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-train.csv")
a
```

Out[2]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	Co
0	LP001002	Male	No	0	Graduate	No	5849	
1	LP001003	Male	Yes	1	Graduate	No	4583	
2	LP001005	Male	Yes	0	Graduate	Yes	3000	
3	LP001006	Male	Yes	0	Not Graduate	No	2583	
4	LP001008	Male	No	0	Graduate	No	6000	
...	...	...	...	...	...	...	...	
609	LP002978	Female	No	0	Graduate	No	2900	
610	LP002979	Male	Yes	3+	Graduate	No	4106	
611	LP002983	Male	Yes	1	Graduate	No	8072	
612	LP002984	Male	Yes	2	Graduate	No	7583	
613	LP002990	Female	No	0	Graduate	Yes	4583	

614 rows × 13 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	LP001002	Male	No	0	Graduate	No	5849	
1	LP001003	Male	Yes	1	Graduate	No	4583	
2	LP001005	Male	Yes	0	Graduate	Yes	3000	
3	LP001006	Male	Yes	0	Not Graduate	No	2583	
4	LP001008	Male	No	0	Graduate	No	6000	
5	LP001011	Male	Yes	2	Graduate	Yes	5417	
6	LP001013	Male	Yes	0	Not Graduate	No	2333	
7	LP001014	Male	Yes	3+	Graduate	No	3036	
8	LP001018	Male	Yes	2	Graduate	No	4006	
9	LP001020	Male	Yes	1	Graduate	No	12841	

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', 'Loan_Statu
s'],
      dtype='object')
```

In [18]:

```
b=a[['ApplicantIncome','CoapplicantIncome','Loan_Amount_Term', 'Credit_History']]
b
```

Out[18]:

	ApplicantIncome	CoapplicantIncome	Loan_Amount_Term	Credit_History
0	5849	0.0	360.0	1.0
1	4583	1508.0	360.0	1.0
2	3000	0.0	360.0	1.0
3	2583	2358.0	360.0	1.0
4	6000	0.0	360.0	1.0
5	5417	4196.0	360.0	1.0
6	2333	1516.0	360.0	1.0
7	3036	2504.0	360.0	0.0
8	4006	1526.0	360.0	1.0
9	12841	10968.0	360.0	1.0

In [19]:

```
c=b.iloc[:,0:4]
d=a.iloc[:,-1]
```

In [20]:

```
c.shape
```

Out[20]:

```
(10, 4)
```

In [21]:

```
d.shape
```

Out[21]:

```
(10,)
```

In [22]:

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

In [23]:

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

In [24]:

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

In [25]:

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

Out[25]:

```
LogisticRegression()
```

In [26]:

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

In [27]:

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

Out[27]:

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

In [28]:

```
logr.classes_
```

Out[28]:

```
array(['N', 'Y'], dtype=object)
```

In [29]:

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

Out[29]:

```
0.02338273383549161
```

In [30]:

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

Out[30]:

```
0.9766172661645084
```

In [31]:

```
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
```

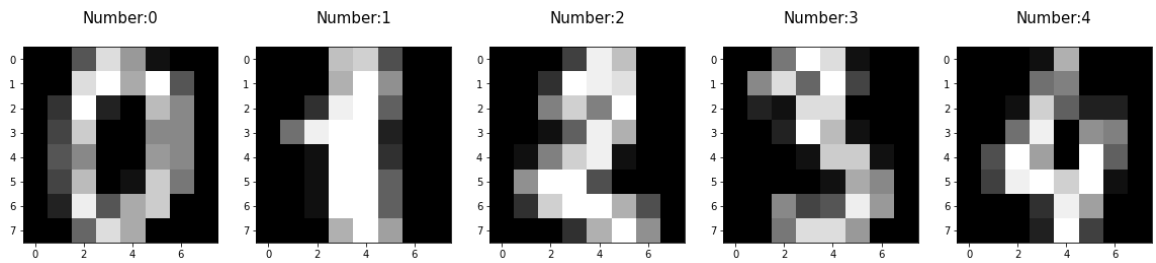
In [32]:

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

```
  'pixel_0_4',
  'pixel_0_5',
  'pixel_0_6',
  'pixel_0_7',
  'pixel_1_0',
  'pixel_1_1',
  'pixel_1_2',
  'pixel_1_3',
  'pixel_1_4',
  'pixel_1_5',
  'pixel_1_6',
  'pixel_1_7',
  'pixel_2_0',
  'pixel_2_1',
  'pixel_2_2',
  'pixel_2_3',
  'pixel_2_4',
  'pixel_2_5',
  'pixel_2_6',
```

In [33]:

```
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 [34]:

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

In [35]:

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

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

In [36]:

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

Out[36]:

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

In [37]:

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

Out[37]:

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

In [38]:

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

Out[38]:

```
0.9648148148148148
```

In [ ]: