In [3]:

```
import numpy  as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [57]:

```
a=pd.read_csv(r"C:\Users\user\Downloads\C8_loan-test.csv")
a
```

Out[57]:

| | Loan_ID | Gender | Married | Dependents | Education | Self_Employed | ApplicantIncome | C |
|---|---|---|---|---|---|---|---|---|
| **0** | LP001015 | Male | Yes | 0 | Graduate | No | 5720 | |
| **1** | LP001022 | Male | Yes | 1 | Graduate | No | 3076 | |
| **2** | LP001031 | Male | Yes | 2 | Graduate | No | 5000 | |
| **3** | LP001035 | Male | Yes | 2 | Graduate | No | 2340 | |
| **4** | LP001051 | Male | No | 0 | Not Graduate | No | 3276 | |
| **...** | ... | ... | ... | ... | ... | ... | ... | |
| **362** | LP002971 | Male | Yes | 3+ | Not Graduate | Yes | 4009 | |
| **363** | LP002975 | Male | Yes | 0 | Graduate | No | 4158 | |
| **364** | LP002980 | Male | No | 0 | Graduate | No | 3250 | |
| **365** | LP002986 | Male | Yes | 0 | Graduate | No | 5000 | |
| **366** | LP002989 | Male | No | 0 | Graduate | Yes | 9200 | |

367 rows × 12 columns

In [58]:

```
from sklearn.linear_model import LogisticRegression
```

In [61]:

```
a=a.head(10)
a
```

Out[61]:

| Education | Self_Employed | ApplicantIncome | CoapplicantIncome | LoanAmount | Loan_Amount_Ter |
|---|---|---|---|---|---|
| Graduate | No | 5720 | 0 | 110.0 | 360 |
| Graduate | No | 3076 | 1500 | 126.0 | 360 |
| Graduate | No | 5000 | 1800 | 208.0 | 360 |
| Graduate | No | 2340 | 2546 | 100.0 | 360 |
| Not Graduate | No | 3276 | 0 | 78.0 | 360 |
| Not Graduate | Yes | 2165 | 3422 | 152.0 | 360 |
| Not Graduate | No | 2226 | 0 | 59.0 | 360 |
| Not Graduate | No | 3881 | 0 | 147.0 | 360 |
| Graduate | NaN | 13633 | 0 | 280.0 | 240 |
| Not Graduate | No | 2400 | 2400 | 123.0 | 360 |

In [62]:

```
a.columns
```

Out[62]:

```
Index(['Loan_ID', 'Gender', 'Married', 'Dependents', 'Education',
       'Self_Employed', 'ApplicantIncome', 'CoapplicantIncome', 'LoanAmoun
t',
       'Loan_Amount_Term', 'Credit_History', 'Property_Area'],
      dtype='object')
```

In [64]:

```
b=a[['Dependents', 'ApplicantIncome', 'CoapplicantIncome', 'LoanAmount',
       'Loan_Amount_Term']]
b
```

Out[64]:

| | Dependents | ApplicantIncome | CoapplicantIncome | LoanAmount | Loan_Amount_Term |
|---|---|---|---|---|---|
| **0** | 0 | 5720 | 0 | 110.0 | 360.0 |
| **1** | 1 | 3076 | 1500 | 126.0 | 360.0 |
| **2** | 2 | 5000 | 1800 | 208.0 | 360.0 |
| **3** | 2 | 2340 | 2546 | 100.0 | 360.0 |
| **4** | 0 | 3276 | 0 | 78.0 | 360.0 |
| **5** | 0 | 2165 | 3422 | 152.0 | 360.0 |
| **6** | 1 | 2226 | 0 | 59.0 | 360.0 |
| **7** | 2 | 3881 | 0 | 147.0 | 360.0 |
| **8** | 2 | 13633 | 0 | 280.0 | 240.0 |
| **9** | 0 | 2400 | 2400 | 123.0 | 360.0 |

In [65]:

```
c=b.iloc[:,0:11]
d=a.iloc[:,-1]
```

In [66]:

```
c.shape
```

Out[66]:

```
(10, 5)
```

In [67]:

```
d.shape
```

Out[67]:

```
(10,)
```

In [68]:

```
from sklearn.preprocessing import StandardScaler
```

In [69]:

```
fs=StandardScaler().fit_transform(c)
```

In [70]:

```
logr=LogisticRegression()
logr.fit(fs,d)
```

Out[70]:

```
LogisticRegression()
```

In [71]:

```
e=[[2,5,77,5,7]]
```

In [72]:

```
prediction=logr.predict(e)
prediction
```

Out[72]:

```
array(['Urban'], dtype=object)
```

In [73]:

```
logr.classes_
```

Out[73]:

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

In [74]:

