

```
In [12]: import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

```
In [13]: df=pd.read_csv(r"C:\Users\user\Downloads\C8_loan-train - C8_loan-train.csv")
df
```

Out[13]:

if_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_Histor
No	5849	0.0	NaN	360.0	1
No	4583	1508.0	128.0	360.0	1
Yes	3000	0.0	66.0	360.0	1
No	2583	2358.0	120.0	360.0	1
No	6000	0.0	141.0	360.0	1
...
No	2900	0.0	71.0	360.0	1
No	4106	0.0	40.0	180.0	1
No	8072	240.0	253.0	360.0	1
No	7583	0.0	187.0	360.0	1
Yes	4583	0.0	133.0	360.0	0

```
In [14]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 614 entries, 0 to 613
Data columns (total 13 columns):
 #   Column                Non-Null Count  Dtype  
---  -
 0   Loan_ID               614 non-null   object 
 1   Gender                601 non-null   object 
 2   Married               611 non-null   object 
 3   Dependents            599 non-null   object 
 4   Education             614 non-null   object 
 5   Self_Employed         582 non-null   object 
 6   ApplicantIncome       614 non-null   int64  
 7   CoapplicantIncome     614 non-null   float64 
 8   LoanAmount            592 non-null   float64 
 9   Loan_Amount_Term      600 non-null   float64 
10  Credit_History        564 non-null   float64 
11  Property_Area         614 non-null   object 
12  Loan_Status           614 non-null   object 
dtypes: float64(4), int64(1), object(8)
memory usage: 62.5+ KB
```

```
In [16]: df['Loan_Status'].value_counts()
```

```
Out[16]: Y      422
         N      192
         Name: Loan_Status, dtype: int64
```

```
In [17]: df.columns
```

```
Out[17]: Index(['Loan_ID', 'Gender', 'Married', 'Dependents', 'Education',
                'Self_Employed', 'ApplicantIncome', 'CoapplicantIncome', 'LoanAmount',
                'Loan_Amount_Term', 'Credit_History', 'Property_Area', 'Loan_Status'],
                dtype='object')
```

```
In [21]: df1=df[['ApplicantIncome', 'CoapplicantIncome', 'LoanAmount', 'Loan_Status']]
```

```
In [29]: x=df1[['ApplicantIncome', 'CoapplicantIncome']]
         y=df1['Loan_Status']
```

```
In [30]: g1={"Loan_Status":{"N":0,'Y':1}}
df=df.replace(g1)
print(df)
```

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	\
0	LP001002	Male	No	0	Graduate	No	
1	LP001003	Male	Yes	1	Graduate	No	
2	LP001005	Male	Yes	0	Graduate	Yes	
3	LP001006	Male	Yes	0	Not Graduate	No	
4	LP001008	Male	No	0	Graduate	No	
..	
609	LP002978	Female	No	0	Graduate	No	
610	LP002979	Male	Yes	3+	Graduate	No	
611	LP002983	Male	Yes	1	Graduate	No	
612	LP002984	Male	Yes	2	Graduate	No	
613	LP002990	Female	No	0	Graduate	Yes	

	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	\
0	5849	0.0	NaN	360.0	
1	4583	1508.0	128.0	360.0	
2	3000	0.0	66.0	360.0	
3	2583	2358.0	120.0	360.0	
4	6000	0.0	141.0	360.0	
..	
609	2900	0.0	71.0	360.0	
610	4106	0.0	40.0	180.0	
611	8072	240.0	253.0	360.0	
612	7583	0.0	187.0	360.0	
613	4583	0.0	133.0	360.0	

	Credit_History	Property_Area	Loan_Status
0	1.0	Urban	1
1	1.0	Rural	0
2	1.0	Urban	1
3	1.0	Urban	1
4	1.0	Urban	1
..
609	1.0	Rural	1
610	1.0	Rural	1
611	1.0	Urban	1
612	1.0	Urban	1
613	0.0	Semiurban	0

[614 rows x 13 columns]

```
In [31]: from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.30)
```

```
In [32]: from sklearn.ensemble import RandomForestClassifier
rfc=RandomForestClassifier()
rfc.fit(x_train,y_train)
```

