In [13]:

```
import numpy as np
import pandas as pd
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
```

In [14]:

```
df=pd.read_csv(r"C:\Users\Mastan Reddy\Downloads\loan1 (1).csv")
df
```

Out[14]:

	Home Owner	Marital Status	Annual Income	Defaulted Borrower
0	Yes	Single	125	No
1	No	Married	100	No
2	No	Single	70	No
3	Yes	Married	120	No
4	No	Divorced	95	Yes
5	No	Married	60	No
6	Yes	Divorced	220	No
7	No	Single	85	Yes
8	No	Married	75	No
9	No	Single	90	Yes

In [15]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10 entries, 0 to 9
Data columns (total 4 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   Home Owner          10 non-null     object
 1   Marital Status      10 non-null     object
 2   Annual Income       10 non-null     int64
 3   Defaulted Borrower  10 non-null     object
dtypes: int64(1), object(3)
memory usage: 448.0+ bytes
```

In [16]:

```
df['Marital Status'].value_counts()
```

Out[16]:

```
Single      4
Married     4
Divorced    2
Name: Marital Status, dtype: int64
```

In [17]:
```
convert={"Home owener":{"Yes":1,"No":0}}
df=df.replace(convert)
df
```

Out[17]:

	Home Owner	Marital Status	Annual Income	Defaulted Borrower
0	Yes	Single	125	No
1	No	Married	100	No
2	No	Single	70	No
3	Yes	Married	120	No
4	No	Divorced	95	Yes
5	No	Married	60	No
6	Yes	Divorced	220	No
7	No	Single	85	Yes
8	No	Married	75	No
9	No	Single	90	Yes

In [18]:
```
convert={"Marital Status":{"Single":1,"Married":2,"Divorced":3}}
df=df.replace(convert)
df
```

Out[18]:

	Home Owner	Marital Status	Annual Income	Defaulted Borrower
0	Yes	1	125	No
1	No	2	100	No
2	No	1	70	No
3	Yes	2	120	No
4	No	3	95	Yes
5	No	2	60	No
6	Yes	3	220	No
7	No	1	85	Yes
8	No	2	75	No
9	No	1	90	Yes

In [29]:
```
x=["Annual Income","Marital Status"]
y=["Yes","No"]
all_inputs=df[x]
all_classes=df["Defaulted Borrower"]
```

In [30]:
```
(x_train,x_test,y_train,y_test)=train_test_split(all_inputs,all_classes,test_s
In [31]: clf=DecisionTreeClassifier(random_state=0)
```

```
In [32]: clf.fit(x_train,y_train)
```

```
Out[32]: DecisionTreeClassifier(random_state=0)
```

```
In [33]: score=clf.score(x_test,y_test)
         print(score)
```

```
0.4
```

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```