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\C9_Data.csv")
a
```

Out[2]:

	row_id	user_id	timestamp	gate_id
0	0	18	2022-07-29 09:08:54	7
1	1	18	2022-07-29 09:09:54	9
2	2	18	2022-07-29 09:09:54	9
3	3	18	2022-07-29 09:10:06	5
4	4	18	2022-07-29 09:10:08	5
...	...	...	...	...
37513	37513	6	2022-12-31 20:38:56	11
37514	37514	6	2022-12-31 20:39:22	6
37515	37515	6	2022-12-31 20:39:23	6
37516	37516	6	2022-12-31 20:39:31	9
37517	37517	6	2022-12-31 20:39:31	9

37518 rows × 4 columns

In [3]:

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

In [11]:

```
b=a[['row_id','user_id','gate_id']]
b
```

Out[11]:

	row_id	user_id	gate_id
0	0	18	7
1	1	18	9
2	2	18	9
3	3	18	5
4	4	18	5
5	5	18	10
6	6	18	11
7	7	18	4
8	8	18	4
9	9	1	7

In [12]:

```
c=b.iloc[:,0:3]
d=a.iloc[:,-1]
```

In [13]:

```
c.shape
```

Out[13]:

```
(10, 3)
```

In [14]:

```
d.shape
```

Out[14]:

```
(10,)
```

In [15]:

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

In [16]:

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

In [17]:

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

In [18]:

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

Out[18]:

```
LogisticRegression()
```

In [20]:

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

In [21]:

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

Out[21]:

```
array([11], dtype=int64)
```

In [22]:

```
logr.classes_
```

Out[22]:

```
array([ 4,  5,  7,  9, 10, 11], dtype=int64)
```

In [23]:

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

Out[23]:

```
1.1739426061051213e-52
```

In [24]:

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

Out[24]:

```
1.1014934484995237e-52
```

In [25]:

```
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
```

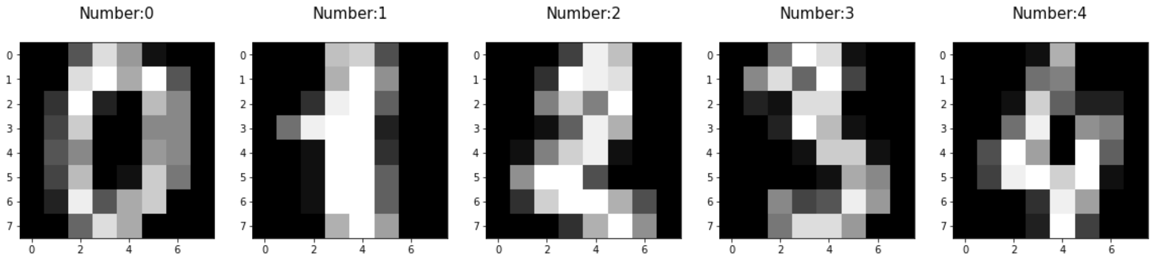
In [26]:

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

```
 'pixel_1_7',
 'pixel_2_0',
 'pixel_2_1',
 'pixel_2_2',
 'pixel_2_3',
 'pixel_2_4',
 'pixel_2_5',
 'pixel_2_6',
 'pixel_2_7',
 'pixel_3_0',
 'pixel_3_1',
 'pixel_3_2',
 'pixel_3_3',
 'pixel_3_4',
 'pixel_3_5',
 'pixel_3_6',
 'pixel_3_7',
 'pixel_4_0',
 'pixel_4_1',
 'pixel_4_2',
 'pixel_4_3',
```

In [27]:

```
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 [28]:

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

In [29]:

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

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

In [30]:

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

Out[30]:

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

In [31]:

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

Out[31]:

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

In [32]:

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

Out[32]:

```
0.9537037037037037
```

In [ ]:

```

```