In [1]:

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

In [2]:

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

Out[2]:

	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	
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 [3]:

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

In [4]:

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

Out[4]:

	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 [5]:

```
a.columns
```

Out[5]:

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

In [6]:

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

Out[6]:

	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 [7]:

```
c=b.iloc[:,0:5]
d=a.iloc[:,-1]
```

In [8]:

```
c.shape
```

Out[8]:

```
(10, 5)
```

In [9]:

```
d.shape
```

Out[9]:

```
(10,)
```

In [11]:

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

In [12]:

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

In [14]:

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

In [15]:

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

Out[15]:

```
LogisticRegression()
```

In [16]:

```
e=[[2,5,77,8,65]]
```

In [17]:

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

Out[17]:

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

In [18]:

```
logr.classes_
```

Out[18]:

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

In [19]:

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

Out[19]:

```
2.4832907804011954e-23
```

In [20]:

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

Out[20]:

```
1.2487144392441901e-08
```

In [21]:

```
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
```

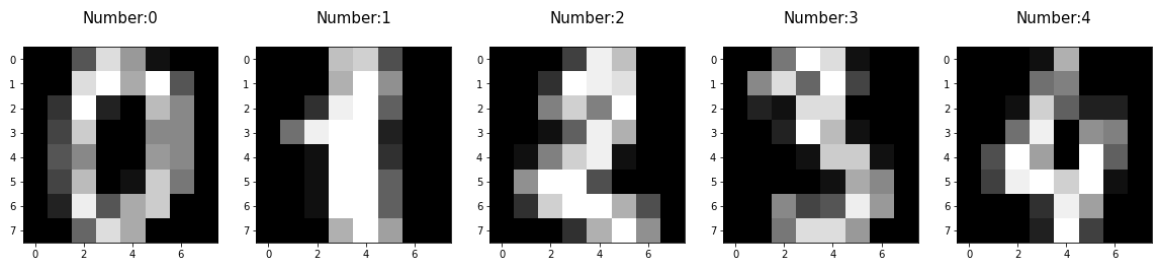
In [22]:

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

```
  'pixel_3_7',
  'pixel_4_0',
  'pixel_4_1',
  'pixel_4_2',
  'pixel_4_3',
  'pixel_4_4',
  'pixel_4_5',
  'pixel_4_6',
  'pixel_4_7',
  'pixel_5_0',
  'pixel_5_1',
  'pixel_5_2',
  'pixel_5_3',
  'pixel_5_4',
  'pixel_5_5',
  'pixel_5_6',
  'pixel_5_7',
  'pixel_6_0',
  'pixel_6_1',
```

In [23]:

```
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 [24]:

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

In [25]:

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

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

In [26]:

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

Out[26]:

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

In [27]:

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

Out[27]:

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

In [28]:

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

Out[28]:

```
0.9518518518518518
```

In [ ]: