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

In [2]:
```
from sklearn.linear_model import LogisticRegression
```

In [3]:
```
df=pd.read_csv("C8 loan csv").dropna()

df
```

Out[3]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	Co
0	LP001015	Male	Yes	0	Graduate	No	5720	
1	LP001022	Male	Yes	1	Graduate	No	3076	
2	LP001031	Male	Yes	2	Graduate	No	5000	
4	LP001051	Male	No	0	Not Graduate	No	3276	
5	LP001054	Male	Yes	0	Not Graduate	Yes	2165	
...	...	...	...	...	...	...	...	
361	LP002969	Male	Yes	1	Graduate	No	2269	
362	LP002971	Male	Yes	3+	Not Graduate	Yes	4009	
363	LP002975	Male	Yes	0	Graduate	No	4158	
365	LP002986	Male	Yes	0	Graduate	No	5000	

In [4]:
```
df.dropna(inplace=True)
```

In [5]:
```
df['Education'].value_counts()
```

Out[5]:
```
Graduate        224
Not Graduate     65
Name: Education, dtype: int64
```

In [6]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 289 entries, 0 to 366
Data columns (total 12 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   Loan_ID            289 non-null    object 
 1   Gender             289 non-null    object 
 2   Married            289 non-null    object 
 3   Dependents         289 non-null    object 
 4   Education          289 non-null    object 
 5   Self_Employed      289 non-null    object 
 6   ApplicantIncome    289 non-null    int64  
 7   CoapplicantIncome  289 non-null    int64  
 8   LoanAmount         289 non-null    float64
 9   Loan_Amount_Term   289 non-null    float64
 10  Credit_History     289 non-null    float64
 11  Property_Area      289 non-null    object 
dtypes: float64(3), int64(2), object(7)
memory usage: 29.4+ KB
```

In [7]:
```
feature_matrix = df[['ApplicantIncome','CoapplicantIncome','LoanAmount','Loan_
target_vector = df['Property_Area']
```

In [8]:
```
feature_matrix.shape
```

Out[8]: (289, 5)

In [9]:
```
target_vector.shape
```

Out[9]: (289,)

In [10]:
```
from sklearn.preprocessing import StandardScaler
```

In [11]:
```
fs = StandardScaler().fit_transform(feature_matrix)
```

In [12]:
```
logr = LogisticRegression()
logr.fit(fs,target_vector)
```

Out[12]: LogisticRegression()

In [13]:
```
feature_matrix.shape
```

Out[13]: (289, 5)

In [14]:
```
target_vector.shape
```

Out[14]: (289,)

In [15]:
```
from sklearn.preprocessing import StandardScaler
```

In [16]:
```
fs = StandardScaler().fit_transform(feature_matrix)
```

In [17]:
```
logr = LogisticRegression()
logr.fit(fs,target_vector)
```

Out[17]:
```
LogisticRegression()
```

In [18]:
```
observation=df[['ApplicantIncome','CoapplicantIncome','LoanAmount','Loan_Amoun
```

In [19]:
```
prediction = logr.predict(observation)
prediction
```

Out[19]:
```
array(['Urban', 'Urban', 'Urban', 'Urban', 'Rural', 'Urban', 'Urban',
       'Rural', 'Urban', 'Urban', 'Urban', 'Rural', 'Urban', 'Rural',
       'Rural', 'Rural', 'Urban', 'Urban', 'Rural', 'Rural', 'Urban',
       'Rural', 'Urban', 'Urban', 'Urban', 'Rural', 'Urban', 'Urban',
       'Rural', 'Urban', 'Urban', 'Urban', 'Urban', 'Rural', 'Urban',
       'Urban', 'Rural', 'Urban', 'Rural', 'Rural', 'Urban', 'Urban',
       'Urban', 'Urban', 'Rural', 'Rural', 'Urban', 'Rural', 'Urban',
       'Urban', 'Rural', 'Urban', 'Rural', 'Urban', 'Urban', 'Urban',
       'Rural', 'Urban', 'Urban', 'Rural', 'Rural', 'Rural', 'Urban',
       'Urban', 'Urban', 'Urban', 'Urban', 'Urban', 'Urban', 'Rural',
       'Urban', 'Urban', 'Rural', 'Urban', 'Rural', 'Urban', 'Urban',
       'Urban', 'Rural', 'Rural', 'Urban', 'Urban', 'Urban', 'Urban',
       'Urban', 'Urban', 'Urban', 'Urban', 'Urban', 'Urban', 'Rural',
       'Rural', 'Rural', 'Rural', 'Rural', 'Urban', 'Rural', 'Urban',
       'Rural', 'Urban', 'Urban', 'Urban', 'Urban', 'Urban', 'Urban',
       'Urban', 'Urban', 'Urban', 'Urban', 'Urban', 'Urban', 'Rural',
       'Urban', 'Rural', 'Urban', 'Urban', 'Urban', 'Urban', 'Urban',
       'Urban', 'Rural', 'Urban', 'Urban', 'Rural', 'Urban', 'Rural',
       'Rural', 'Urban', 'Urban', 'Urban', 'Rural', 'Urban', 'Urban',
```

In [20]:
```
logr.classes_
```

Out[20]:
```
array(['Rural', 'Semiurban', 'Urban'], dtype=object)
```

In [21]:
```
logr.predict_proba(observation)[0][1]
```

Out[21]:
```
0.0
```

random Forest

In [22]:
```
df['Education'].value_counts()
```

Out[22]:
```
Graduate        224
Not Graduate     65
Name: Education, dtype: int64
```

In [23]:
```
x=df[['ApplicantIncome','CoapplicantIncome','LoanAmount','Loan_Amount_Term','C
y=df['Education']
```

In [24]:
```
g1={'Education':{"Graduate":1, "Not Graduate":2}}
df=df.replace(g1)
df
```

Out[24]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	Coap
0	LP001015	Male	Yes	0	1	No	5720	
1	LP001022	Male	Yes	1	1	No	3076	
2	LP001031	Male	Yes	2	1	No	5000	
4	LP001051	Male	No	0	2	No	3276	
5	LP001054	Male	Yes	0	2	Yes	2165	
...	...	...	...	...	...	...	...	
361	LP002969	Male	Yes	1	1	No	2269	
362	LP002971	Male	Yes	3+	2	Yes	4009	
363	LP002975	Male	Yes	0	1	No	4158	
365	LP002986	Male	Yes	0	1	No	5000	
366	LP002989	Male	No	0	1	Yes	9200	

289 rows × 12 columns

In [25]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,train_size=0.70)
```