```
Out[32]: RandomForestClassifier()
```

```
In [33]: parameters={'max_depth':[1,2,3,4,5],  
                    'min_samples_leaf':[5,10,15,20,25],  
                    'n_estimators':[10,20,30,40,50]}
```

```
In [34]: 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[34]: 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 [35]: grid_search.best_score_
```

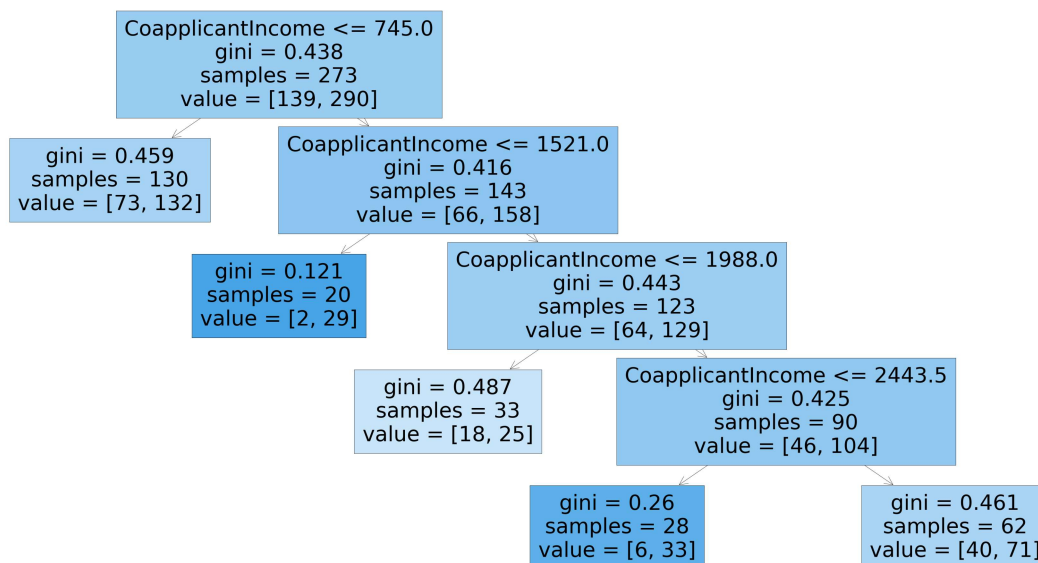
```
Out[35]: 0.6829710932405999
```

```
In [36]: rfc_best=grid_search.best_estimator_
```

In [37]: `from sklearn.tree import plot_tree`

```
plt.figure(figsize=(80,40))
plot_tree(rfc_best.estimators_[5],feature_names=x.columns,filled=True)
```

Out[37]: [Text(1275.4285714285713, 1956.96, 'CoapplicantIncome <= 745.0\n'gini = 0.438\n'nsamples = 273\n'value = [139, 290]'),
Text(637.7142857142857, 1522.0800000000002, 'gini = 0.459\n'nsamples = 130\n'value = [73, 132]'),
Text(1913.1428571428569, 1522.0800000000002, 'CoapplicantIncome <= 1521.0\n'gini = 0.416\n'nsamples = 143\n'value = [66, 158]'),
Text(1275.4285714285713, 1087.2, 'gini = 0.121\n'nsamples = 20\n'value = [2, 29]'),
Text(2550.8571428571427, 1087.2, 'CoapplicantIncome <= 1988.0\n'gini = 0.443\n'nsamples = 123\n'value = [64, 129]'),
Text(1913.1428571428569, 652.3200000000002, 'gini = 0.487\n'nsamples = 33\n'value = [18, 25]'),
Text(3188.5714285714284, 652.3200000000002, 'CoapplicantIncome <= 2443.5\n'gini = 0.425\n'nsamples = 90\n'value = [46, 104]'),
Text(2550.8571428571427, 217.44000000000005, 'gini = 0.26\n'nsamples = 28\n'value = [6, 33]'),
Text(3826.2857142857138, 217.44000000000005, 'gini = 0.461\n'nsamples = 62\n'value = [40, 71]')]



In []:

In []: