113	4	4.5	4.5	9.04	0

500 rows × 8 columns

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

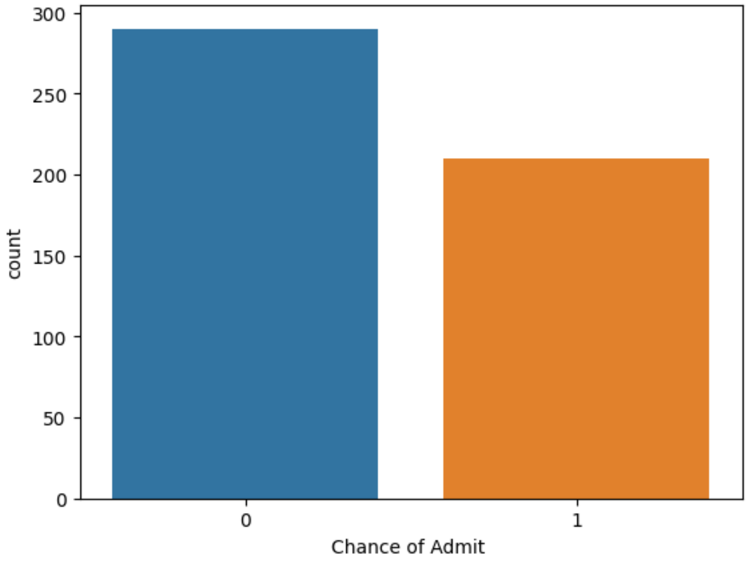
```
y
```

```
0      1
1      1
2      0
3      1
```

```
     4      0
           ..
     495    1
     496    1
     497    1
     498    0
     499    1
     Name: Chance of Admit , Length: 500, dtype: int64
```

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

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



```
y.value_counts()
```

```
     0    290
     1    210
     Name: Chance of Admit , dtype: int64
```

```
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)
```

```
xt=x_train.shape
print("The shape of traning data is:",xt)
```

```
     The shape of traning data is: (375, 8)
```

```
xxt=x_test.shape
print("The shape of testing data is:",xxt)
```

```
     The shape of traning data is: (125, 8)
```

```
yt=y_train.shape
print("The shape of traning data is:",yt)
```

```
     The shape of traning data is: (375,)
```

```
yyt=y_test.shape
print("The shape of testing data is:",yyt)
```

```
     The shape of traning data is: (125,)
```

```
from sklearn.tree import DecisionTreeClassifier
```

```
classifier=DecisionTreeClassifier(random_state=0)
```

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

```
DecisionTreeClassifier
```

```
y_pred=classifier.predict(x_test)  #prediction based on input means x
```

```
#comparing acutal and predict
result=pd.DataFrame({
    'actual':y_test,  #already known values
    'predicted':y_pred  #predictedvalues
})
```

```
result
```

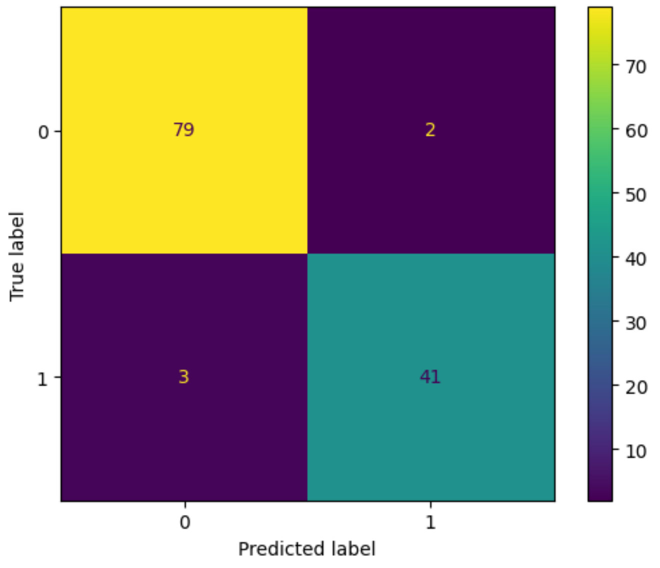
	actual	predicted
90	0	0
254	1	1
283	1	1
445	1	1
461	0	0
...	...	...
430	0	0
49	1	0
134	1	1
365	1	1
413	0	0

125 rows × 2 columns

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

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

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7990a6004130>
```



```
accuracy=accuracy_score(y_test,y_pred)
print("The accuracy is:",accuracy)
```

```
The accuracy is: 0.96
```

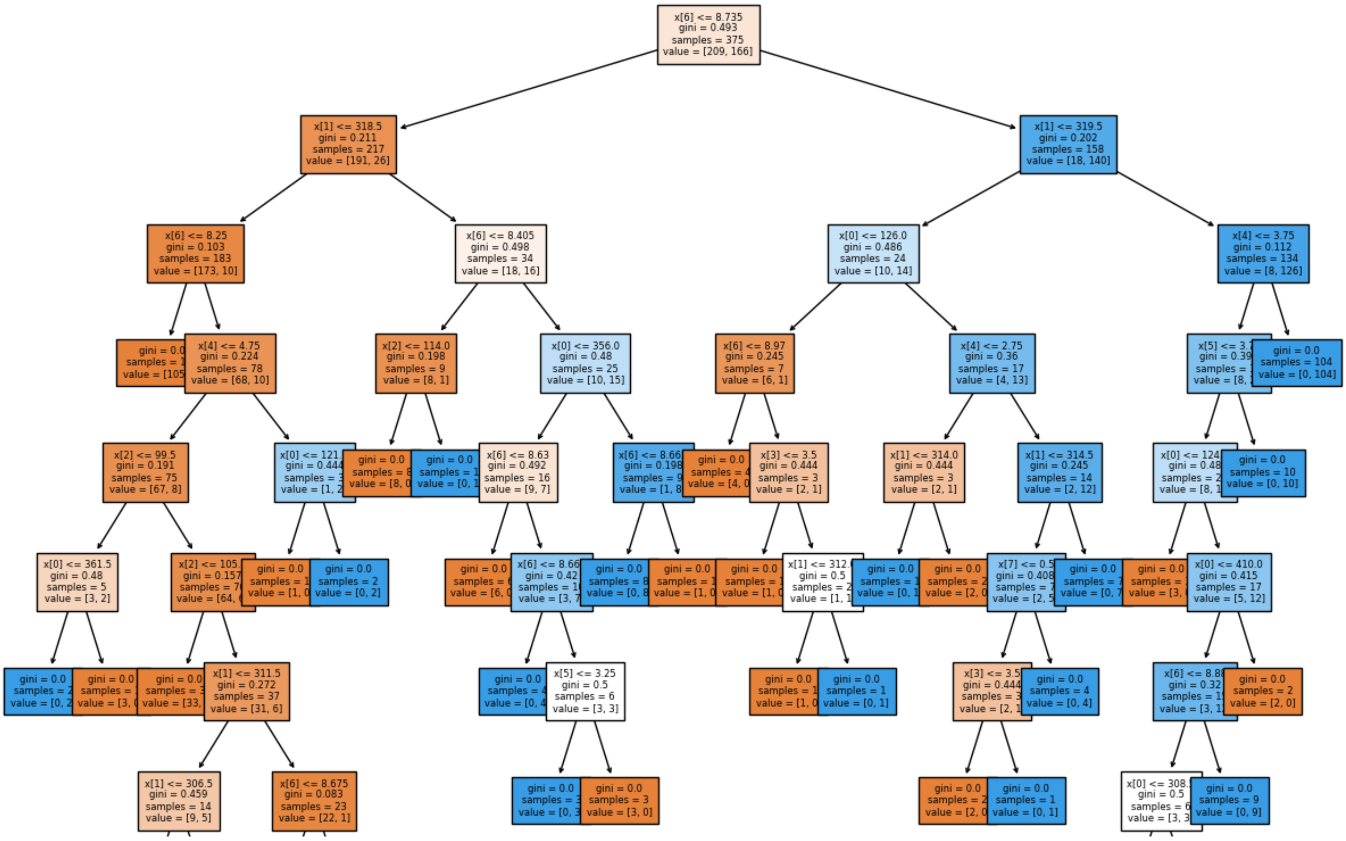
```
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.96      0.98      0.97        81
           1       0.95      0.93      0.94        44

    accuracy                           0.96       125
   macro avg       0.96      0.95      0.96       125
weighted avg       0.96      0.96      0.96       125
```

