Loan Data score

In [132]:

```python
import numpy as np
import pandas as pd
import seaborn as sb
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
df=pd.read_csv(r"C:\Users\magam\Downloads\loan1.csv")
df
```

Out[132]:

	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 [133]:

```python
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 [134]:

```
df.describe()
```

Out[134]:

	Annual Income
count	10.000000
mean	104.000000
std	45.631373
min	60.000000
25%	77.500000
50%	92.500000
75%	115.000000
max	220.000000

In [135]:

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

Out[135]:

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

In [136]:

```
df['Home Owner'].value_counts()
```

Out[136]:

```
Home Owner
No     7
Yes    3
Name: count, dtype: int64
```

In [137]:

```
convert={'Home Owner':{'Yes':1,'No':0},'Marital Status':{'Single':1,'Married':2,'
df=df.replace(convert)
df
```

Out[137]:

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

In [138]:

```
x=['Home Owner','Marital Status','Annual Income']
y=['Yes','No']
all_inputs=df[x]
all_classes=df["Defaulted Borrower"]
```

In [139]:

```
x_train,x_test,y_train,y_test=train_test_split(all_inputs,all_classes,train_size=
clf=DecisionTreeClassifier(random_state=0)
clf.fit(x_train,y_train)
```

Out[139]:

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

In [140]:

```
score=clf.score(x_test,y_test)
print(score)
```

```
0.5714285714285714
```