In [26]:
```
from sklearn.ensemble import RandomForestClassifier
rfc = RandomForestClassifier()
rfc.fit(x_train,y_train)
```

Out[26]:
```
RandomForestClassifier()
```

In [27]:
```
parameters = {'max_depth':[1,2,3,4,5],'min_samples_leaf':[5,10,15,20,25],
              'n_estimators': [10,20,30,40,50]
              }
```

In [28]:
```
from sklearn.model_selection import GridSearchCV
grid_search = GridSearchCV(estimator=rfc,param_grid=parameters,cv=2,scoring="a
grid_search.fit(x_train,y_train)
```

Out[28]:
```
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')
```

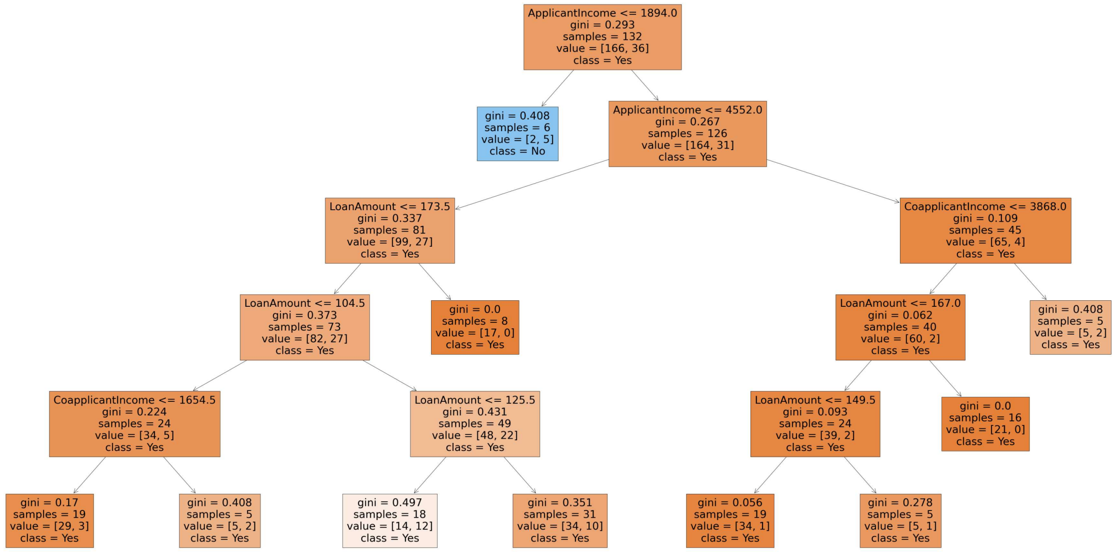
In [29]:
```
grid_search.best_score_
```

Out[29]: 0.801980198019802

In [30]:
```
rfc_best = grid_search.best_estimator_
```

In [31]:
```
from sklearn.tree import plot_tree
plt.figure(figsize = (80,40,))
plot_tree(rfc_best.estimators_[5],feature_names=x.columns,class_names=['Yes','
```