```
import matplotlib.pyplot as plt
```

```
from sklearn.tree import plot_tree
```

```
plt.figure(figsize=(15,15))
plot_tree(classifier, fontsize=6,filled=True );
```


x[6] <= 8.735 gini = 0.493 samples = 375 value = [209, 166]
x[1] <= 318.5 gini = 0.211 samples = 217 value = [191, 26]
x[1] <= 319.5 gini = 0.202 samples = 158 value = [18, 140]
x[6] <= 8.25 gini = 0.103 samples = 183 value = [173, 10]
x[6] <= 8.405 gini = 0.498 samples = 34 value = [18, 16]
x[0] <= 126.0 gini = 0.486 samples = 24 value = [10, 14]
x[4] <= 3.75 gini = 0.112 samples = 134 value = [8, 126]
gini = 0.0 samples = 1 value = [105
x[4] <= 4.75 gini = 0.224 samples = 78 value = [68, 10]
x[2] <= 114.0 gini = 0.198 samples = 9 value = [8, 1]
x[0] <= 356.0 gini = 0.48 samples = 25 value = [10, 15]
x[6] <= 8.97 gini = 0.245 samples = 7 value = [6, 1]
x[4] <= 2.75 gini = 0.36 samples = 17 value = [4, 13]
x[5] <= 3.7 gini = 0.39 samples = 1 value = [8, 2
gini = 0.0 samples = 104 value = [0, 104]
x[2] <= 99.5 gini = 0.191 samples = 75 value = [67, 8]
x[0] <= 121. gini = 0.444 samples = 3 value = [1, 2
gini = 0.0 samples = 8 value = [8, 0
gini = 0.0 samples = 1 value = [0, 1
x[6] <= 8.63 gini = 0.492 samples = 16 value = [9, 7]
x[6] <= 8.66 gini = 0.198 samples = 9 value = [1, 8
gini = 0.0 samples = 4 value = [4, 0
x[3] <= 3.5 gini = 0.444 samples = 3 value = [2, 1]
x[1] <= 314.0 gini = 0.444 samples = 3 value = [2, 1]
x[1] <= 314.5 gini = 0.245 samples = 14 value = [2, 12]
x[0] <= 124 gini = 0.48 samples = 2 value = [8, 1
gini = 0.0 samples = 10 value = [0, 10]
x[0] <= 361.5 gini = 0.48 samples = 5 value = [3, 2]
x[2] <= 105. gini = 0.157 samples = 7 value = [64,
gini = 0.0 samples = 1 value = [1, 0
gini = 0.0 samples = 2 value = [0, 2]
gini = 0.0 samples = 6 value = [6, 0
x[6] <= 8.66 gini = 0.42 samples = 1 value = [3, 7
gini = 0.0 samples = 8 value = [0, 8
gini = 0.0 samples = 1 value = [1, 0
gini = 0.0 samples = 1 value = [1, 0
x[1] <= 312. gini = 0.5 samples = 2 value = [1, 1
gini = 0.0 samples = 1 value = [0, 1
gini = 0.0 samples = 2 value = [2, 0
x[7] <= 0.5 gini = 0.408 samples = 7 value = [2, 5
gini = 0.0 samples = 7 value = [0, 7
gini = 0.0 samples = value = [3,
x[0] <= 410.0 gini = 0.415 samples = 17 value = [5, 12]
gini = 0.0 samples = 2 value = [0, 2
gini = 0.0 samples = value = [3, 0
gini = 0.0 samples = 3 value = [33,
x[1] <= 311.5 gini = 0.272 samples = 37 value = [31, 6]
gini = 0.0 samples = 4 value = [0, 4
x[5] <= 3.25 gini = 0.5 samples = 6 value = [3, 3]
gini = 0.0 samples = 1 value = [1, 0
gini = 0.0 samples = 1 value = [0, 1]
x[3] <= 3.5 gini = 0.444 samples = 3 value = [2, 1
gini = 0.0 samples = 4 value = [0, 4]
x[6] <= 8.88 gini = 0.32 samples = 15 value = [3, 12
gini = 0.0 samples = 2 value = [2, 0]
x[1] <= 306.5 gini = 0.459 samples = 14 value = [9, 5]
x[6] <= 8.675 gini = 0.083 samples = 23 value = [22, 1]
gini = 0.0 samples = 3 value = [0, 3
gini = 0.0 samples = 3 value = [3, 0]
gini = 0.0 samples = 2 value = [2, 0
gini = 0.0 samples = 1 value = [0, 1]
x[0] <= 308. gini = 0.5 samples = 6 value = [3, 3
gini = 0.0 samples = 9 value = [0, 9]