```
logr.predict_proba(e)[0][0]
```

Out[74]:

```
3.893396859975968e-31
```

In [75]:

```
logr.predict_proba(e)[0][1]
```

Out[75]:

```
6.319103730318476e-12
```

In [76]:

```
import re
from sklearn.datasets import load_digits
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import sklearn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
```

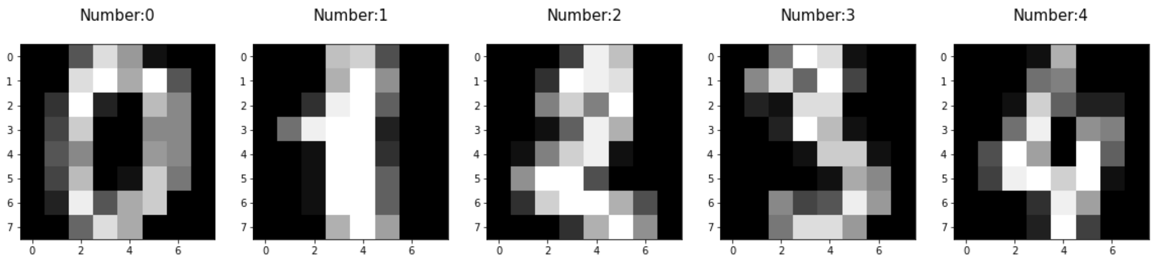
In [77]:

```
digits=load_digits()
digits
```

```
  'pixel_6_3',
  'pixel_6_4',
  'pixel_6_5',
  'pixel_6_6',
  'pixel_6_7',
  'pixel_7_0',
  'pixel_7_1',
  'pixel_7_2',
  'pixel_7_3',
  'pixel_7_4',
  'pixel_7_5',
  'pixel_7_6',
  'pixel_7_7'],
 'target_names': array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9]),
 'images': array([[[ 0.,  0.,  5., ...,  1.,  0.,  0.],
         [ 0.,  0., 13., ..., 15.,  5.,  0.],
         [ 0.,  3., 15., ..., 11.,  8.,  0.],
         ...,
         [ 0.,  4., 11., ..., 12.,  7.,  0.],
         [ 0.,  2., 14., ..., 12.,  0.,  0.],
```

In [78]:

```
plt.figure(figsize=(20,4))
for index,(image,label)in enumerate(zip(digits.data[0:5],digits.target[0:5])):
    plt.subplot(1,5,index+1)
    plt.imshow(np.reshape(image,(8,8)),cmap=plt.cm.gray)
    plt.title('Number:%i\n'%label,fontsize=15)
```




In [79]:

```
x_train,x_test,y_train,y_test=train_test_split(digits.data,digits.target,test_size=0.30)
```

In [80]:

```
print(x_train.shape)
print(x_test.shape)
print(y_train.shape)
print(y_test.shape)
```

```
(1257, 64)
(540, 64)
(1257,)
(540,)
```

In [81]:

```
logre=LogisticRegression(max_iter=10000)
logre.fit(x_train,y_train)
```

Out[81]:

```
LogisticRegression(max_iter=10000)
```

In [82]:

```
logre.predict(x_test)
```

Out[82]:

