

```
In [1]: import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

```
In [2]: df=pd.read_csv("C10_loan1.csv")
df
```

```
Out[2]:
```

	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 [3]: df['Defaulted Borrower'].value_counts()
```

```
Out[3]: No      7
Yes      3
Name: Defaulted Borrower, dtype: int64
```

```
In [4]: x=df[['Annual Income','Annual Income']]
y=df['Defaulted Borrower']
```

```
In [5]: g1={"'Defaulted Borrower'":{"Yes":1,"No":2}}
df=df.replace(g1)
df
```

```
Out[5]:
```

	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 [6]: from sklearn.model_selection import train_test_split
```

```
In [7]: x_train,x_test,y_train,y_test=train_test_split(x,y,train_size=0.70)
```

```
In [8]: from sklearn.ensemble import RandomForestClassifier
```

```
In [9]: rfc=RandomForestClassifier()
rfc.fit(x_train,y_train)
```

```
Out[9]: RandomForestClassifier()
```

```
In [10]: parameters={'max_depth':[1,2,3,4,5],
                    'min_samples_leaf':[5,10,15,20,25],
                    'n_estimators':[10,20,30,40,50]}
}
```

```
In [11]: from sklearn.model_selection import GridSearchCV
grid_search =GridSearchCV(estimator=rfc,param_grid=parameters,cv=2,scoring="accuracy")
grid_search.fit(x_train,y_train)
```

```
Out[11]: GridSearchCV(cv=2, estimator=RandomForestClassifier(),
                    param_grid={'max_depth': [1, 2, 3, 4, 5],
                                'min_samples_leaf': [5, 10, 15, 20, 25],
                                'n_estimators': [10, 20, 30, 40, 50]},
                    scoring='accuracy')
```

```
In [12]: grid_search.best_score_
```

```
Out[12]: 0.7083333333333333
```

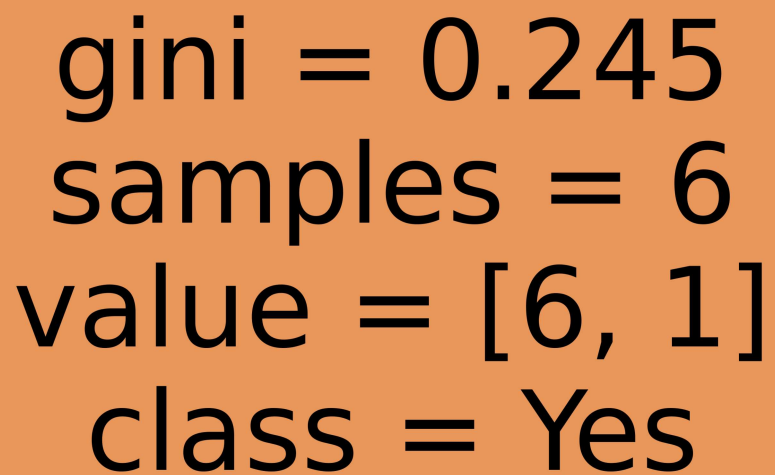
```
In [13]: rfc_best=grid_search.best_estimator_
```

```
In [14]: from sklearn.tree import plot_tree
```

```
plt.figure(figsize=(80,40))
```

```
plot_tree(rfc_best.estimators_[5],feature_names=x.columns,class_names=['Yes','No'])
```

```
Out[14]: [Text(2232.0, 1087.2, 'gini = 0.245\nsamples = 6\nvalue = [6, 1]\nclass = Yes')]
```



gini = 0.245
samples = 6
value = [6, 1]
class = Yes