Out[31]:
```
[Text(2391.428571428571, 1993.2, 'ApplicantIncome <= 1894.0\ngini = 0.293\nsa
mples = 132\nvalue = [166, 36]\nclass = Yes'),
 Text(2072.5714285714284, 1630.8000000000002, 'gini = 0.408\nsamples = 6\nval
ue = [2, 5]\nclass = No'),
 Text(2710.285714285714, 1630.8000000000002, 'ApplicantIncome <= 4552.0\ngini
= 0.267\nsamples = 126\nvalue = [164, 31]\nclass = Yes'),
 Text(1594.2857142857142, 1268.4, 'LoanAmount <= 173.5\ngini = 0.337\nsamples
= 81\nvalue = [99, 27]\nclass = Yes'),
 Text(1275.4285714285713, 906.0, 'LoanAmount <= 104.5\ngini = 0.373\nsamples
= 73\nvalue = [82, 27]\nclass = Yes'),
 Text(637.7142857142857, 543.5999999999999, 'CoapplicantIncome <= 1654.5\ngin
i = 0.224\nsamples = 24\nvalue = [34, 5]\nclass = Yes'),
 Text(318.85714285714283, 181.19999999999982, 'gini = 0.17\nsamples = 19\nval
ue = [29, 3]\nclass = Yes'),
 Text(956.5714285714284, 181.19999999999982, 'gini = 0.408\nsamples = 5\nvalu
e = [5, 2]\nclass = Yes'),
 Text(1913.1428571428569, 543.5999999999999, 'LoanAmount <= 125.5\ngini = 0.4
31\nsamples = 49\nvalue = [48, 22]\nclass = Yes'),
 Text(1594.2857142857142, 181.19999999999982, 'gini = 0.497\nsamples = 18\nva
lue = [14, 12]\nclass = Yes'),
 Text(2232.0, 181.19999999999982, 'gini = 0.351\nsamples = 31\nvalue = [34, 1
0]\nclass = Yes'),
 Text(1913.1428571428569, 906.0, 'gini = 0.0\nsamples = 8\nvalue = [17, 0]\nc
lass = Yes'),
 Text(3826.2857142857138, 1268.4, 'CoapplicantIncome <= 3868.0\ngini = 0.109
\nsamples = 45\nvalue = [65, 4]\nclass = Yes'),
 Text(3507.428571428571, 906.0, 'LoanAmount <= 167.0\ngini = 0.062\nsamples =
40\nvalue = [60, 2]\nclass = Yes'),
 Text(3188.5714285714284, 543.5999999999999, 'LoanAmount <= 149.5\ngini = 0.0
93\nsamples = 24\nvalue = [39, 2]\nclass = Yes'),
 Text(2869.7142857142853, 181.19999999999982, 'gini = 0.056\nsamples = 19\nva
lue = [34, 1]\nclass = Yes'),
 Text(3507.428571428571, 181.19999999999982, 'gini = 0.278\nsamples = 5\nvalu
e = [5, 1]\nclass = Yes'),
 Text(3826.2857142857138, 543.5999999999999, 'gini = 0.0\nsamples = 16\nvalue
= [21, 0]\nclass = Yes'),
 Text(4145.142857142857, 906.0, 'gini = 0.408\nsamples = 5\nvalue = [5, 2]\nc
lass = Yes')]
```


ApplicantIncome <= 1894.0
gini = 0.293
samples = 132
value = [166, 36]
class = Yes
gini = 0.408
samples = 6
value = [2, 5]
class = No
ApplicantIncome <= 4552.0
gini = 0.267
samples = 126
value = [164, 31]
class = Yes
LoanAmount <= 173.5
gini = 0.337
samples = 81
value = [99, 27]
class = Yes
CoapplicantIncome <= 3868.0
gini = 0.109
samples = 45
value = [65, 4]
class = Yes
LoanAmount <= 104.5
gini = 0.373
samples = 73
value = [82, 27]
class = Yes
gini = 0.0
samples = 8
value = [17, 0]
class = Yes
LoanAmount <= 167.0
gini = 0.062
samples = 40
value = [60, 2]
class = Yes
gini = 0.408
samples = 5
value = [5, 2]
class = Yes
CoapplicantIncome <= 1654.5
gini = 0.224
samples = 24
value = [34, 5]
class = Yes
LoanAmount <= 125.5
gini = 0.431
samples = 49
value = [48, 22]
class = Yes
LoanAmount <= 149.5
gini = 0.093
samples = 24
value = [39, 2]
class = Yes
gini = 0.0
samples = 16
value = [21, 0]
class = Yes
gini = 0.17
samples = 19
value = [29, 3]
class = Yes
gini = 0.408
samples = 5
value = [5, 2]
class = Yes
gini = 0.497
samples = 18
value = [14, 12]
class = Yes
gini = 0.351
samples = 31
value = [34, 10]
class = Yes
gini = 0.056
samples = 19
value = [34, 1]
class = Yes
gini = 0.278
samples = 5
value = [5, 1]
class = Yes