```
array([1, 1, 9, 9, 0, 2, 3, 3, 9, 0, 3, 9, 3, 5, 4, 0, 9, 8, 7, 3, 4, 8,
       5, 5, 7, 7, 3, 7, 5, 3, 9, 9, 7, 7, 5, 5, 5, 9, 5, 2, 6, 0, 7, 4,
       4, 0, 5, 6, 3, 6, 8, 6, 2, 3, 1, 7, 2, 0, 3, 1, 8, 4, 6, 3, 8, 2,
       1, 4, 7, 9, 8, 5, 4, 8, 5, 1, 1, 4, 5, 2, 2, 7, 9, 9, 8, 5, 6, 1,
       0, 7, 9, 5, 1, 8, 3, 7, 0, 4, 3, 6, 1, 3, 6, 0, 3, 9, 1, 8, 5, 8,
       5, 0, 0, 7, 6, 7, 4, 0, 4, 5, 7, 9, 6, 8, 6, 8, 2, 2, 1, 0, 1, 6,
       4, 3, 3, 6, 4, 4, 2, 1, 8, 2, 5, 8, 5, 3, 7, 7, 0, 3, 0, 2, 4, 2,
       3, 1, 8, 1, 6, 4, 9, 5, 4, 2, 8, 0, 4, 5, 6, 0, 4, 1, 3, 9, 8, 3,
       9, 0, 4, 0, 9, 1, 4, 0, 6, 6, 1, 1, 7, 8, 0, 0, 8, 7, 7, 8, 1, 2,
       2, 1, 3, 1, 6, 0, 3, 7, 1, 4, 4, 7, 9, 8, 0, 8, 0, 5, 2, 6, 6, 4,
       6, 1, 8, 9, 7, 7, 7, 5, 7, 5, 4, 8, 5, 6, 6, 0, 3, 7, 3, 7, 2, 6,
       9, 6, 8, 9, 1, 9, 7, 8, 3, 5, 3, 3, 0, 6, 4, 7, 4, 2, 2, 5, 0, 6,
       7, 8, 3, 6, 7, 5, 6, 2, 7, 9, 0, 1, 7, 7, 3, 7, 6, 8, 7, 0, 6, 0,
       7, 8, 6, 0, 5, 5, 8, 9, 2, 4, 0, 1, 5, 5, 4, 8, 9, 1, 0, 4, 8, 7,
       2, 0, 7, 4, 1, 9, 3, 2, 1, 8, 7, 8, 3, 3, 0, 5, 9, 4, 5, 5, 6, 2,
       0, 4, 1, 8, 9, 5, 4, 1, 7, 6, 3, 1, 8, 4, 3, 8, 9, 9, 1, 7, 0, 5,
       2, 1, 5, 5, 6, 6, 3, 7, 5, 7, 9, 9, 7, 0, 1, 4, 0, 4, 1, 9, 2, 9,
       4, 6, 8, 8, 6, 3, 4, 7, 4, 2, 7, 9, 7, 4, 8, 9, 0, 1, 4, 9, 4, 7,
       5, 1, 6, 1, 7, 4, 4, 3, 3, 1, 4, 8, 9, 0, 6, 9, 0, 2, 4, 9, 2, 3,
       4, 7, 0, 8, 1, 3, 1, 3, 3, 4, 9, 7, 6, 6, 1, 3, 0, 8, 2, 2, 1, 9,
       2, 3, 9, 4, 2, 2, 2, 8, 5, 8, 5, 1, 7, 0, 8, 0, 0, 5, 4, 5, 2, 8,
       1, 4, 8, 2, 1, 3, 7, 9, 0, 9, 5, 9, 6, 9, 4, 1, 2, 3, 9, 6, 0, 2,
       4, 3, 2, 1, 5, 3, 1, 3, 6, 2, 3, 9, 7, 7, 5, 1, 8, 5, 1, 0, 5, 6,
       7, 5, 3, 3, 4, 1, 2, 3, 2, 6, 4, 2, 5, 6, 8, 4, 6, 0, 4, 2, 1, 6,
       1, 9, 2, 3, 6, 4, 8, 6, 0, 7, 3, 5])
```

In [83]:

```
logre.score(x_test,y_test)
```

Out[83]:

```
0.9537037037037037
```

In [84]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [90]:

```
b=a.head(10)
b
```

Out[90]:

| | Loan_ID | Gender | Married | Dependents | Education | Self_Employed | ApplicantIncome | Coa |
|---|---|---|---|---|---|---|---|---|
| **0** | LP001015 | Male | Yes | 0 | Graduate | No | 5720 | |
| **1** | LP001022 | Male | Yes | 1 | Graduate | No | 3076 | |
| **2** | LP001031 | Male | Yes | 2 | Graduate | No | 5000 | |
| **3** | LP001035 | Male | Yes | 2 | Graduate | No | 2340 | |
| **4** | LP001051 | Male | No | 0 | Not Graduate | No | 3276 | |
| **5** | LP001054 | Male | Yes | 0 | Not Graduate | Yes | 2165 | |
| **6** | LP001055 | Female | No | 1 | Not Graduate | No | 2226 | |
| **7** | LP001056 | Male | Yes | 2 | Not Graduate | No | 3881 | |
| **8** | LP001059 | Male | Yes | 2 | Graduate | NaN | 13633 | |
| **9** | LP001067 | Male | No | 0 | Not Graduate | No | 2400 | |

In [91]:

```
b=b[['Dependents', 'ApplicantIncome', 'CoapplicantIncome', 'LoanAmount',
     'Loan_Amount_Term','Property_Area']]
b
```

Out[91]:

| | Dependents | ApplicantIncome | CoapplicantIncome | LoanAmount | Loan_Amount_Term | Prope |
|---|---|---|---|---|---|---|
| **0** | 0 | 5720 | 0 | 110.0 | 360.0 | |
| **1** | 1 | 3076 | 1500 | 126.0 | 360.0 | |
| **2** | 2 | 5000 | 1800 | 208.0 | 360.0 | |
| **3** | 2 | 2340 | 2546 | 100.0 | 360.0 | |
| **4** | 0 | 3276 | 0 | 78.0 | 360.0 | |
| **5** | 0 | 2165 | 3422 | 152.0 | 360.0 | |
| **6** | 1 | 2226 | 0 | 59.0 | 360.0 | S |
| **7** | 2 | 3881 | 0 | 147.0 | 360.0 | |
| **8** | 2 | 13633 | 0 | 280.0 | 240.0 | |
| **9** | 0 | 2400 | 2400 | 123.0 | 360.0 | S |

In [92]:

```
b['Property_Area'].value_counts()
```

Out[92]:

```
Urban        7
Semiurban    2
Rural        1
Name: Property_Area, dtype: int64
```

In [93]:

```
x=b.drop('Property_Area',axis=1)
y=b['Property_Area']
```

In [116]:

```
g1={"Property_Area":{'Urban':1,'Semiurban ':2,'Rural':3}}
b=b.replace(g1)
print(b)
```

```
  Dependents  ApplicantIncome  CoapplicantIncome  LoanAmount  \
0          0             5720                  0       110.0
1          1             3076               1500       126.0
2          2             5000               1800       208.0
3          2             2340               2546       100.0
4          0             3276                  0        78.0
5          0             2165               3422       152.0
6          1             2226                  0        59.0
7          2             3881                  0       147.0
8          2            13633                  0       280.0
9          0             2400               2400       123.0

   Loan_Amount_Term Property_Area
0             360.0             1
1             360.0             1
2             360.0             1
3             360.0             1
4             360.0             1
5             360.0             1
6             360.0     Semiurban
7             360.0             3
8             240.0             1
9             360.0     Semiurban
```

In [117]:

```
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 [118]:

```
from sklearn.ensemble import RandomForestClassifier
```

In [119]:

```
rfc=RandomForestClassifier()
rfc.fit(x_train,y_train)
```

Out[119]:

```
RandomForestClassifier()
```

In [120]:

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

In [121]:

```
from sklearn.model_selection import GridSearchCV
```

In [122]:

```
grid_search=GridSearchCV(estimator=rfc,param_grid=parameters,cv=2,scoring="accuracy")
grid_search.fit(x_train,y_train)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\model_selection\_split.
py:666: UserWarning: The least populated class in y has only 1 members, wh
ich is less than n_splits=2.
  warnings.warn(("The least populated class in y has only %d"
```

Out[122]:

```
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 [123]:

```
grid_search.best_score_
```

Out[123]:

```
0.75
```

In [124]:

```
rfc_best=grid_search.best_estimator_
```

In [125]:

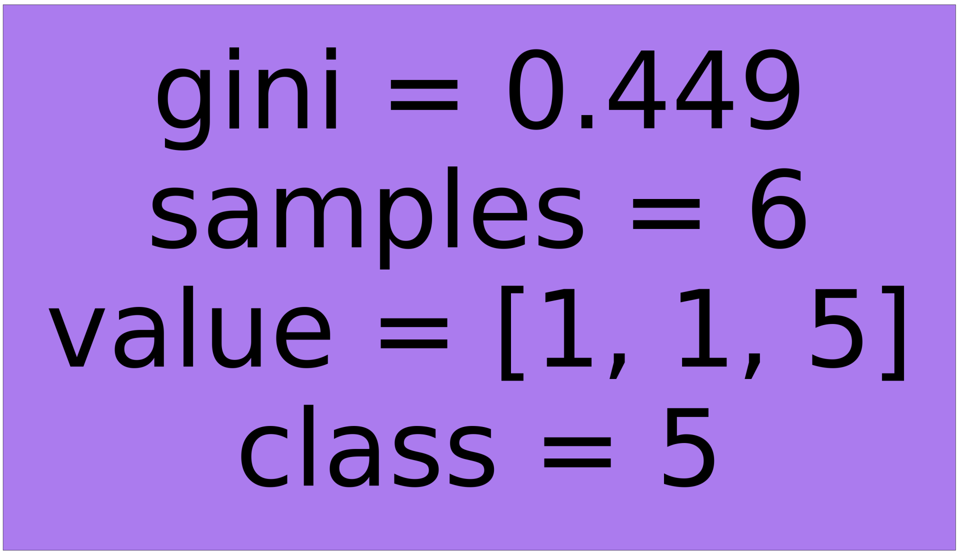
```
from sklearn.tree import plot_tree
```

In [127]:

```
plt.figure(figsize=(80,40))
plot_tree(rfc_best.estimators_[5],feature_names=x.columns,class_names=['Yes','No','5'],fi
```

Out[127]:

```
[Text(2232.0, 1087.2, 'gini = 0.449\nsamples = 6\nvalue = [1, 1, 5]\nclass
= 5')]
```




In [ ]:

```
```